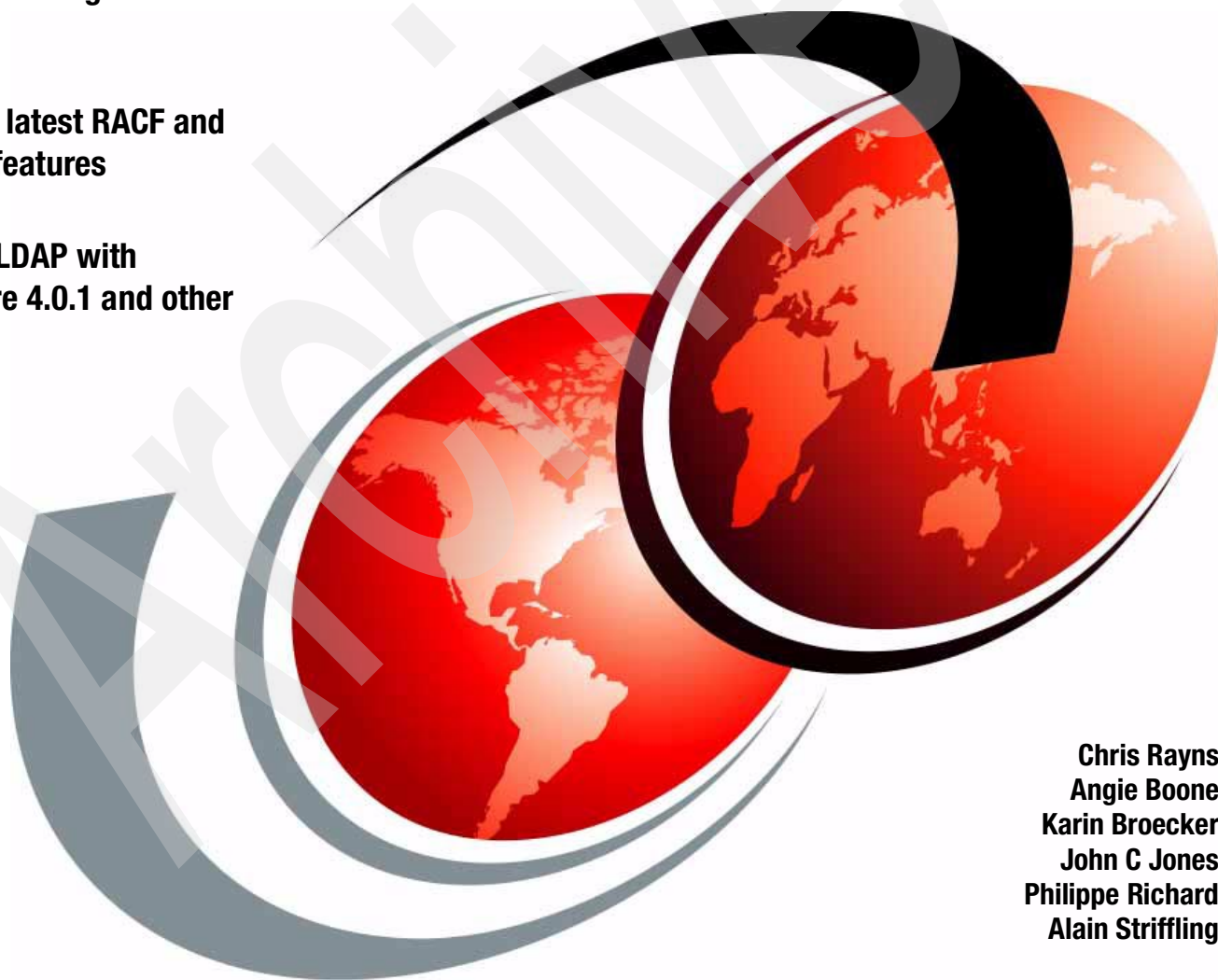


Putting the Latest z/OS Security Features to Work

Install and configure an LDAP server

Utilize the latest RACF and
Kerberos features

Integrate LDAP with
WebSphere 4.0.1 and other
products



Chris Rayns
Angie Boone
Karin Broecker
John C Jones
Philippe Richard
Alain Striffling

Redbooks



International Technical Support Organization

Putting the Latest z/OS Security Features to Work

June 2002

Archived

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (June 2002)

This edition applies to SecureWay Security Server for z/OS Version 1 Release 2, for use with the z/OS Operating System.

© Copyright International Business Machines Corporation 2002. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Preface	ix
The team that wrote this redbook.	ix
Become a published author	x
Comments welcome.	xi
Part 1. z/OS 1.2 enhancements	1
Chapter 1. z/OS 1.2 LDAP Directory enhancements	3
1.1 LDAP client enhancements	4
1.2 LDAP server enhancements	5
1.2.1 Server front-end performance/scalability	6
1.2.2 SDBM enhancements	7
1.2.3 TDBM Native authentication	9
1.2.4 Kerberos authentication	10
1.3 Miscellaneous changes	12
Part 2. OS/390 2.10 enhancements	13
Chapter 2. RACF enhancements	15
2.1 Program control enhancements	16
2.1.1 Overview	16
2.1.2 Introduction	16
2.1.3 Description	16
2.1.4 RACF services	17
2.1.5 Program Control enhancements example errors	18
2.2 Application Identity Mapping (AIM)	20
2.2.1 Overview	20
2.2.2 RACF utilities	21
2.2.3 AIM's purpose	21
2.2.4 Benefits of AIM	22
2.2.5 Migrating and installation of AIM	23
2.2.6 Stage enablement	23
2.2.7 Installation	24
2.2.8 Space requirements	30
2.2.9 When there are more than 130 user IDs with the same UID	31
2.2.10 New Serialization	33
2.3 Digital certificate enhancements	33
2.3.1 Digital certificate enhancements in Release 10	34
2.3.2 Generating a digital certificate with RACF	36
2.3.3 Installing a RACF digital certificate into your browser	38
2.4 PKIServ - Web interface to certificate generation	48
2.4.1 Overview	48
2.4.2 Directory structure provided	50
2.4.3 The configuration file	51
2.4.4 User interface	53
2.4.5 Installation and configuration	69

2.5 OS/390 UNIX Superuser granularity support	79
2.6 Service Updates	79
2.6.1 APAR OW39128	79
2.6.2 APAR OW38799	79
2.6.3 APAR OW42092	80
2.7 RVARY command enhancement	80
2.7.1 Introduction	80
2.7.2 Logical console security	81
2.7.3 RVARY console samples	81
Chapter 3. Network Authentication and Privacy Service	85
3.1 Introduction to Kerberos	86
3.1.1 Kerberos protocol overview	86
3.1.2 Inter-realm operation	91
3.1.3 Some assumptions	92
3.2 Implementing NAPS	92
3.2.1 SKRBKDC daemon setup	92
3.2.2 Setting up the Kerberos environment variable files	97
3.2.3 Setting up HFS for Kerberos cache files	98
3.3 Kerberos integrated with RACF	98
3.3.1 Defining RRSF in local mode	99
3.3.2 RACF setup for Kerberos realms	101
3.4 Kerberos principals	104
3.4.1 Local principals	104
3.4.2 Foreign principals	107
3.5 Kerberos commands	108
3.5.1 Description of the Kerberos commands	108
3.5.2 Kerberos command examples	109
3.6 Auditing	116
3.7 Windows 2000 use of Kerberos	116
3.8 Windows 2000 interoperation with OS/390 NAPS	117
3.8.1 Install Windows 2000 DNS server	117
3.8.2 Set up peer trust between the Kerberos realms	118
3.8.3 Define the 390 KDC to each Windows 2000 workstation	119
3.9 DB2 Version 7 usage of Kerberos	119
3.9.1 DB2 Connect Version 7.1 setup	119
Part 3. LDAP and its use with other products	127
Chapter 4. Policy Director usage of the LDAP server on z/OS	129
4.1 Test environment	130
4.1.1 z/OS LDAP server setup	130
4.1.2 Policy Director Setup	137
Chapter 5. IBM Host On-Demand Version 6	143
5.1 LDAP native authentication	144
5.1.1 Native authentication requirements	145
5.1.2 Installation of native authentication	146
5.1.3 Starting and stopping native authentication services	146
5.1.4 Working with native authentication	148
5.1.5 Problem determination	149
5.2 Telnet-negotiated security	150
5.2.1 Session negotiation	151
5.3 Express logon feature	152

5.3.1 Overview	153
5.3.2 The RACF-secured sign-on PassTicket	157
5.3.3 Configuring the TN3270 server	158
5.3.4 Configuring the client	161
5.3.5 Installing the client certificate into the browser	164
Chapter 6. z/OS 1.2 LDAP/WebSphere Application Server	177
6.1 LDAP and WebSphere 4.0.1	178
6.2 WebSphere Application Server 4.0.1 overview	179
6.2.1 Installing WebSphere Application Server 4.0.1	180
6.2.2 WebSphere Application Server 4.0.1 and LDAP	181
6.2.3 Installation dialog	184
6.2.4 Problems encountered and lessons learned	187
6.3 Migrating WebSphere Application Server 4.0.1 from RDBM to TDBM	194
6.3.1 Migrating existing RDBM data to a TDBM database	194
6.3.2 Setting up the schema for TDBM	205
6.3.3 Restarting WebSphere Application Server after TDBM migration	211
6.3.4 Tips and recommendations for debugging/tracing	215
Chapter 7. LDAP server on z/OS	219
7.1 Input file description	220
7.1.1 Customizing ldap.profile	220
7.1.2 Customizing ldap.db2.profile	221
7.1.3 Customizing ldap.racf.profile	223
7.1.4 Customizing ldap.slapped.profile	223
7.2 Starting the ldapcnf utility	223
7.3 Executing system and RACF jobs	225
7.3.1 Update proclib	225
7.3.2 Update PARMLIB	225
7.3.3 Execute RACF jobs	225
7.4 Executing DB2 jobs	225
7.5 Start the LDAP server	226
7.6 Finalize setup of LDAP server	226
7.6.1 Load schemas	226
7.6.2 Load the suffix entry for the TDBM backend	227
7.6.3 Verify configuration	227
7.7 Implementing password encryption with z/OS LDAP	228
Chapter 8. LDAP with RACF digital certificates	233
8.1 Preparing to use RACDCERT commands	234
8.2 Server authentication	234
8.3 Client Authentication	239
Chapter 9. Overview of security on Linux	243
9.1 Disable unneeded services	244
9.2 Use Secure Shell for remote access	247
9.3 Use shadow password utilities	253
9.4 Use the Pluggable Authentication Module (PAM)	255
9.5 Monitor security news and alerts	259
9.6 Use hardening tools	260
9.7 Integrate VM and z/VM security	262
Appendix A. Notes about using SDBM with the LDAP Server	265

Related publications	271
IBM Redbooks	271
Other resources	271
Referenced Web sites	272
How to get IBM Redbooks	272
IBM Redbooks collections	272
Index	273

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.


This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Redbooks(logo)TM 
AFS®
AIX®
AnyNet®
C/VMTM
CICS®
DB2®
DB2 ConnectTM
IBM®
IBM.COMTM
IMSTM

MVSTM
OpenEdition®
OS/2®
OS/390®
PAL®
Parallel Sysplex®
PerformTM
RAA®
RACF®
RedbooksTM
SAA®

S/390®
SecureWay®
SPTM
SP1®
System/390®
Tivoli®
VTAM®
WebSphere®
z/OSTM
z/VMTM
zSeriesTM

The following terms are trademarks of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both:

Lotus®
Lotus Notes®

Notes®
DominoTM

Word Pro®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Preface

Many significant enhancements have been made to the security features available on the z/OS and OS/390 platforms. This IBM Redbook will help you install, customize, and configure the new functions provided with Version 1.2 of SecureWay Security Server for z/OS and OS/390 2.10. It is useful for system programmers, security administrators, and webmasters enabling e-business on these platforms.

LDAP enhancements on z/OS 1.2 that are discussed include extended operations client APIs, Kerberos authentication, the LDAP Configuration Utility (LDAPCNF), new SDBM features, TDBM native authentication. Techniques to increase server front-end performance and scalability are also described.

Security enhancements on OS/390 2.10 are achieved by new and improved RACF features, and by a new component of the Security Server for OS/390, called Network Authentication and Privacy Services (NAPS). NAPS is the IBM OS/390 implementation of Kerberos Version 5.

LDAP can provide directory services to a wide range of other applications. This redbook provides detailed information on integrating LDAP with other products to implement secure distributed computing environments and extend host data access across intranets and the Internet via simple to use and yet highly protected communications. It also gives step-by-step instructions for integrating LDAP with WebSphere Application Server, including sample code for migrating from RDBM to TDBM.

The steps necessary to configure and start an LDAP server with TDBM and SDBM are presented, along with instructions for implementing password encryption for the TDBM backend on a z/OS LDAP server. LDAP can be configured to provide RACF digital certificate support; this is discussed in detail. Finally, some basic security techniques for Linux on zSeries and S/390 are presented.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Montpellier Center.

Chris Rayns is a project leader at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on all areas of S/390 Security. Before joining the ITSO, Chris worked in IBM Global Services in the UK as an IT Specialist.

Angie Boone is an Advisory IT Networking Specialist on the TCP/IP customer support team at IBM Raleigh. She has more than ten years experience in the installation, implementation, and support of OS/390.

Karin Broecker has been with IBM for four years. She works with the zSeries New Workload team assisting customers as they begin to expand their e-business capabilities. Previously Karin worked at the Design Center for eTransaction Processing, helping customers to develop leading-edge cross-platform solutions.

John C Jones is a Certified I/T Specialist working for the S/390 New Technology Center in Poughkeepsie, NY. He has 22 years of experience in the MVS and OS/390 data processing fields. His areas of expertise include system programming, data management, and security. Jack currently works with customers to get their OS/390 systems ready for Internet access. Jack is a regular presenter at international conferences such as Guide, SHARE, the International Security Conference, and OS/390 Expo.

Philippe Richard works in the IBM EMEA Products and Solutions Support Center in Montpellier, where he is involved in benchmarks and education. Philippe graduated as a master in Enterprise management and Economy, and joined IBM France in 1985 to work in software support for MVS. Philippe has held a number of positions in this field, including teaching and systems programming, and he provided systems programming technical support for the 1992 Winter Olympics. Philippe has several years experience as a Technical Project Manager involved in implementing and consulting on Parallel Sysplex projects, and he has written many articles about this topic for various IBM publications.

Alain Striffling is an IT specialist in IBM France. He joined IBM in 1970 as an SE, supporting MVS customers in France. Since 1993, he has specialized in the area of host system security.

Thanks to the following people for their contributions to this project:

Patrick Kappeler
PSSC Montpellier

George Baker
Networking Technical Support, Raleigh

Travis Finucane
Tivoli Systems

Ira Chavis and the LDAP development organization (in particular, Karen Gdaniec)

Casey Cooley
Customer Support Specialist, Raleigh

Isabelle Grimalt
Montpellier

Jonathan Briggs, RACF Specialist
IGS, Strategic Outsourcing, IBM UK Leeds

Paul De Graaff
Formerly of ITSO, Security Project Leader

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

ibm.com/redbooks

- Send your comments in an Internet note to:

redbook@us.ibm.com

- Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Archived



Part 1

z/OS 1.2 enhancements

Archived

Archived

z/OS 1.2 LDAP Directory enhancements

In this chapter we describe the installation and implementation of the enhancements to the z/OS LDAP client and server, as well as miscellaneous other changes. The following enhancements have been made to the LDAP element of z/OS 1.2:

LDAP client enhancements

- ▶ DNS server locate
- ▶ Caching LDAP search results
- ▶ Extended operations
- ▶ Kerberos authentication

LDAP server enhancements

- ▶ LDAP configuration utility (LDAPCNF)
- ▶ Server front-end performance/scalability
- ▶ SDBM enhancements
 - Additional user segments
 - New search filters
 - User/groups connection management
- ▶ TDBM native authentication
- ▶ Kerberos authentication

Miscellaneous changes

- ▶ LDAP Server executables must now always be APF authorized.
- ▶ New schema (.ldif) files

1.1 LDAP client enhancements

The following sections provide brief descriptions of these enhancements.

DNS server locate

Client applications written in C/C++ can now omit the explicit hostname/portname information in the LDAP URL. Instead, this information can be obtained through the new API which can perform the following operations:

- ▶ Obtain LDAP server information published in the Domain Name Server (DNS).
- ▶ Obtain LDAP server information from a local configuration file.
- ▶ Store LDAP server information obtained from DNS into a local configuration file.

Caching LDAP search results

These enhancements allow subsequent `ldap_search` requests using the same search constraints (host, port, search base, authenticated bind dn, filter, etc.) to be satisfied from the cache. Global caching does not require modification or recompilation of existing LDAP applications.

- ▶ New client environment variables have been added to create a global client-side cache.
- ▶ New C/C++ APIs to create a client-side cache associated with a specific LDAP connection are available.

Extended operations client APIs

Extended operations are defined for special purposes and may perform several LDAP operations before returning a result.

- The operation to perform is identified by OID.
- Input/output data and format is defined by the documentation for the OID.

Note the following:

- ▶ z/OS LDAP server does not support any extended operations (LDAP_PROTOCOL_ERROR will be returned).
- ▶ z/OS LDAP client provides APIs for extended operations supported by *other* LDAP servers:
 - `ldap_extended_operation`
 - `ldap_extended_operation_s`
 - `ldap_parse_extended_result`

Kerberos authentication

Starting with z/OS 1.2, a client binding to an LDAP server can authenticate with a valid Kerberos ticket obtained from a Ticket Granting Server. LDAP client and LDAP server have been enhanced for Kerberos support, as follows:

- ▶ **LDAP C/C++ client**
 - The client can bind to any LDAP server supporting Kerberos V5 server, including the LDAP servers on z/OS, and AIX as well as Active Directory.
 - The Kerberos DLL must be in STEPLIB, LIBPATH, LPALIB or LINKLIB.
- ▶ **LDAP utility programs** (ldapsearch, etc.)

These have been updated to perform kV5 bind, as well as Kerberos authenticated referrals. Default credentials for the current login will be used:

```
ldapsearch -h... -p... -v3 -s gssapi -b...
```

1.2 LDAP server enhancements

The following sections provide brief descriptions of these enhancements.

LDAP configuration utility LDAPCNF

The LDAPCNF z/OS UNIX utility was designed to help customers perform z/OS LDAP server configuration. It is invoked under the OMVS shell, as shown in Figure 1-1:

```
cd /usr/lpp/ldap/sbin
ldapcnf -i /usr/local/aiains/ldap/ldap.profile
```

Figure 1-1 Invoking the LDAPCNF utility under the OMVS shell

Customization is done in four input HFS files:

- ldap.profile (the main ldap configuration file)
- ldap.slapped.profile (SSL configuration...)
- ldap.db2.profile (DB2 object names)
- ldap.racf.profile

LDAPCNF produces 10 members in an MVS PDS, the PDS name is specified in the OUTPUT_DATASET keyword of ldap.profile. Error messages may also be produced.

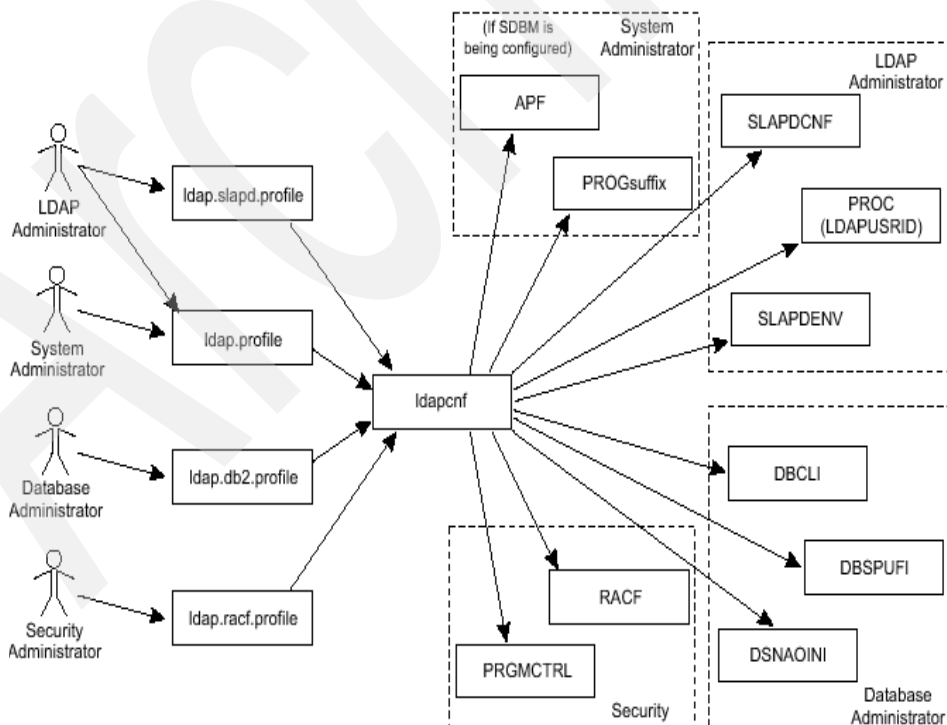


Figure 1-2 LDAP CNF utility

LDAPCNF always configures a TDBM and optionally an SDBM.

Advanced features supported are:

- Password encryption
- Referrals
- Replication
- SSL
- Kerberos authentication
- Native authentication

LDAPCNF restrictions are:

- LDAP runs as a started task.
- RACF is the security server.
- RDBM is not supported.
- Multiple TDBM backends are not supported.
- It cannot extend or enhance an existing LDAP server.
- It listens only on one SSL and one non-SSL port.

If you cannot use LDAPCNF because of one of these restrictions, you should manually update the slapdcnf file.

1.2.1 Server front-end performance/scalability

An LDAP server can now support up to approximately 65500 concurrent clients/connections, and can listen on:

- A specific IP address
- Multiple secure and non-secure ports for incoming requests (as opposed to one non-secure and one secure)

New configuration parameters are:

- *commThreads*: The number of threads initialized in the communication pool.
- *listen*: The IP address/port number to listen on for incoming client requests. This new parameter may be specified more than once. It deprecates former options: *security*, *port* and *securePort* options of the configuration file.

Example 1-1 The new listen configuration parameter

```
listen ldaps://:500
listen ldap://:400
listen ldap://us.endicott.ibm.com:777
listen ldaps://9.130.77.27:999
```

Some LDAP Server command line option are changed due to use of listen:

- h host, -p port and -s secureport are deprecated
- l replaces the above options

Example 1-2 The slapd command with the new -l parameter

```
slapd -l ldap://:389 -l ldaps://:636
```

- *idleConnectionTimeout*: The wait time in seconds on an idle client. The default is 0 (indefinitely).

1.2.2 SDBM enhancements

The following sections provide brief descriptions of these enhancements.

Additional user segments

LDAP RACF schemas are updated to manage new segments in the user profile, as shown in Figure 1-3.

LNOTES	objectclass	racfLNotesSegment	
	requires	objectClass	
	allows	racfLNotesShortName	
NDS	objectclass	racfNDSegment	
	requires	objectClass	
	allows	racfNDSUserName	
KERB	objectclass	racfKerberosInfo	
	requires	objectClass	
	allows	racfCurKeyVersion	
		krbPrincipalName	
		maxTicketAge	

Figure 1-3 New segments in LDAP RACF schemas

New search filters

The following filters are available for search operations on the RACF backend:

- racfLNotesShortName=value
- racfNDSUserName=value
- krbPrincipalName=value
- racfOmvsUid=value
- racfOmvsGroupId=value

For these new filters, only simple values are allowed. For LNOTES, NDS and KERB filters, the user must have read access to IRR.RUSERMAP in the FACILITY class.

Example 1-3 A new ldapsearch filter

```
ldapsearch . . . -b sysplex=potinfo "racfomvsuid=123456"
```

User/groups connection management

User connection to RACF groups can now be managed through LDAP commands. It is possible to display, modify, add, or remove user connections to groups. A new SDBM subtree for connect entries has been added (see Figure 1-4 on page 8).

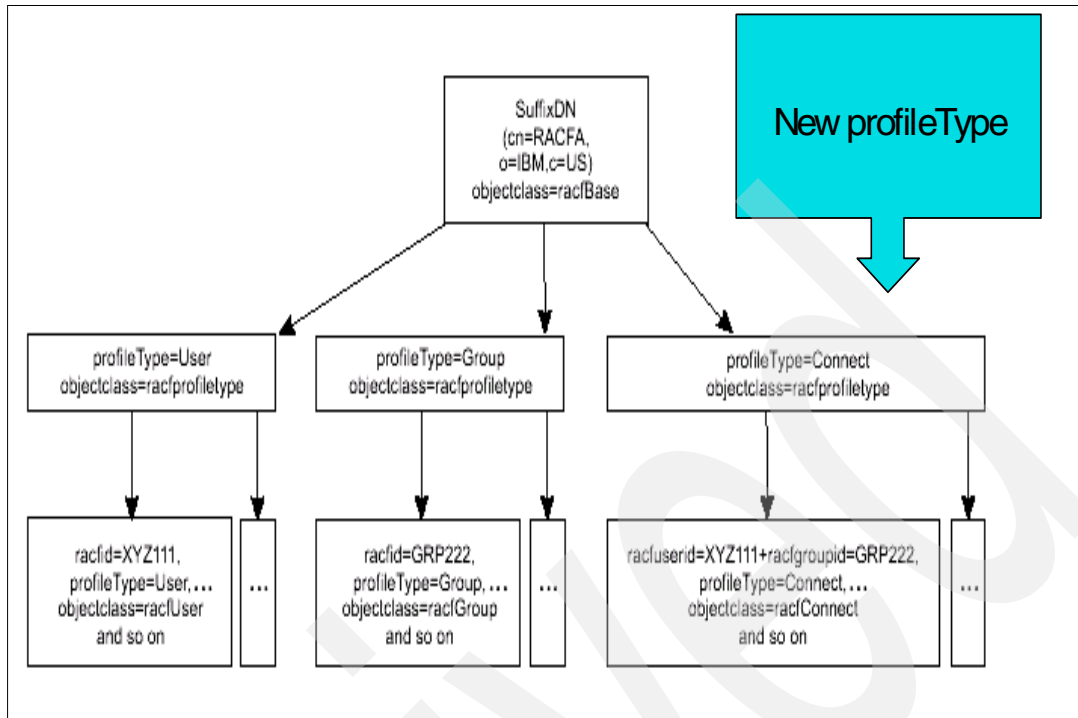


Figure 1-4 New SDBM Subtree with Connect ProfileType

The following figures show several examples of `ldapsearch` or `ldapadd` executed under ALAINS authority (SPECIAL Racf).

```
/bin/ldapadd -h 9.100.203.110 -p 3389 +
-D racfid=ALAINS,profiletype=user,sysplex=portinfo +
-w s3cret +
-f /usr/local/alains/ldap/connect.add
```

the HFS file connect.add contains:

```
dn:
racuserid=ALAINS1+racgroupid=DB2ADM,profiletype=connect,sysplex=portinfo
objectclass: racfconnect
racuserid: ALAINS1
racgroupid: DB2ADM
```

Figure 1-5 Connect a user to a group

```
/bin/ldapsearch -h 9.100.203.110 -p 3389 +
-D racfid=ALAINS,profiletype=user,sysplex=portinfo +
-w s3cret +
-b profiletype=CONNECT,sysplex=portinfo +
"(racfuserid=ALAINS1)"
```

result of this search is:

```
racfuserid=ALAINS1+racfgroupid=OMVSGRP,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS1+racfgroupid=SUPMVS,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS1+racfgroupid=SYS1,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS1+racfgroupid=USER,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS1+racfgroupid=DB2ADM,profiletype=CONNECT,sysplex=portinfo
```

Figure 1-6 Display connect groups for a user

A more complex search can also be performed, as shown in Figure 1-7.

```
/bin/ldapsearch -h 9.100.203.110 -p 3389 +
-D racfid=ALAINS,profiletype=user,sysplex=portinfo +
-w s3cret +
-b profiletype=CONNECT,sysplex=portinfo +
"(&(racfuserid=ALAI*)(racfgroupid=OMVSGRP))"
```

result of this search is:

```
racfuserid=ALAIN+racfgroupid=OMVSGRP,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS+racfgroupid=OMVSGRP,profiletype=CONNECT,sysplex=portinfo
racfuserid=ALAINS1+racfgroupid=OMVSGRP,profiletype=CONNECT,sysplex=portinfo
```

Figure 1-7 Display users beginning by ALAI connected to group OMVSGRP*

1.2.3 TDBM Native authentication

This new feature allows a client to bind to a TDBM backend with a password that is passed to and verified by RACF even if SDBM is not implemented together with TDBM.

When binding to LDAP, the end user needs to enter a distinguished name (Cn= ou= o= c=) and the RACF password associated to the RACF userid specified in the `ibm-nativeld` attribute. If the RACF userid is not defined, then a regular LDAP bind is attempted.

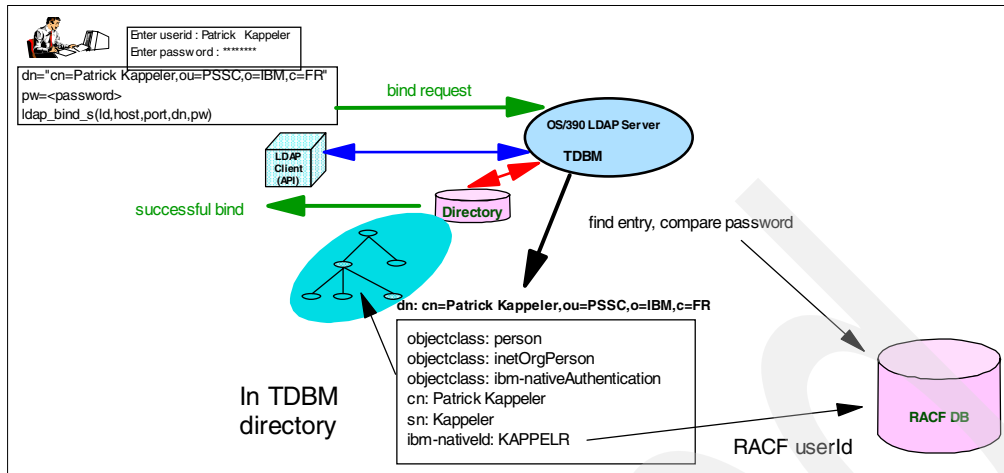


Figure 1-8 Bind to LDAP server with Native authentication

To allow Native authentication, you must:

1. Load the appropriate schema in LDAP, by `ldapmodify`:

```
/bin/ldapmodify -h -p -D -w -f /usr/local/alains/ NativeAuthentication.ldif
```

This is not necessary if `schema.IBM.ldif` is already loaded.

2. Some parameters have to be configured in the `ldap.slapd.profile`. The parameters and their meanings are as follows:

`TDBM_USENATIVEAUTH='selected'`

Only entries containing the attribute `ibm-nativeId` will participate in native authentication. (The choices are `selected`, `all`, and `off`.)

`TDBM_NATIVEUPDATEALLOWED='on'`

Specifies whether updating of the password in RACF is allowed. The choices are `on` and `off`.

`TDBM_NATIVEAUTHSUBTREE='o=IBMMOP'`

Specifies the subtree that contains entries that will use native authentication. "All" may also be specified.

1.2.4 Kerberos authentication

LDAP server is now able to accept binds and secure authentication using a valid Kerberos ticket. Kerberos is used only for LDAP authentication; there is no support for Kerberos integrity and confidentiality.

After successful authentication, LDAP directory access control is performed on the basis of the LDAP distinguished name. Mapping has to be done between the Kerberos "principal" and the LDAP distinguished name.

LDAP server Kerberos configuration

Use the following steps to configure Kerberos on the LDAP server.

1. Install and configure the z/OS Network Authentication and Privacy Service (NAPS) on the machine where the LDAP server will run, even though the install of NAPS is required, the KDC does not have to run on z/OS.
2. Create the LDAP server Kerberos KDC, or generate the server's keytab file.

3. Specify the Kerberos options in the `ldap.slapd.profile` (or in the `slapd.conf` file if you don't use `ldapcnf` utility).

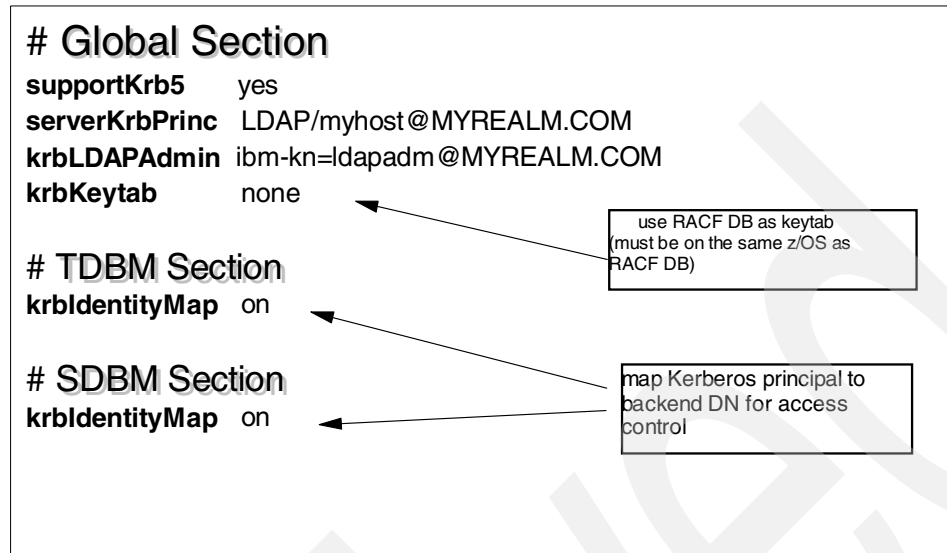


Figure 1-9 LDAP Kerberos configuration options

4. Update the current schema with `MS.ActiveDirectory.Idif` and `SecurityIdentities.Idif`, unless you have already updated your schema with `schema.user.Idif`, which includes both.

Mapping Kerberos principals to LDAP distinguished names

The bind principal, `principal@realm` can be used as the access-id in an ACL entry, or mapped to a distinguished name that already exists in an ACL entry. The default DN for a Kerberos identity not mapped to TDBM or SDBM DNs will be `ibm-kn=principal@realm`. The options are:

- SDBM style DN
- TDBM style DN

Mapping principals may yield a list of alternate DNs. Permission will be the union of all alternate DN's in the list. The following is an example of default mapping, assuming a principal name of `jeff@IBM.COM`.

- Default mapping
setting up a new ACL entry

```
dn: cn=Scott,O=IBM,c=US
aclEntry: access-id:ibm-kn=jeff@IBM.COM:normal=r
Jeff will have read access to normal data in Scott entry
```

Figure 1-10 Default mapping

- Principal name has been defined in a RACF user Kerberos segment:
`ALU JEFF KERB(KERBNAME(jeff@IBM.COM))`

```
dn: cn=Scott,o=IBM,c=US
aclentry:racfid=JEFF,pofiletype=user,sysplex=portinfo:normal:rw
    jeff will have update access to normal data in Scott entry
```

Figure 1-11 Principal defined in a RACF user segment

```
dn: cn=Jeff,o=IBM,c=US
objectClass:ibm-securityIdentities
altSecurityIdentities:KERBEROS:jeff@IBM.COM

dn: cn=Scott,o=IBM,c=US
aclEntry:cn=jeff,o=IBM,c=US:normal:rw
    jeff will have update access to normal data in Scott entry
```

Figure 1-12 Principal name associated with DN

More sophisticated methods with definition of aliases are also available. They permit you to give **dn: cn=jeff,o=IBM,c=US** the authorizations of another existing DN. By combining all these methods, you can build a list of DNs associated with a principal name.

1.3 Miscellaneous changes

The following new features and restrictions have been implemented.

- ▶ LDAP server executables must now be always APF-authorized.
- ▶ New .ldif files have been included. Many of new .ldif files for new functions (Kerberos, Native authentication, and so forth) are available in /usr/lpp/ldap/etc/ .

Most of them are included in:

- schema.user.ldif
- schema.IBM.ldif

- | | |
|------------------------------------|----------------------------|
| ▪ ChangeLog.ldif | ▪ OnDemandServer.ldif |
| ▪ CommServer.ldif | ▪ OtherStandard.ldif |
| ▪ ComponentBroker.ldif | ▪ PolicyDirector.ldif |
| ▪ DB2.ldif | ▪ RACF.ldif |
| ▪ DMTF.ldif | ▪ RACF.2.ldif |
| ▪ EntrustPKIV4.ldif | ▪ Registered Software.ldif |
| ▪ EntrustPKIV5.ldif | ▪ RFC2252.ldif |
| ▪ IBM.ldif | ▪ RFC2256.ldif |
| ▪ Kerberos-V1.ldif | ▪ RFC2587.ldif |
| ▪ ManagedSystemInfrastructure.ldif | ▪ RFC2713.ldif |
| ▪ MCI.ldif | ▪ RFC2714.ldif |
| ▪ MetaDirectory.ldif | ▪ SecurityIdentities.ldif |
| ▪ MS.ActiveDirectory.ldif | ▪ System.ldif |
| ▪ NativeAuthentication.ldif | ▪ System-V2.ldif |
| ▪ Netscape.ldif | ▪ UniversalMessaging.ldif |
| ▪ Netscape-V2.ldif | ▪ UNIX.ldif |
| ▪ NFI.ldif | ▪ WebSphereNaming.ldif |
| ▪ nisSchema.ldif | ▪ X.520.ldif |

Figure 1-13 Complete list of schema modules



Part 2

OS/390 2.10 enhancements

ARCHIVED

Archived

RACF enhancements

This chapter provides detailed information about the Release 10 enhancements to the RACF element of the SecureWay Security Server for OS/390.

The enhancements are presented as separate sections in this chapter:

- ▶ Section 2.1, “Program control enhancements” on page 16
- ▶ Section 2.2, “Application Identity Mapping (AIM)” on page 20
- ▶ Section 2.3, “Digital certificate enhancements” on page 33
- ▶ Section 2.4, “PKIServ - Web interface to certificate generation” on page 48
- ▶ Section 2.5, “OS/390 UNIX Superuser granularity support” on page 79
- ▶ Section 2.6, “Service Updates” on page 79
- ▶ Section 2.7, “RVARY command enhancement” on page 80

2.1 Program control enhancements

This section describes the enhancements made to Program Access to Data Sets (PADS). UNIX System Services need RACF program control support to provide good security and integrity for UNIX servers and daemons. RACF will also produce new diagnostic messages.

2.1.1 Overview

One of the continuing objectives of OS/390 is to encourage server consolidation, enabling more applications to run on OS/390. As this occurs, however, and especially as customers and vendors create new server applications (or port existing ones) they may experience difficulty configuring program control using the RACF PROGRAM class for traditional MVS libraries and the program-controlled extended attribute for OS/390 UNIX files. Program control is required if servers are to run properly with good security and integrity.

2.1.2 Introduction

The support provided by Program Control Enhancements will make it easier for a customer to determine which programs, either in traditional MVS load libraries or in the OS/390 UNIX hierarchical file system (HFS), they need to define as controlled programs. Plus it will help preserve the security and integrity afforded to servers and daemons while they run; prior to Version 2 Release 10 this was used at startup time only, preventing the introduction of uncontrolled programs into the execution of the server or daemon. This can prevent compromise of security or integrity of the server or daemon.

By protecting the BPX.DAEMON and BPX.SERVER resources in the FACILITY class, with fewer difficulties in defining programs to RACF and the HFS, customers will also get good security and integrity for their servers and daemon running under OS/390 UNIX. RACF and/or OS/390 UNIX will provide messages to aid in the definition of the appropriate programs.

This new support helps with the definition of programs for use with RACF PADS and to execute in a controlled processing environment. This is achieved through the improved messages that result when execution time errors occur, and through the ability of the RACROUTE REQUEST=AUTH post processing exit (ICHRXC02) to determine that a PADS request failed because the environment was not controlled.

2.1.3 Description

RACF will provide a new service, IRRENS00. A bit in the RACF Communication Vector Table (RCVT) indicates that the service is available. The address of the service is located by the RCVT, as it is for several other RACF services.

The new service will provide the following functions:

1. Mark the environment uncontrolled (dirty).

The mark-uncontrolled function first checks that the environment is allowed to become uncontrolled (dirty). It checks the current state of the environment to see if it is controlled or uncontrolled. If it is controlled (clean), it can become uncontrolled if keep-controlled is not in effect. If the environment was controlled when the service was called, it is marked uncontrolled, and a message provided by the caller is saved describing the reason it became uncontrolled.

If the environment is controlled, and cannot become uncontrolled (keep-controlled is in effect), then IRRENS00 displays any previously saved message via WTO to help the administrator correct the problem. The message was saved when a caller requested that the environment be kept controlled, and provided a message on that request.

2. Keep environment controlled (clean).

The keep-controlled function marks the current environment as keep-controlled (keep-clean) if it is currently controlled, and saves a message provided by the caller describing the reason the environment must remain controlled. If the environment is already uncontrolled (dirty), the request fails and IRRENS00 will issue any previously saved message via WTO to help the administrator correct the problem. This message was saved when a caller requested that the environment be marked uncontrolled. If there is no saved message, a general message is issued.

3. Reset keep-controlled.

This function turns off the specified keep-controlled indicators and clears the related saved message when an authorized caller knows it is safe to do so. In general, OS/390 UNIX turns off the OS/390 UNIX keep-controlled indicator, and RACF turns off the RACF keep-controlled indicator. Support is provided for resetting both.

In addition to the messages saved by IRRENS00, RACF will provide messages to aid diagnosis in some cases, specifically when it denies:

- Use of PADS for a data set due to a dirty environment
- Use of PADS due to the absence of the currently executing program in the conditional access list, or due to presence of other programs defined with PADCHK in the environment that are not found in the conditional access list of the data set profile
- Authority to load an uncontrolled module due to the presence of execute-controlled modules in the environment
- Authority to load any module to the existence of open PADS data sets in the environment

These messages should make it easier for the security administrator to determine what PROGRAM definitions or WHEN(PROGRAM(...)) conditional access list entries need modification.

2.1.4 RACF services

RACF provides a new service IRRENS00 (Environment Service) located via the RCVT. IRRENS00 provides the following functions:

► Keep-Controlled

This function verifies that the environment is currently controlled, and if so, sets flags indicating that it must remain controlled. It also saves a message supplied by the caller indicating the reason it must remain controlled. It keeps separate flags for UNIX System Services requests and RACF requests.

► Mark-Uncontrolled

This function determines whether the environment must remain controlled by inspecting the keep-controlled indicators and, if necessary, the PADS data set list from the Accessor Environment Element (ACEE) and the CDEs for modules in the address space, and if not marks it uncontrolled and also marks any existing TCBs and CDEs as uncontrolled. However, if a keep-controlled request is outstanding, mark-uncontrolled returns an error code and either issues a WTO saved message from a previous keep-controlled request or generates messages to indicate what RACF needed the environment kept controlled. For a successful request, mark-uncontrolled will also save a caller-provided message indicating why the environment became uncontrolled, which it can issue on subsequent keep-controlled requests if necessary.

► Reset Keep-Controlled

This function resets the keep-controlled indicators and removes any saved messages relating to previous keep-controlled requests. It is intended only for use by OS/390 UNIX System Services when they have determined that the environment no longer needs to be kept controlled.

OS/390 UNIX System Services should use IRRENS00 to tell RACF to keep the environment clean, to mark the environment dirty, and to reset the keep-controlled indicator.

If you have set up conditional access lists specifying WHEN(PROGRAM(...)) for DATASET profiles, additional error messages will be sent to the syslog for failed attempts to access data sets. If these conditional access lists are not being used to allow access, they should be deleted.

2.1.5 Program Control enhancements example errors

We deliberately removed a data set from the RACF PROGRAM class for documentation purposes, so we could demonstrate and show what error messages you would receive in your own installation.

We decided to remove the data set CEE.SCEERUN, and stopped and re-started our LDAP and HTTP servers.

By using the following **RLIST** command we can find and display the profile within the RACF PROGRAM class that contains the CEE.SCEERUN data set profile:

```
RLIST PROGRAM *
```

From our **RLIST** command we found the data set CEE.SCEERUN was in the RACF PROGRAM profile *, as shown in Example 2-1.

Example 2-1 RACF RLIST command output

CLASS	NAME
-----	----
PROGRAM	*
MEMBER	CLASS NAME
-----	-----
PMBR	

DATA SET NAME	VOLSER	PADS	CHECKING
-----	-----	-----	-----
CEE.SCEERUN			NO
CSF.SCSFMODE			NO
CSF.SCSFMODE1			NO
DSN510.SDSNLOAD			NO
DSN610.SDSNLOAD			YES

We then remove the data set from the * profile in the RACF class PROGRAM, for test purposes, by issuing the following command:

```
RALTER PROGRAM * DELMEM('CEE.SCEERUN')
```

Before we can test for any errors by stopping and starting LDAP and HTTP servers, we must issue the following RACF **SETROPTS** command:

```
SETR WHEN(PROGRAM) REFRESH
```


Now we can stop and restart the LDAP and HTTP servers. From the startup of the LDAP server the message shown in Figure 2-1 is displayed, informing us that the program EDC\$UEY in data set CEE.SCEERUN caused the environment to become uncontrolled (dirty) even though the LDAP server was up and running.

```

ICH420I PROGRAM EDC$UEY FROM LIBRARY CEE.SCEERUN CAUSED THE
ENVIRONMENT TO BECOME UNCONTROLLED.
BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR SERVER (BPX.SERVER)
PROCESSING.
ICH420I PROGRAM EDC$UEY FROM LIBRARY CEE.SCEERUN CAUSED THE
ENVIRONMENT TO BECOME UNCONTROLLED.
BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR SERVER (BPX.SERVER)
PROCESSING.
ICH420I PROGRAM EDC$UEY FROM LIBRARY CEE.SCEERUN CAUSED THE
ENVIRONMENT TO BECOME UNCONTROLLED.
BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR SERVER (BPX.SERVER)
PROCESSING.

```

Figure 2-1 RACF error message from restart of the LDAP server

With the HTTP server, we had no error messages displayed at startup or in the log, so at this point we do not know whether the environment is in a controlled (clean) or uncontrolled (dirty) state. Upon a further test (trying to use HTTP to obtain a communication connection with a digital certificate through the browser) a message is displayed, as shown in Figure 2-2. We can also check the log on OS/390 and we will find the same error messages, as shown in Figure 2-1.

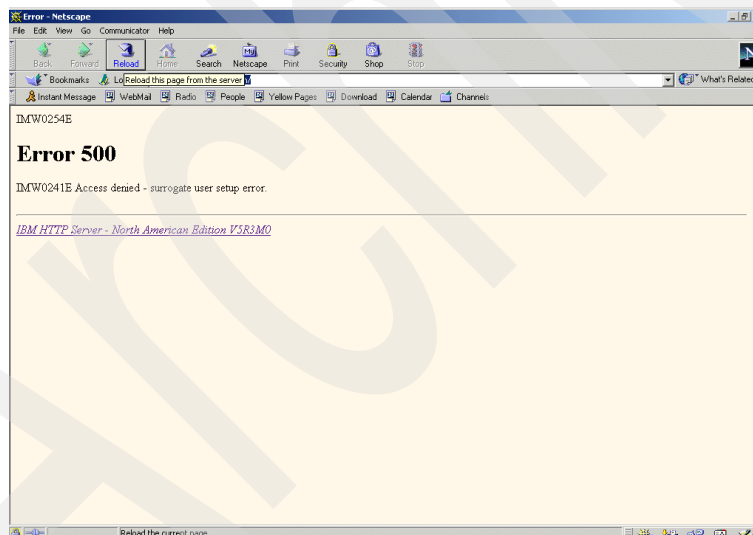


Figure 2-2 Error message displayed from requesting digital certificate

Note: It is possible that after making changes or defining new applications, the application will start up. However, the application could be in an uncontrolled (dirty) state, and until the application is used in a way that a password verification or some authorized/trusted/restrictive command is used, no error messages will be displayed

After these tests we added back the CEE.SCEERUN data set to the * profile in the class PROGRAM using the RACF **RALTER** command:

```
RALTER PROGRAM * ADDMEM('CEE.SCEERUN'//NOPADCHK)
```

We now need to refresh so other applications can use the CEE.SCEERUN data set without getting RACF violations or unauthorized program checks. We do this with the following RACF **SETROPTS** command:

```
SETROPTS WHEN(PROGRAM) REFRESH
```

2.2 Application Identity Mapping (AIM)

This section details the new Application Identity Mapping (AIM) function, and how to migrate RACF User IDs with OMVS, NDS, and LNOTES segments, along with GID, UID, UNAME, and SNAME fields. A new utility, **IRRIRA00** migrates the UNIXMAP, NOTELINK, and NDSLINK mapping profiles to alias entries, which require less space on the RACF database. Updates to the **ADDUSER** and **ALTUSER** commands prevent you from associating application user identities for Lotus Notes for OS390 and Novell Directory Services for OS390 with more than one RACF user ID.

2.2.1 Overview

As the SecureWay Security Server for OS/390 (RACF) supports more and more interoperability with products ported to OS/390, there exists a growing need to manage and associate the user (client) identities created by those products with traditional OS/390 (RACF) User IDs and in some cases with group IDs.

Already we have cases where this has occurred, with OS/390 support for UNIX, Lotus Domino (Notes), and Novell Directory Services (NDS). In these cases to date, RACF has maintained the identity associations using additional profiles called *mapping* or *linking* profiles in the RACF database using the classes UNIXMAP, NOTELINK and NDSLINK. However, the use of mapping profiles has the following disadvantages:

- ▶ Each additional profile requires an entry in an index block in the RACF database and at least 256 bytes of space in a data block. This increases the amount of space used in the RACF database.
- ▶ When RACF needs to use a mapping profile to determine the RACF identity that matches another identity, RACF might need to perform I/O to the RACF database to read both the relevant index block and the data block containing the mapping profile.
- ▶ RACF operations to maintain the additional mapping profiles occur from internal exits in the **ADDUSER**, **ALTUSER**, **DELUSER**, **ADGROUP**, **ALTGROUP** and **DELGROUP** commands. Because these operations occur from command processors, rather than within the RACF database manager, they have the following problems:
 - They do not occur if any installation or vendor-written code manipulates user or group profiles or the relevant product-specific segments directly via **ICHEINTY** or **RACROUTE**, thus leading to potential mismatches between the user/group profiles and the mapping profile.
 - They lack referential integrity because they do not occur using the same serialization that occurs for maintaining the user and group profiles. This can also lead to mismatches between the user/group profiles and the mapping profiles, as two updates to a user or group profile could “collide” and leave an inconsistent mapping profile.

AIM provides an alternative technique to mapping profiles which extends the RACF database manager processing to recognize the fields that relate RACF user and group identities to those of other products, for example, the UID and GID fields in the OMVS segment, and to automatically maintain an alternate index structure containing alias index entries that can locate a user or group profile given the other product's identity information.

Without actually reading a profile, as with the existing mapping profile support, an alias index entry may be associated with more than one user ID or group name in the case of the OMVS segment. An alias locate request will return the first user ID or group name found in the alias index entry.

2.2.2 RACF utilities

A new conversion utility, **IRRIRA00** (the Internal Reorganization of Aliases utility) can process an existing RACF database. **IRRIRA00** activates AIM in four stages controlled by the installation and lists the current stage of its input database. The utility updates all active RACF data sets, including active backup data sets; all data sets making up a RACF database *must* be at the same stage.

IRRIRA00 opens the master primary RACF data set and, if active, the master backup RACF data set, to write the stage indicator into the ICB. Update authority to each data set is required to allow the data set to be opened. **IRRIRA00** obtains serialization to prevent activities such as **RVARY** and **SETROPTS** commands from being processed while the utility is running.

The creation of alias index entries and deletion of mapping profiles done by **IRRIRA00** will *not* propagate to other databases with RRSF. The utility must be run against each database when that system is ready to enter a new stage. RACF databases do not need to be at the same stage to be part of the same RRSF network unless installation or vendor-written code is used to manipulate mapping class profiles using **RACROUTE** or **ICHEINTY**. Command propagation will work correctly between systems whose RACF databases are at different stages.

The **IRRUT200** utility is modified to process the alias index structures, to detect errors, and to display the formatted alias index blocks. **IRRUT200** will use the existing INDEX, INDEX FORMAT and MAP options to control the analysis and formatting of the alias index structure. The formatted printout of alias index blocks will contain the number of base profiles associated with the entry and the base profile names rather than the RBA. When more than one base profile name is associated with an entry, each base profile name will appear on a separate line under the BASE PROFILES column, with other columns left blank.

IRRUT400 is modified to build an alias index entry structure in the output RACF database if it is present in the input RACF database. **IRRUT400** will build this structure from the sequence set in a way similar to that currently used to rebuild the regular index structure. Index compression and free space allocation, for example, will be done as it is for other index blocks.

2.2.3 AIM's purpose

The intention of AIM is to provide lookup by alias fields of base profile names. A base profile is a user or group profile that contains a set alias field. Table 2-1 shows the alias fields.

Table 2-1 Alias fields

CLASS	SEGMENT	FIELD
GROUP	OMVS	GID
USER	OMVS	UID
USER	NDS	UNAME
USER	LNOTES	SNAME

When AIM is at enablement stage 2 or 3, the lookup performed by certain callable services is via a new Icheinty Parameter List (IPL) operation, an Alias Locate. AIM enablement stages allow for:

- ▶ Stage 0 - Inactivation, prior support applies
- ▶ Stage 1 - Temporary stage allows new conversion utility to run, prior support applies
- ▶ Stage 2 - Use AIM, but if not found, seamlessly use prior support
- ▶ Stage 3 - Mapping profiles for prior support removed, only AIM is used

Before AIM was introduced there was “generic identity mapping,” which used mapping profiles to look up aliases:

- ▶ UNIXMAP for UID/GID
If UNIXMAP is inactive, a sequential search through the database is performed.
- ▶ NOTELINK for SNAME
- ▶ NDSLINK for UNAME

Classes must be active for lookup by callable services to occur.

Mapping profiles are maintained by the command envelope. Generic identity mapping had two shortcomings:

1. Space overhead: Most alias fields have a one-to-one mapping with a base profile.
Mapping profiles:
 - The callable services return from them a 1 - 8 character User/GroupID
 - Like all profiles, they have a certain amount of overhead, which can chew up space if you have, for example, 250,000 OS/390 UNIX users.
2. Lack of referential integrity.
The USR/GRP manipulations and mapping profile updates are not atomic operations (separate IPLs with no serialization between them).

2.2.4 Benefits of AIM

There are various benefits of AIM, including:

- ▶ It reduces the amount of RACF database I/O needed to update and retrieve mapping information on the database.
- ▶ It does away with mapping profiles: it uses alias index. To get to any profile, you must go through index blocks. Alias index sequence set (level 1 blocks) entries contain the base profile name instead of the data block pointers found in regular index blocks. This certainly saves space, since there is no profile.
- ▶ It provides referential integrity because the RACF manager updates the alias index under a new serialization paradigm when USR/GRP alias fields are manipulated, or looked at with the new Alias Locate IPL.
- ▶ In prior releases, NDS UNAME and LNOTES SNAME alias fields have a one-to-one mapping enforced by the command envelope. Generic identity mapping would only find out that a mapping profile existed after the base profile was updated. There would be error messages, but the only action taken would be to not update the mapping profile, leaving the two profiles out of sync. Starting with OS/390 Version 2, Release 10, both AIM and generic identity mapping can determine pre-existence (of either mapping profile or alias index, depending on enablement stage) before updating the base profile. If detected, the

base profile is NOT updated, leaving the base profile and mapping profile and/or alias index in sync.

- ▶ It can save DASD space once in stage 3 and the old mapping profiles are deleted, and with the staged approach we can accept the new support in a controlled manner.
- ▶ It provides better data integrity and more reliable lookups for the mapping information.
- ▶ Less administrative time will be required because the need to detect and correct out-of-sync conditions between user, group, and mapping profiles will be eliminated.

2.2.5 Migrating and installation of AIM

The prerequisite for AIM migration is to be at OS/390 Version 2 Release 10. A new utility that has been added at this level, **IRRIRA00** converts the current RACF database to use AIM, allowing you to advance through enablement stages. **IRRIRA00** converts a RACF database created before OS/390 release 10 to a database that supports the new alias identity mapping function. It runs only against an active primary and backup database. It serializes against RACF resources to prevent disruptive competing updates. It may run while the database is shared between systems. The sysplex Coupling Facility (CF) is updated as needed.

Table 2-2 Stage Transitions

Stage Transition	Utility Processing
0 -> 1	Create alias index entries corresponding to supported mapping profiles and user/group identities. Advance alias index stage to 1
1 -> 2	Advance alias index stage to 2
2 -> 3	Delete mapping profiles in the supported classes. Advance alias index stage to 3

2.2.6 Stage enablement

Prior to OS/390 Version 2 Release 10, a mapping profile is used if the appropriate class (UNIXMAP) is active. If UNIXMAP is not active, a database search through all USER/GROUP profiles with an OMVS segment is performed until a match is found for the UID/GID.

Virtual Lookaside Facility (VLF) before OS/390 Version 2 Release 10 is only applicable for OMVS UID/GID, and then only if IRRUMAP or IRRGMAP classes are active.

The stage transitions are:

- | | |
|---------|--|
| Stage 0 | Behavior of all runtime components (manager, commands, and callable services) has AIM completely disabled. |
| Stage 1 | The manager will maintain Alias index, disallow Alias Locates and purge VLF if necessary. The commands still maintain VLF and the old mapping profiles. The callable services will still use and update VLF. If the VLF lookup is either not tried, or unsuccessful, then the lookup is through either the mapping profile or, for OMVS UID/GID, an IPLNEXT search of the database, until a user/group with the particular OMVS alias field is found. Stage 1 was contrived to allow IRRIRA00 , via manager calls, to build the alias index but to deny its use by Alias Locates from within the callable services. |
| Stage 2 | The manager will maintain Alias index, allow Alias Locates and purge VLF if necessary. The commands still maintain VLF and the old mapping profiles. The |

callable services will use an Alias Locate. If that fails, then the callable service will continue with stage 1 behavior. LOGRECs will be cut when:

- ▶ Alias Locate has “not found” RC, but the base profile was found via stage 1 behavior.
- ▶ Alias Locate has an RC other than 0 or “not found.”

Stage 2 is the AIM assurance and compatibility mode.

Stage 3

This stage should not be entered until after stage 2 has completed without getting any LOGRECs indicating AIM failure. Additionally, all systems sharing the database must be at a code level supporting AIM. The manager will maintain Alias index, allow Alias Locates, VLF for IRRUMAP and IRRGMAP no longer purged. The commands no longer maintain VLF or the old mapping profiles. The callable services will *only* use an Alias Locate. The callable services will cut a LOGREC if the Alias Locate has an RC other than 0 or “not found.”

Table 2-3 Stage enablement table

STAGE	MANAGER	COMMANDS	CALLABLE SERVICES
0	Will not maintain. Will purge VLF. Will disallow alias locates.	Will maintain VLF and mapping profiles.	Will look for info via: VLF Mapping Profile or Database Search
1	Will maintain alias index. Will purge VLF. Will disallow alias locates.	Will maintain VLF and mapping profiles.	Will look for info via: VLF Mapping Profile or Database Search
2	Will maintain indices. Will purge VLF. Will allow alias locates.	Will maintain VLF and mapping profiles.	Will look for info via: Alias Locate VLF Mapping Profile or Database Search
3	Will maintain alias index. Will allow alias locates.	Will not maintain VLF nor mapping profiles.	Will look for info via Alias Locate

2.2.7 Installation

New databases that have been created with **IRRMIN00 PARM=NEW**, are initialized at stage 3. For existing databases **IRRMIN00 PARM=UPDATE** needs to be coded. AIM is invoked by RACF commands (**AU/ALU/DU/AG/ALG/DG**) affecting designated alias fields and by **ICHEINTY** macro service affecting designated alias fields. The migration to AIM is done by submitting the utility **IRRIRA00** to go through each stage.

We recommend before running **IRRIRA00** that you run the following utilities:

- ▶ **RACFICE** - To obtain information about the number of matching UID/GIDs.
- ▶ **IRRUT400** - For backout purposes.
- ▶ **IRRUT200** - To obtain statistics on UID/GIDs and the number of UNIXMAP, NOTELINK and NDSLINK for reference purposes before and after the migration to an AIM RACF database.

We also recommend that when you plan for the migration using our installation procedures you do the changes at a quiet time or during a scheduled outage.

Attention: IRRIRA00 will allow for 130 *eight* character User IDs sharing the same UID; the 131st would fail. Most installations have *seven* character TSO User IDs, so more than 130 would probably fit in. If your installation has more than one user ID with the UID segment of 0 or any other matching UID segments, we strongly recommend running the new RACF utility RACFICE. For more information on RACFICE refer to *OS/390 Security Server 1999 Updates Technical Presentation Guide*, SG24-5627.

The use of **RACFICE** is described in the next steps:

1. Unload the RACF database using the RACF database unload utility (**IRRDBU00**) for input to the RACFICE utility. Figure 2-3 shows sample JCL for the **IRRDBU00** utility.

```
//IRRDBU00 JOB (999,POK),'COPY RACF',NOTIFY=&SYSUID,
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//IRRDBU00 EXEC PGM=IRRDBU00,PARM='NOLOCK'
//SYSPRINT DD SYSOUT=*
//INDD1 DD DISP=OLD,DSN=SYS1.RACFESA
//OUTDD DD DSN=BRIGGS.RACF.IRRDBU00,DISP=(,CATLG),
//          VOL=SER=PDGTS1,
//          SPACE=(CYL,(25,5)),
//          DCB=(LRECL=4096,RECFM=VB),
//          UNIT=3390
```

Figure 2-3 Sample IRRDBU00 JCL

2. Run the **RACFICE** utility job using the sample shown in Figure 2-4 as input. The sample sort statements will look for any UIDs that have been used more than once.

```
*****
* Name: UIDS *
* *
* Find all the OS390 UNIX UIDs which are used more than once. *
*****
COPY FROM(DBUDATA) TO(TEMP0001) USING(RACF)
OCCURS FROM(TEMP0001) LIST(PRINT) -
PAGE -
TITLE('UIDS: OS/390 UNIX UIDs Used More Than Once') -
DATE(YMD/) -
TIME(12:) -
BLANK -
ON(19,10,CH) HEADER('OS/390 UNIX UID') -
ON(VALCNT) HEADER('Number of Times Used') -
HIGHER(1)
```

Figure 2-4 Sample sort statements

Edit the **RACFICE** job to use your sample sort member UIDs as the parameter input. Amend the **IRRDBU00** data set with the one you have just created. Figure 2-5 shows sample JCL for the **RACFICE** job.

```
//RACFICE JOB (999,P0K),'RACFICE',NOTIFY=&SYSUID,
//      CLASS=A,MSGCLASS=T,TIME=1439,
//      REGION=5000K,MSGLEVEL=(1,1)
//      JCLLIB ORDER=GRAAFF.RACFICE.CNTL
//      SET ADUDATA=GRAAFF.SMF.UNLOAD.D042201      IRRADU00 DATA
//      SET DBUDATA=BRIGGS.RACF.IRRDBU00          IRRDBU00 DATA
//      SET ICECNTL=GRAAFF.RACFICE.CNTL          ICETOOL DATA
//X500      EXEC RACFICE,REPORT=UIDS
```

Figure 2-5 Sample JCL for **RACFICE** job

Figure 2-6 displays the output from the **RACFICE** job.

```
OS/390 UNIX UIDs Used More Than Once

OpenEdition UID      Number of Times Used
-----
0000000000          39
0000099999           3
```

Figure 2-6 Output from **RACFICE** job using **UIDs** parameter

Note: If your installation has more than 130 matching UIDs, see 2.2.9, “When there are more than 130 user IDs with the same UID” on page 31.

3. Run the **IRRUT400** utility to backup the Primary RACF database. Figure 2-7 shows sample JCL for **IRRUT400**.

```
//IRRUT400 JOB (999,P0K),'COPY RACF',NOTIFY=&SYSUID,
//      CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//STEP1      EXEC PGM=IRRUT400,PARM='NOLOCKINPUT,FREESPACE(20)'
//SYSPRINT   DD SYSOUT=A
//INDD1      DD DSN=SYS1.RACFESA,DISP=OLD
//OUTDD1     DD DSN=SYS1.RACF.BRIGGS,DISP=(,CATLG),
//           VOL=SER=PDGSY1,
//           SPACE=(CYL,(43,0,0),,CONTIG),
//           DCB=DSORG=PSU,
//           UNIT=3390
```

Figure 2-7 Sample **IRRUT400** JCL

4. After the backup has run successfully, it is essential to run the **IRRUT200** utility to obtain the statistics on the **UNIXMAP** class, and the **LNOTES** and **NDS** segments. Figure 2-8 on page 27 shows sample JCL for the **IRRUT200** utility.


```

//IRRUT200 JOB (999,POK),'COPY RACF',NOTIFY=&SYSUID,
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//STEP      EXEC   PGM=IRRUT200
//SYSRACF   DD     DSN=SYS1.RACFESA,DISP=SHR
//SYSUT1    DD     UNIT=SYSDA,SPACE=(CYL,(70)),
//          DCB=(LRECL=4096,RECFM=F)
//SYSUT2    DD     SYSOUT=A
//SYSPRINT  DD     SYSOUT=A
//SYSIN     DD     *
            INDEX FORMAT
            MAP ALL
            END
/*

```

Figure 2-8 Sample IRRUT200 JCL

Check the output from the **IRRUT200** job, in particular the statistics at the bottom of the output as shown in Figure 2-9.

```

NUMBER OF NOTELINK ENTRIES - 0000009
NUMBER OF NDSLINK ENTRIES - 0000007
NUMBER OF UNIXMAP ENTRIES - 0000083

```

Figure 2-9 Extracts from IRRUT200 output

5. We recommend that before you start the **IRRIRA00** utility, you inactivate your RACF backup database.
6. Check to see what RACF databases are online, using the **RVARY LIST** command, as follows:

```
RVARY LIST
```

Figure 2-10 displays the output from the **RVARY LIST** command.

ACTIVE	USE	NUMBER	VOLUME	DATASET
YES	PRIM	1	PDGCAT	SYS1.RACFESA
YES	BACK	1	PDGSY1	SYS1.RACF.BKUP1

ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.

Figure 2-10 RVARY LIST command output

7. Inactivate the RACF backup database by using the **RVARY INACTIVE** command as follows:

```
RVARY INACTIVE,DATASET(SYS1.RACF.BKUP1)
```

After entering this command you must respond to the WTO message through an operator console with the correct **STATUS** password.

Figure 2-11 on page 28 displays the status of the RACF databases after the inactivation of the RACF Backup database.

```

ICH15013I RACF DATABASE STATUS:
ACTIVE  USE    NUMBER  VOLUME    DATASET
-----  ---    -
YES     PRIM     1     PDGCAT     SYS1.RACFESA
NO      BACK     1     *DEALLOC  SYS1.RACF.BKUP1
ICH15020I RVARV COMMAND HAS FINISHED PROCESSING.

```

Figure 2-11 Output generated by the RVARV INACTIVE command

8. You can now run the first stage of the **IRRIRA00** utility, to go from stage 0 to stage 1, as shown in Figure 2-12.

```

//IRRIRA00 JOB (999),'COPY RACF',NOTIFY=&SYSUID,
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//IRA00     EXEC  PGM=IRRIRA00,PARM='STAGE(1)'
//SYSPRINT DD   SYSOUT=A

```

Figure 2-12 Sample IRRIRA00 job: Moving to stage 1

Figure 2-13 shows the output from the stage migration. You get the warning message IRR66007I stating that the RACF backup database is not active since you inactivated the RACF backup database; this message is acceptable.

```

IRR66017I The system is currently operating in stage 0.
IRR66018I Stage 1 requested. Database now at requested stage 1.
IRR66007I Backup RACF database not converted to stage 1. It is not active.

```

Figure 2-13 Output from IRRIRA00 moving from stage 0 to stage 1

You now move to the next step of the stage enablement by executing the utility **IRRIRA00** again. The only modification required is to change the STAGE parameter from 1 to 2, as shown in Figure 2-14.

```

//IRRIRA00 JOB (999),'COPY RACF',NOTIFY=&SYSUID,
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//IRA00     EXEC  PGM=IRRIRA00,PARM='STAGE(2)'
//SYSPRINT DD   SYSOUT=A

```

Figure 2-14 Sample IRRIRA00 job: Moving to stage 2

Figure 2-15 shows the output from the second **IRRIRA00** job.

```

IRR66017I The system is currently operating in stage 1.
IRR66018I Stage 2 requested. Database now at requested stage 2.
IRR66007I Backup RACF database not converted to stage 2. It is not active.

```

Figure 2-15 Output from IRRIRA00 moving from stage 1 to stage 2

To complete the migration, we run the utility **IRRIRA00** to move from stage 2 to stage 3. Again, we must emphasize that you do not go to stage 3 until you have run in stage 2, without getting any LOGRECs indicating an AIM failure. Additionally, all systems sharing the database must be at the same code level supporting AIM.

Note: It is not possible to go back after you have reached stage 3 unless you back out using the backup utility of the RACF primary database taken in step 3.

Sample JCL for executing the utility **IRRIRA00** to change from stage 2 to stage 3 is shown in Figure 2-16.

```
//IRRIRA00 JOB (999),'COPY RACF',NOTIFY=&SYSUID,
//          CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1)
//IRA00     EXEC  PGM=IRRIRA00,PARM='STAGE(3)'
//SYSPRINT DD   SYSOUT=A
```

Figure 2-16 Sample IRRIRA00 job: Moving to stage 3

Figure 2-17 shows the output of the stage 3 enablement through the **IRRIRA00** utility.

```
IRR66017I The system is currently operating in stage 2.
IRR66018I Stage 3 requested. Database now at requested stage 3.
IRR66007I Backup RACF database not converted to stage 3. It is not active.
```

Figure 2-17 Output from IRRIRA00 moving from stage 2 to stage 3

9. After the successful migration to stage 3 enablement, the class **UNIXMAP** has to be inactivated from RACF, as shown in Example 2-2.

Example 2-2 SETROPTS command to inactivate the UNIXMAP class

```
SETROPTS NOGENCMD(UNIXMAP)
SETROPTS NOGENERIC(UNIXMAP)
SETROPTS NOCLASSACT(UNIXMAP)
```

Note: The same steps are needed for the **NOTELINK** and **NDSLINK** classes if they are in use at your installation.

10. The **COFVLFXx** member requires updates to remove references to the **IRRGMAP** and **IRRUMAP** definitions, because they are related to the **UNIXMAP** class. In our installation, member **COFVLF00** is in use, as shown in Figure 2-18.

```
/*
CLASS NAME(IRRGMAP)          /* Open Edition GID to GROUP mapping */
    EMAJ(GMAP)                /* MAJOR NAME OF IRRGMAP class */
                                /* Default MAXVIRT = 4096 4K blocks */
                                /*                      = 16Mb */
/*
CLASS NAME(IRRGMAP)          /* Open Edition UID to USER mapping */
    EMAJ(UMAP)                /* MAJOR NAME OF IRRGMAP class */
                                /* Default MAXVIRT = 4096 4K blocks */
                                /*                      = 16Mb */
/*
CLASS NAME(IRRSMP)           /* OpenEdition security packet */
    EMAJ(SMAP)
```

Figure 2-18 Sample COFVLFXx member in SYS1.PARMLIB

11. Run the **IRRUT200** utility again to check for the codes that have replaced OMVS UID/GID, LNOTES SNAME, and NDS UNAME, as shown in Figure 2-19 on page 30. Also, at the

bottom of the statistics section of the IRRUT200 job, there will be no references anymore to the UNIXMAP, NOTELINK and NDSLINK classes.

- **010302** - OMVS GID alias index key prefix
- **020802** - OMVS UID alias index key prefix
- **020C02** - LNOTES SNAME alias index key prefix
- **020D02** - NDS UNAME alias index key prefix

06F	0006	0103020076AE3B
088	0000	020802000000000
411	0000	020C02aims01
5C1	0000	020D02AIMS06

Figure 2-19 Extract from IRRUT200 after AIM migration

12. Replace the RACF backup database with a copy of the RACF Primary database, using the **IRRUT400** utility, as shown in Figure 2-20.

//BRIGGSZ	JOB	(999,P0K), 'COPY RACF', NOTIFY=&SYSUID,
//		CLASS=A, MSGCLASS=T, MSGLEVEL=(1,1)
//STEP1		EXEC PGM=IRRUT400, PARM='NOLOCKINPUT, FREESPACE(20)
//SYSPRINT	DD	SYSOUT=A
//INDD1	DD	DSN=SYS1.RACFESA, DISP=OLD
//OUTDD1	DD	DSN=SYS1.RACF.BKUP1, DISP=(,CATLG),
//		VOL=SER=PDGSY1,
//		SPACE=(CYL,(43,0,0),,CONTIG),
//		DCB=DSORG=PSU,
//		UNIT=3390

Figure 2-20 Sample IRRUT400 JCL

13. After successful completion of the **IRRUT400** utility, activate the backup RACF again, using the **RVARY ACTIVE** command, as follows:

```
RVARY ACTIVE,DATASET(SYS1.RACF.BKUP1)
```

You must respond to the WTO message through the operator console with the correct **RVARY SWITCH** password.

2.2.8 Space requirements

The space saving comes when stage 3 is entered and the mapping profiles (including their regular index entries) are deleted.

Mapping profiles are defined in the general resource template, and therefore their (regular) index key contains:

- ▶ 8 characters of class name (right padded with blanks) and a '-'
- ▶ UNIXMAP class has an alias preface of 'U' for UID and 'G' for GID
- ▶ Alias value

Thus, each mapping profile level 1 index entry has a length of 31—or in the case of OMVS, 32—plus the length of the alias. Therefore, an uncompressed level 1 mapping profile index entry is longer than a corresponding alias index entry.

Additionally, there is space taken up by the mapping profile itself within a datablock.

Table 2-4 shows a space saving example (ignoring index levels 2 through N, and compression), for a 1-1 mapped UID, whose base profile name length is 7.

Table 2-4 AIM space table

AIM	Generic Identity Mapping
ALIAS INDEX ENTRY = 32	MAPPING PROFILE INDEX ENTRY = 31+ MAPPING PROFILE = 117/256 (SLOT SIZE)
TOTAL 32	TOTAL 287+

A profile is not contiguous within a datablock. Each distinct segment (base segment or otherwise) of a profile begins upon its own 256 byte slot within a datablock. Therefore, if the segment information takes up less than 256 bytes (117 in this case), 256 bytes have still been reserved. The hex value of the UID is converted to decimal, and then each decimal nibble is converted to EBCDIC. This results in a length between 1 and 10.

Due to the nature of the alias index keys (3 hex bytes denoting template #, segment #, and field #, followed by the alias value), a simple range table would put all into the first split. That is because, previously, index keys were all in EBCDIC and only templates '01'X and '02'X contain them.

2.2.9 When there are more than 130 user IDs with the same UID

The **IRRIRA00** utility fails when it is used to move a database from stage 0 to stage 1 and the database contains more than 130 8 character User IDs mapping to the same GID/UID. The utility will fail after some higher number if your User IDs have fewer characters. Before **IRRIRA00** can run again against the database successfully, the number of mappings must be reduced and the **BLKUPD** utility must be used to correct some information in the ICB for the database.

Figure 2-21 shows the messages **IRRIRA00** issues when it detects that the database maps more user IDs to a single UID than it can handle. **IRRIRA00** does not proceed to stage 1.

```
IRR66017I The system is currently operating in stage 0.
IRR66016I Unexpected RACF manager return code updating entry JP in class
USER. Return code 132. Reason code 0.
IRR66009I Last entry processed successfully was HOERNER in class USER.
```

Figure 2-21 Error messages issued by IRRIRA00

Altering or deleting the user IDs that share the same UID will not eliminate these messages the next time **IRRIRA00** is run.

The following steps must be performed before **IRRIRA00** can successfully move a database to stage 1 again:

1. Familiarize yourself with the use of the **BLKUPD** command. It is documented in the *SecureWay Security Server for OS/390 (RACF) Diagnosis Guide*, SY27-2639
2. Issue the **BLKUPD** command shown in Example 2-3.

Example 2-3 BLKUPD command

```
READY
BLKUPD 'SYS1.RACFESA'
```

BLKUPD:

3. Read in the block at relative byte address 0, indicating that it will be updated as shown in Example 2-4.

Example 2-4 READ keyword of the BLKUPD command

BLKUPD:
READ 0 UPDATE
BLKUPD:

4. List the byte value at offset x'16F'. If it is not zero, then issue the replace command to set it to zero. Issue the **LIST NEW** keyword again to verify the change is correct. The commands are shown in Example 2-5.

Example 2-5 LIST and REP keyword example of BLKUPD

BLKUPD:
LIST NEW RANGE(X'16F',1)
RBA=000000000000
016F 01
* . *

BLKUPD:
REP x'00' OFFSET(x'16F') VER(x'01')
IRR63004I REPLACE complete.
BLKUPD:

5. List the 12 bytes at offset x'3E0' and replace them with zeros. Issue the **LIST NEW** keyword to verify the change, as shown in Example 2-6.

Example 2-6 LIST and REP keyword the BLKUPD command to update the ICB

LIST RANGE(X'3E0',12)
RBA=000000000000
03E0 00000005 20000000 00052000
* *

BLKUPD:
REP X'0000000000000000000000000000' OFFSET(X'3E0') VER(X'0000000520000000000000000000')
IRR63004I REPLACE complete.
BLKUPD:
LIST NEW RANGE(X'3E0',12)
RBA=000000000000
03E0 00000000 00000000 00000000
* *

BLKUPD:

6. Save the updates using the **END SAVE** keyword of the **BLKUPD** command, as shown in Example 2-7.

Example 2-7 Saving changes to the ICB using the END SAVE keyword of BLKUPD

BLKUPD:
END SAVE
IRR63013I READ ended. Block saved.
BLKUPD:

7. End the **BLKUPD** session using the **END** subcommand, as shown in Example 2-8 on page 33.

```
BLKUPD:
END
READY
```

These steps will leave some residual storage in the database. Use the **IRRUT400** utility to clean up the unused storage. You can now restart the migration process using the procedure outlined in 2.2.5, “Migrating and installation of AIM” on page 23.

2.2.10 New Serialization

This section describes the new Serialization:

1. ENQ SYSZRAC5 ALIAS SCOPE=SYSTEMS RNL=NO

- This allows the manager to hold the ENQ shared, but only if an input IPL is an alias Locate, or if it affects the relationship between a base profile and one of its participating AIM alias fields.
- **RVARY** will hold the ENQ exclusive.
- **IRRIRA00**, depending upon the stage change requested, will either hold the ENQ shared or exclusive and then shared.

2. ENQ SYSZRAC4 TEMPLATES#IIBASE PROFILE NAME SCOPE=SYSTEMS RNL=NO

Only the manager gets this ENQ. It is always exclusive for IPLs altering or deleting profiles which may contain an AIM alias field. The minor name consists of the hexadecimal value for the template number, coupled with the base profile name.

3. ENQ SYSZRAC4 TEMP#IISEG#IIFIELD#IIALIAS SCOPE=SYSTEMS RNL=NO

Only the manager gets this ENQ. It is gotten either exclusive (IPL operation add/alt/del) or shared (IPL operation alias locate). The minor name consists of the hexadecimal value for the template number, segment number, and field name, coupled with the alias value. OMVS UID/GID have hexadecimal alias values, NDS UNAME and LNOTES SNAME have EBCDIC alias values.

2.3 Digital certificate enhancements

The RACF component of the OS/390 Release 4 Security Server provides the ability to store digital certificates in the RACF database, and to associate a digital certificate with a RACF user ID. Typically, this is used to map a browser user certificate to a RACF user ID for controlling access to OS/390 resources.

A crucial part of implementing certificates is managing the certificates used by the server application, and ensuring an uncompromised chain of trust. These certificates also have associated encryption keys that are private and must not be revealed.

In OS/390 Release 8, the SecureWay Security Server provides functions to help manage server certificates and protect server private keys in a uniform and secure way. The primary application interface to these new functions is provided by Open Cryptographic Enhanced Plug-ins (OCEP), a new component for Security Server. The functions are incorporated into two plug-ins: one for data library services and one for trust policy manager. OCEP functions

are to be used by applications complying with Common Data Security Architecture (CDSA) standard interfaces. This makes it easier for application developers and independent software vendors to develop and port applications to the OS/390 platform. It also helps customers apply consistent security rules to e-business applications that use digital certificates.

A new function, Certificate Name Filtering (CNF) is introduced in OS/390 Security Server Version 2 Release 9 and made available to Version 2 Release 8 through APAR OW40129.

2.3.1 Digital certificate enhancements in Release 10

In OS/390 Security Server Version 2 Release 10, the **RACDCERT** command has some additional parameters for the keywords **GENCERT** and **EXPORT**, along with an additional keyword, **DEBUG**. The new components of the **RACDCERT** command are shown in Figure 2-22.

```
RACDCERT
GENCERT
[KEYUSAGE(HANDSHAKE DATAENCRYPT DOCSIGN CERTSIGN)]
[ALTNAME[(IP(numeric-ip-address)
          DOMAIN('internet-domain-name') EMAIL('email-address')
          UR('universal-resource-identifier'))]]
EXPORT [FORMAT(CERTDER CERTB64 PKCS12DER PKCS12B64)]
DEBUG
```

Figure 2-22 New components of the **RACDCERT** command

The **KEYUSAGE** keyword of the **RACDCERT GENCERT** command specifies the appropriate values for the KeyUsage certificate extension, of which one or more of the values might be coded. For Certificate Authority (CA) certificates, the default is **CERTSIGN** and is always set. There is no default for certificates that are not Certificate Authority certificates.

The subparameters of the keyword **KEYUSAGE** are explained in Table 2-5.

Table 2-5 Subparameters of keyword **KEYUSAGE**

Subparameter	Description
HANDSHAKE	Most commonly used for facilitates identification and key exchange during security handshakes, such as SSL, which sets the digitalSignature and keyEncipherment indicators. Used with server, clients, and firewalls.
DATAENCRYPT	Encrypts data, which sets the dataEncipherment indicator. Needed for firewalls.
DOCSIGN	Specifies a legally binding signature, which sets the nonRepudiation indicator.
CERTSIGN	Specifies a signature for other digital certificates and CRLs, which sets the keyCertSign and cRLSign indicators. Acts as, or is the Certificate Authority for signing certificates.

The **ALTNAME** keyword of the **RACDCERT GENCERT** specifies the appropriate values for the subjectAltName extension, of which one or more of the values might be coded. If required for the extension, RACF converts the entire value to ASCII.

Note: RACF assumes the terminal code page is IBM-1047 and translates to ASCII accordingly.

The subparameters of the keyword **ALTNAME** are explained in Table 2-6.

Table 2-6 Subparameter of keyword **ALTNAME**

Subparameter	Description
IP (numeric-ip-address)	Specifies a quoted string containing a fully qualified numeric-ip-address in IPV4 dotted decimal form, which is four decimal numbers (each number must be a value from 0-255) separated by period, for example 9.12.14.247
DOMAIN (internet-domain-name)	Specifies a quoted string containing a fully qualified internet-domain-name such as www.widgits.com. RACF does not check this value's validity.
EMAIL (email-address)	Specifies a quoted string containing a fully qualified e-mail-address such as homer at moes bar.com. RACF replaces the word "at" with the @ symbol (x'7C') to conform with RFC822. If RACF cannot locate the word at, it assumes the address is already in RFC822 form and makes no attempt to alter it other than converting it to ASCII.
URI (universal-resource-identifier)	Specifies the universal-resource-identifier, such as http://www.widgits.com. RACF does not check the validity of this value.

The **EXPORT FORMAT(format-type)** keyword of the **RACDCERT** command specifies the format of the exported certificate.

Important: While the **CERTDER** and **CERTB64** keywords indicate to export only a certificate, the **PKCS12DER** and **PKCS12B64** keywords export the certificate and the private key (which must exist and not be an ICSF key).

The package produced by specifying one of the **PKCS12xxx** keywords is encrypted, using the password specified according to the **PKCS#12** standard.

The values of the **FORMAT** keyword are displayed in Table 2-7.

Table 2-7 Values of the **FORMAT** keyword

Value	Description
CERTDER	Specifies a DER-encoded X.509 certificate.
CERTB64	Specifies a DER-encoded X.509 certificate that has been encoded using Base64.
PKCS12B64	Specifies a DER-encoded (then Base64-encoded) PKCS#12 package.
PKCS12DER	Specifies a DER-encoded PKCS#12 package. This value is now the more acceptable format to be specified.

The **PKCS12B64** keyword is the default if the **PASSWORD** keyword is specified, otherwise the **CERTB64** keyword is the default.

The **DEBUG** keyword of the **RACDCERT GENCERT** command displays additional diagnostic information pertaining to encryption calls and RACF-invoked **ICHEINTY ALTER**, **RACROUTE REQUEST=EXTRACT**, and **RACROUTE REQUEST=DEFINE** failures. When a problem is encountered, customers can use this keyword to gather diagnostic information for the IBM support center.

2.3.2 Generating a digital certificate with RACF

This section describes how to define a digital certificate using RACF.

1. Use the **RACDCERT GENCERT** command to define the digital certificate with one of the new parameters, **KEYUSAGE**.

Note: The LABEL keyword is case-sensitive.

Example 2-9 shows the command we used to create a digital certificate.

Example 2-9 RACDCERT GENCERT command

```
RACDCERT GENCERT SUBJECTSDN(CN('JONATHAN BRIGGS') OU('ITSO')  
O('IBM') L('POUGHKEEPSIE') SP('NEW YORK') C('US')) SIZE(1024)  
WITHLABEL('JONBOY') SIGNWITH(CERTAUTH LABEL('ITSO CA'))  
KEYUSAGE(HANDSHAKE)
```

The informational message IRRD113I shown in Figure 2-23 indicates a date problem. Basically we tried to sign a certificate with a CA certificate that expired before the certificate expired. That is why it is added with *NOTRUST*.

IRRD113I The certificate that you are creating has an incorrect date range.
The certificate is added with NOTRUST status.

Figure 2-23 Informational message from the RACDCERT GENCERT command

2. To confirm the expiry date on the ITSO CA Certificate Authority, display the CA certificate using the **RACDCERT CERTAUTH LIST** command, as follows:

```
RACDCERT CERTAUTH LIST(LABEL('ITSO CA'))
```

The start and end dates for the digital certificate of the Certificate Authority "ITSO CA" are shown in Figure 2-24.

```
Digital certificate information for CERTAUTH:  
  
Label: ITSO CA  
Status: TRUST  
Start Date: 2000/02/02 01:00:00  
End Date: 2001/02/03 00:59:59  
Serial Number:  
    >00<  
Issuer's Name:  
    >CN=IBM-ITSO MAIN CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=NY.C=US<  
Subject's Name:  
    >CN=IBM-ITSO MAIN CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=NY.C=US<  
Private Key Type: Non-ICSF  
Private Key Size: 1024  
Ring Associations:  
    Ring Owner: GRAAFF  
    Ring:  
        >pauls-keyring<  
    Ring Owner: TCP/IPU  
    Ring:  
        >telnetserver<
```

Figure 2-24 Output from RACDCERT CERTAUTH LIST command

List the digital certificate created earlier with the **RACDCERT LIST** command, as follows:

```
RACDCERT ID(BRIGGS) LIST(LABEL('JONBOY'))
```

From the output, you can see the digital certificate generated using the **RACDCERT GENCERT** command picked up today's date, along with the *NOTRUST* status, as shown in Figure 2-25.

```
Digital certificate information for user BRIGGS:

Label: JONBOY
Status: NOTRUST
Start Date: 2000/06/14 00:00:00
End Date: 2001/06/14 23:59:59
Serial Number:
    >08<
Issuer's Name:
>CN=IBM-ITSO MAIN CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=NY.C=US<
Subject's Name:
    >CN=Jonathan Briggs.OU=itso.O=ibm.L=poughkeepsie.SP=new york.
C=us<
Key Usage: HANDSHAKE
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
*** No rings associated ***
```

Figure 2-25 RACDCERT LIST command output example

3. Amend the newly created digital certificate, to become *trusted*, with the following command:

```
RACDCERT ID(BRIGGS) ALTER(LABEL('JONBOY')) TRUST
```

If you now use the **RACDCERT LIST** command again, you see that the digital certificate is now in *trust* status, as shown in Figure 2-26.

```
Digital certificate information for user BRIGGS:

Label: JONBOY
Status: TRUST
Start Date: 2000/06/14 00:00:00
End Date: 2001/06/14 23:59:59
Serial Number:
    >08<
Issuer's Name:
>CN=IBM-ITSO MAIN CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=NY.C=US<
Subject's Name:
    >CN=Jonathan Briggs.OU=itso.O=ibm.L=poughkeepsie.SP=new york.
C=us<
Key Usage: HANDSHAKE
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
*** No rings associated ***
```

Figure 2-26 RACDCERT LIST command output example

4. Now that you have generated the certificate for a client (Jonathan Briggs's digital certificate), export the digital certificate to an OS/390 dataset using the **RACDCERT EXPORT** command.

The digital certificate can be exported in any one of the four formats mentioned in Table 2-7 on page 35. In this example we used **PKCS12DER** format, as shown in Example 2-10.

Example 2-10 RACDCERT EXPORT command

```
RACDCERT ID(BRIGGS) EXPORT(LABEL('JONBOY')) DSN(LEMON.P12)
FORMAT(PKCS12DER) PASSWORD('password')
```

The output dataset containing the digital certificate in a DER-encoded X.509 certificate format can now be imported into your browser. The procedure to import certificates is different for Netscape Navigator and Microsoft's Internet Explorer. See the next section for a discussion of this procedure.

2.3.3 Installing a RACF digital certificate into your browser

To install the digital certificate you just generated into a browser, you first have to transfer it to your workstation (PC). To download the digital certificate to the workstation, take the following steps:

1. Open a DOS session from Windows by selecting **Start -> Programs -> Command Prompt**.
2. Change to the TEMP directory, into which the digital certificate is downloaded, by typing `cd` the `C:` prompt.
3. Start an FTP session with the OS/390 host system. Once connected, enter your RACF user ID and password.
4. After successfully connecting, change directory (command **CD**) to the high-level qualifier (briggs) of the data set where the digital certificate is stored.
5. Type "bin" so the digital certificate is downloaded in binary format.
6. Issue the `get` command for the digital certificate.

Figure 2-27 on page 38 shows the sequence of events.

The following two sections describe how to install the certificate on Microsoft's Internet Explorer and on Netscape Navigator.

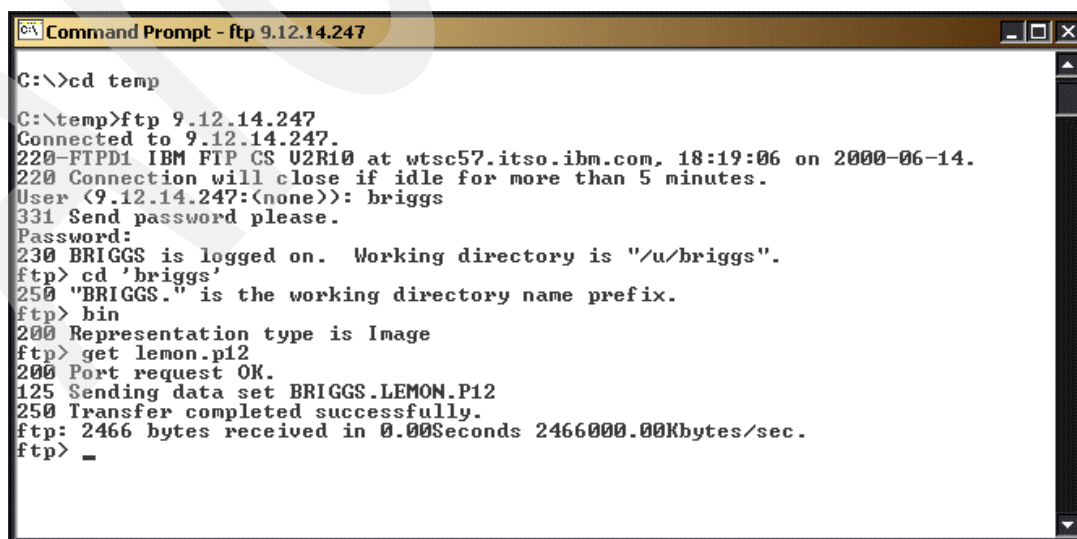


Figure 2-27 Command prompt screen

Importing the digital certificate to Internet Explorer

The following steps will import the digital certificate into the Internet Explorer.

1. Use Windows Explorer to find and select the file into which you saved the digital certificate. Double-click the file to invoke the Certificate Import Wizard program, as shown in Figure 2-28.



Figure 2-28 Certificate Import Wizard welcome screen

The Certificate Import Wizard guides you through the process to install the digital certificate into the correct certificate store of the Internet Explorer browser.

2. Click **Next** to continue the installation of the digital certificate. Figure 2-29 shows the next screen of the Certificate Import Wizard.

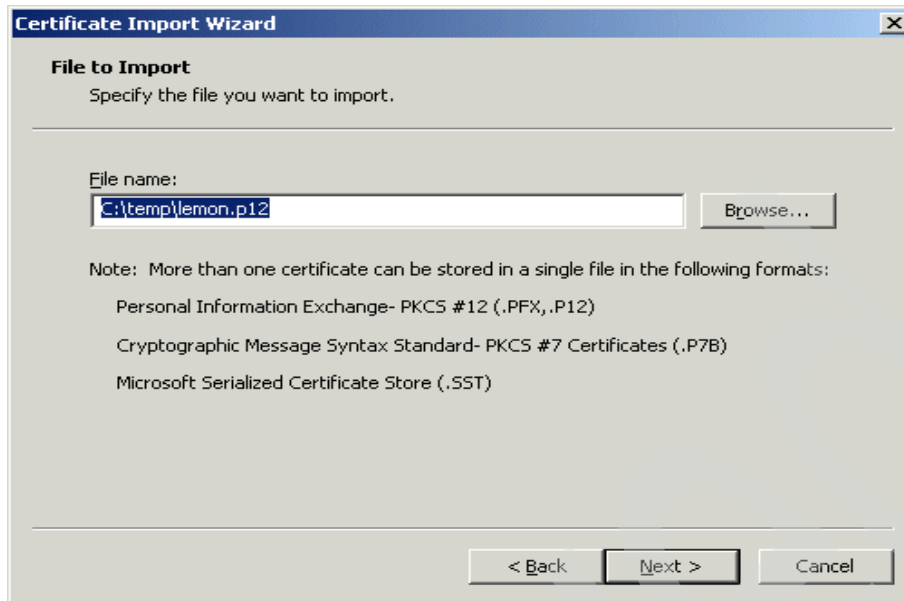


Figure 2-29 File to Import screen

Verify that the path is correct and click **Next**. Figure 2-30 on page 40 shows the next screen, the Password prompt screen. This is the password you specified when you exported the file using the `RACDCERT EXPORT` command. Select the check box option next to "Mark the private key as exportable," then click **Next**.

Note: If you do not select this check box, you are not able to export the certificate with the private key.



Figure 2-30 Password prompt screen

3. The screen shown in Figure 2-31 allows you to specify where the key is to be stored. We recommend that you select “Automatically select the certificate store based on the type of certificate.” This lets the Certificate Import Wizard automatically decide where to store the digital certificate.

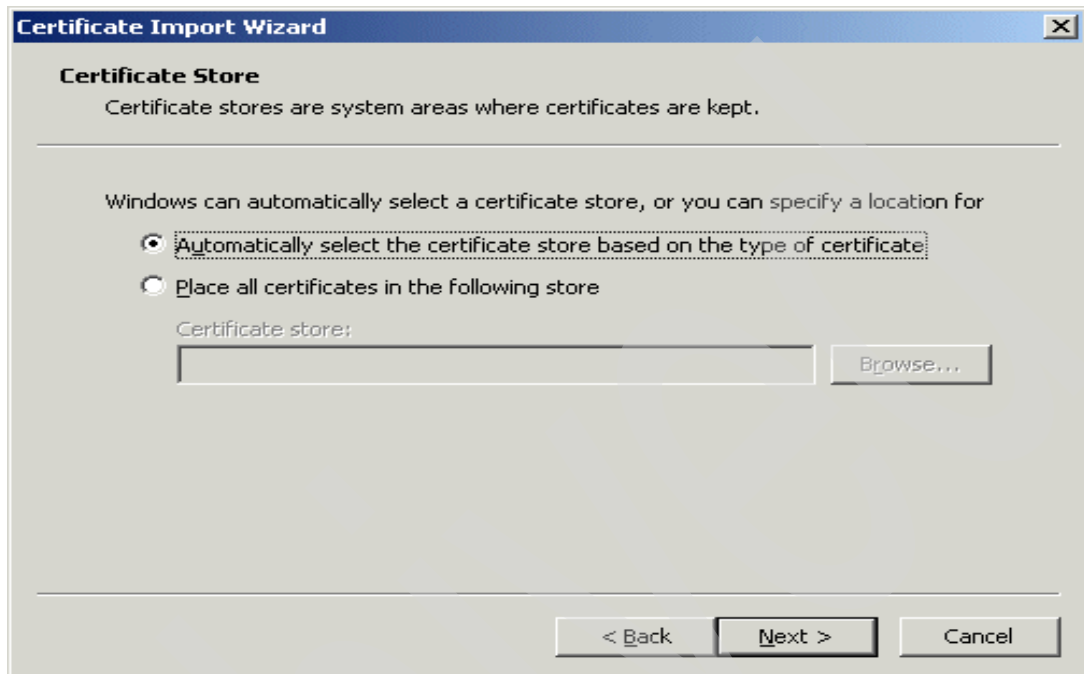


Figure 2-31 Certificate store screen

4. Click **Next** to continue the installation. The screen returned, shown in Figure 2-32 on page 41, informs you that installation of the digital certificate is complete. Click **Finish** to continue.



Figure 2-32 the Complete the Certificate import wizard screen

A successful import notification is displayed, as shown in Figure 2-33.



Figure 2-33 Certificate import wizard OK screen

5. Verify that the digital certificate indeed allows for client authentication. To do so, select **Tools -> Internet Options**. This brings up the Internet Options panel, shown in Figure 2-34 on page 42. Click the **Content** tab.

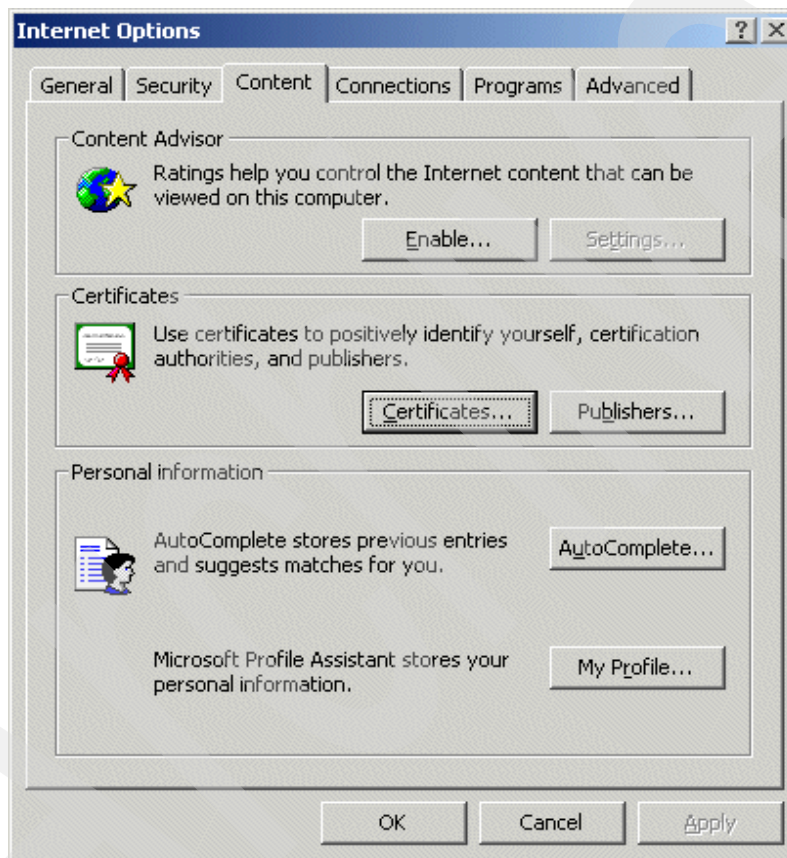


Figure 2-34 Internet Options panel

Click **Certificates** to bring up a list of digital certificates. Highlight the digital certificate and click the **Advanced** button, as shown in Figure 2-35 on page 43.

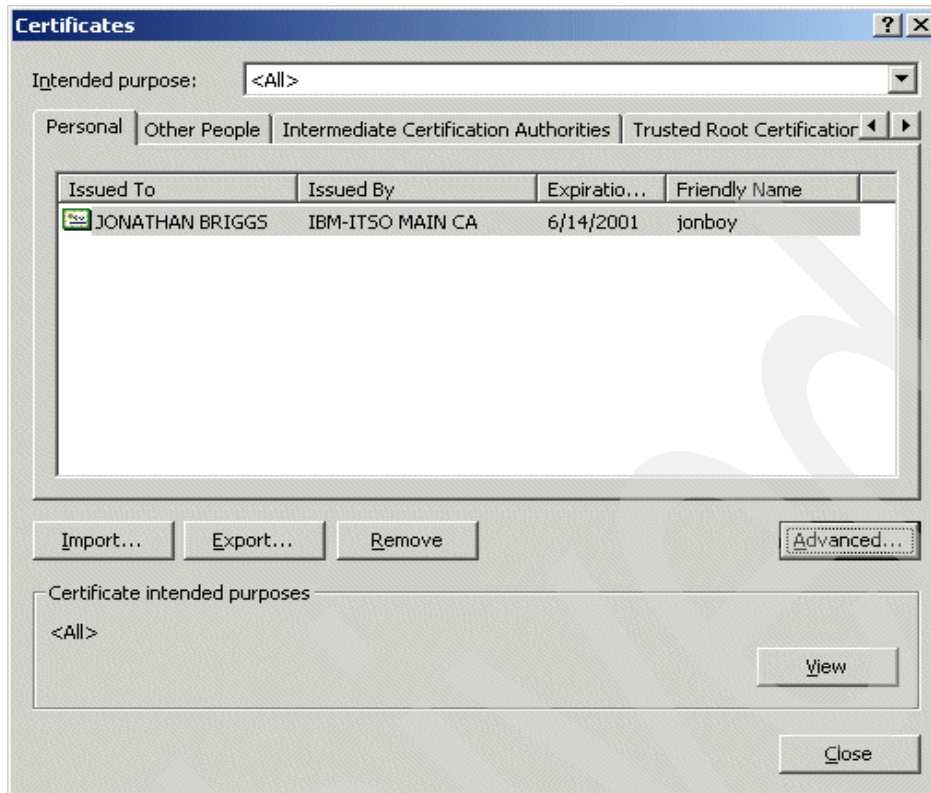


Figure 2-35 Certificates screen

The Advanced screen appears with various certificate purposes checked. By default, the Client Authentication box is not selected. Check the “Client Authentication” box, then click **OK** to save, as shown in Figure 2-36 on page 44.

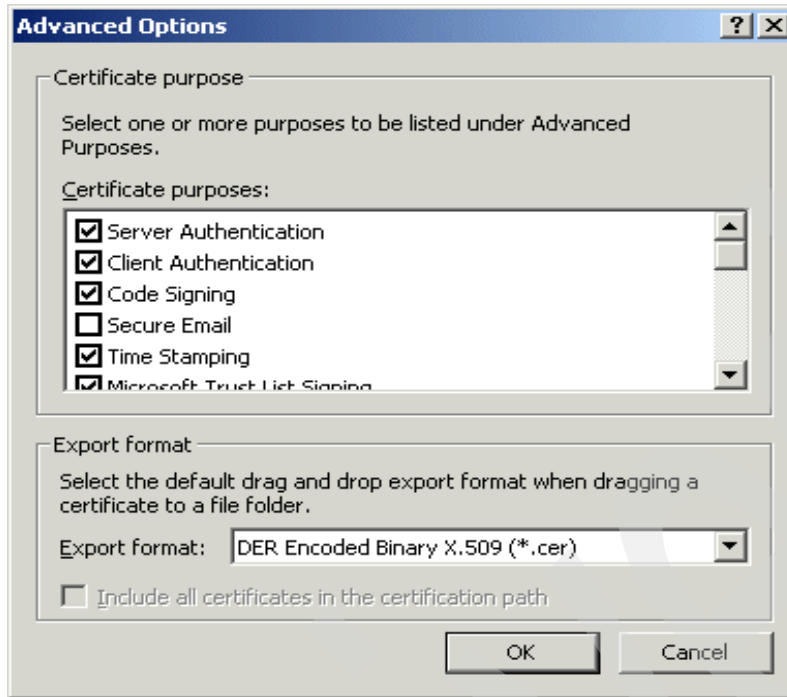


Figure 2-36 Advanced Options screen

Note: The check mark for the Client Authentication box may or may not be necessary. We tried it with various certificates and sometimes it was needed while other times it was not.

Importing the digital certificate to Netscape Navigator

The following steps will import the digital certificate into the Netscape Navigator:

1. Start Netscape Navigator by clicking its icon on your desktop, or by any other means you normally use to start Netscape Navigator. Once Netscape Navigator is running, click the **Security** icon on the task bar. This opens up the Security Info panel shown in Figure 2-37 on page 45.

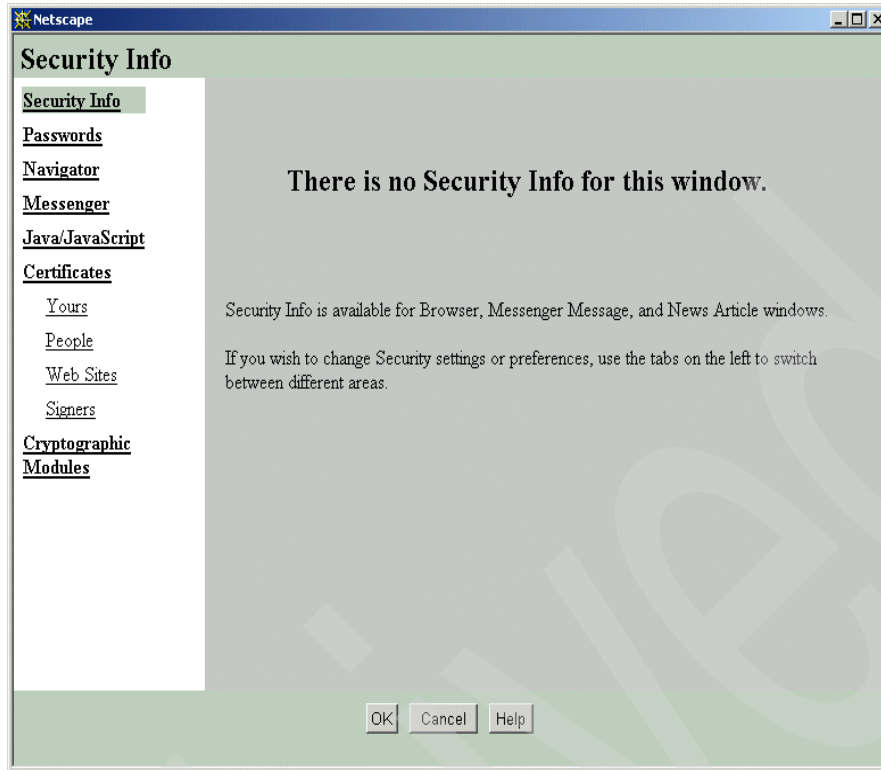


Figure 2-37 Netscape Navigator Security Info panel

2. The Security window shows a Certificate option. To import the digital certificate, select **Yours** for this option. The Security window changes, displaying your digital certificates stored in the certificate database, as shown in Figure 2-38 on page 46.

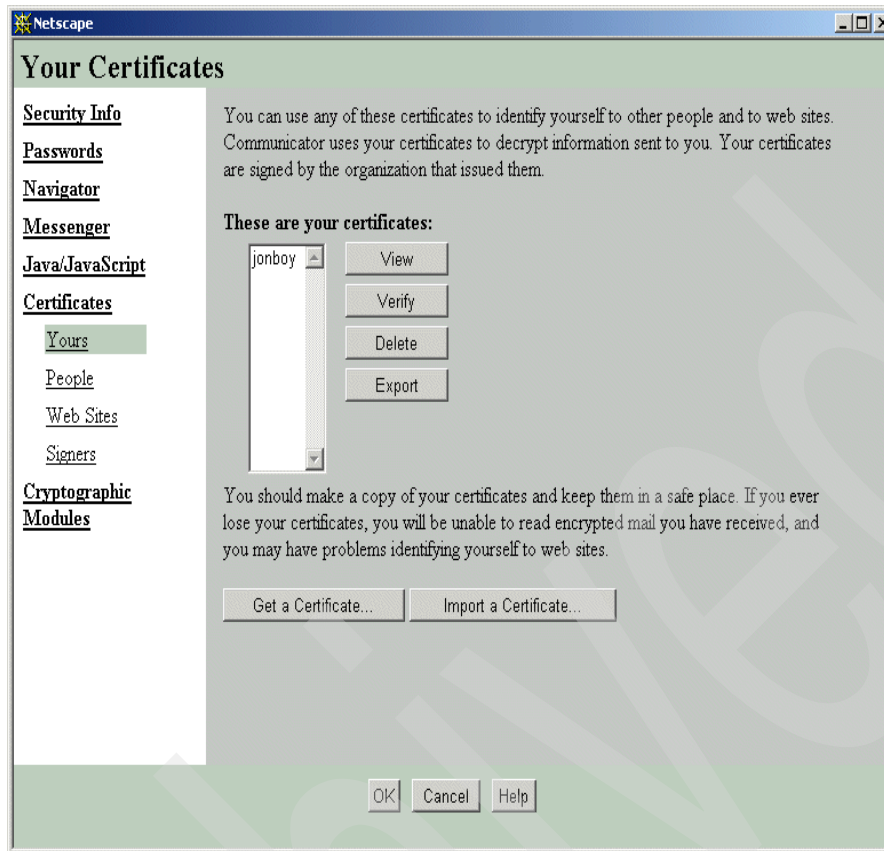


Figure 2-38 Your Certificates panel

3. Click the **Import a Certificate** button, which takes you into the Windows Explorer. Locate the digital certificate to be imported and open the file, as shown in Figure 2-39.

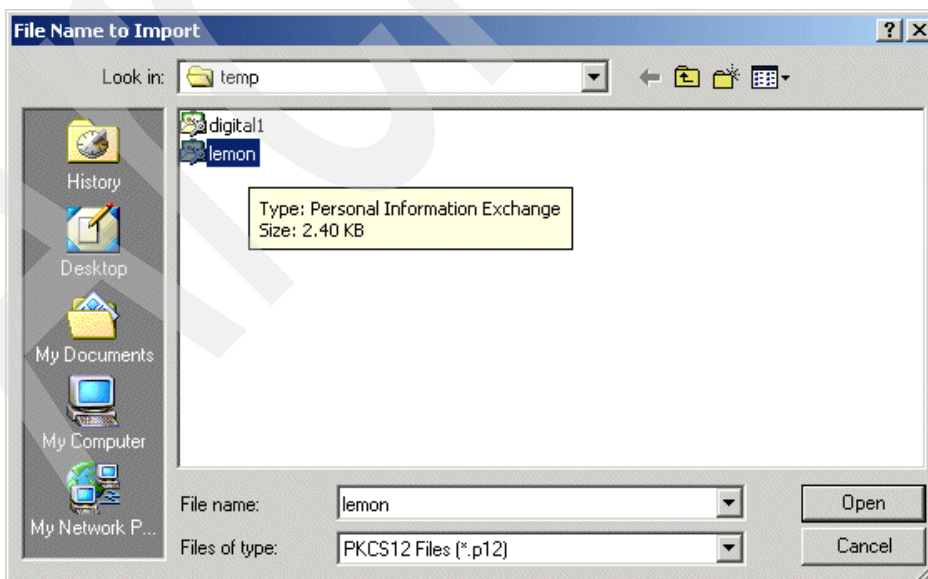


Figure 2-39 File name to import screen

A confirmation message is displayed, as shown in Figure 2-40.

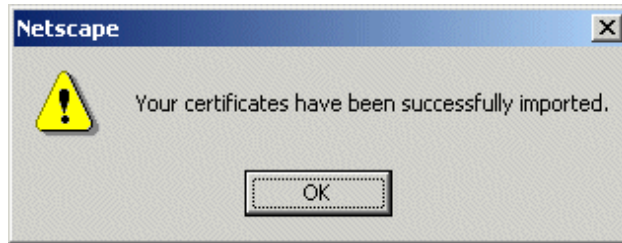


Figure 2-40 Confirmation prompt

Now that it has been imported, you can view information about the digital certificate by highlighting the certificate name and clicking **View**, as shown in Figure 2-41.

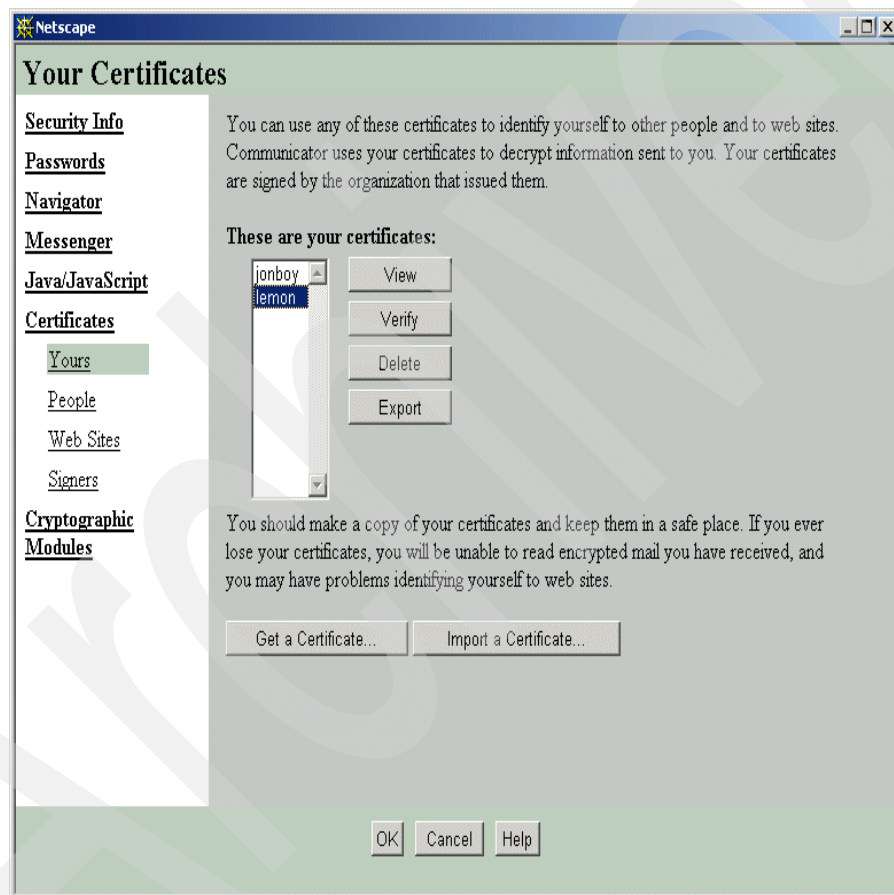


Figure 2-41 Your Certificates panel

Figure 2-42 on page 48 displays information about the digital certificate, namely, who the certificate belongs to and who the issuer is.

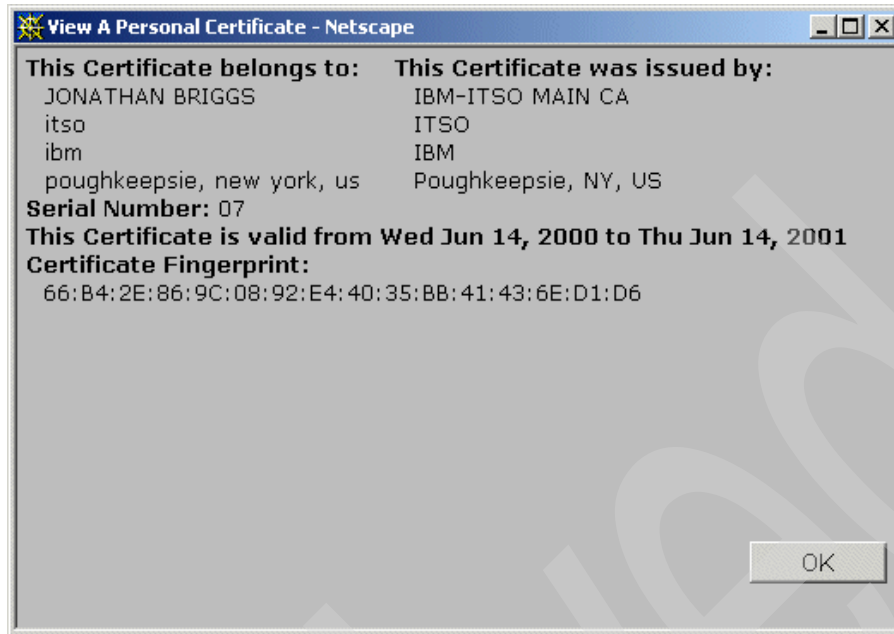


Figure 2-42 Details of digital certificate

2.4 PKIServ - Web interface to certificate generation

As you may have determined from reading through 2.3, “Digital certificate enhancements” on page 33, installing a client certificate using the procedures outlined there is not very user friendly.

RACF APAR OW45211 and SAF APAR OW45212 introduce a Web-based front-end to digital certificate generation for clients using a Web browser such as Netscape Navigator or Microsoft’s Internet Explorer (IE).

This extends the Public Key Infrastructure (PKI) services supplied with APAR OW31933, which provided the RACF Auto Registration (RAR) application. The RAR application shipped sample HTML and REXX code to provide a Web-based front-end for linking a client’s digital certificate with its associated RACF user ID.

The new services provide REXX code to generate digital certificates. Since this is aimed at replacing the functionality supplied earlier by the CA Servlet that shipped with the IBM HTTP Server for OS/390, as of OS/390 Version 2 Release 10, the CA Servlet function is no longer shipped.

2.4.1 Overview

Figure 2-43 on page 49 shows an overview of the components involved when generating a client’s digital certificate using the Web-based front-end.

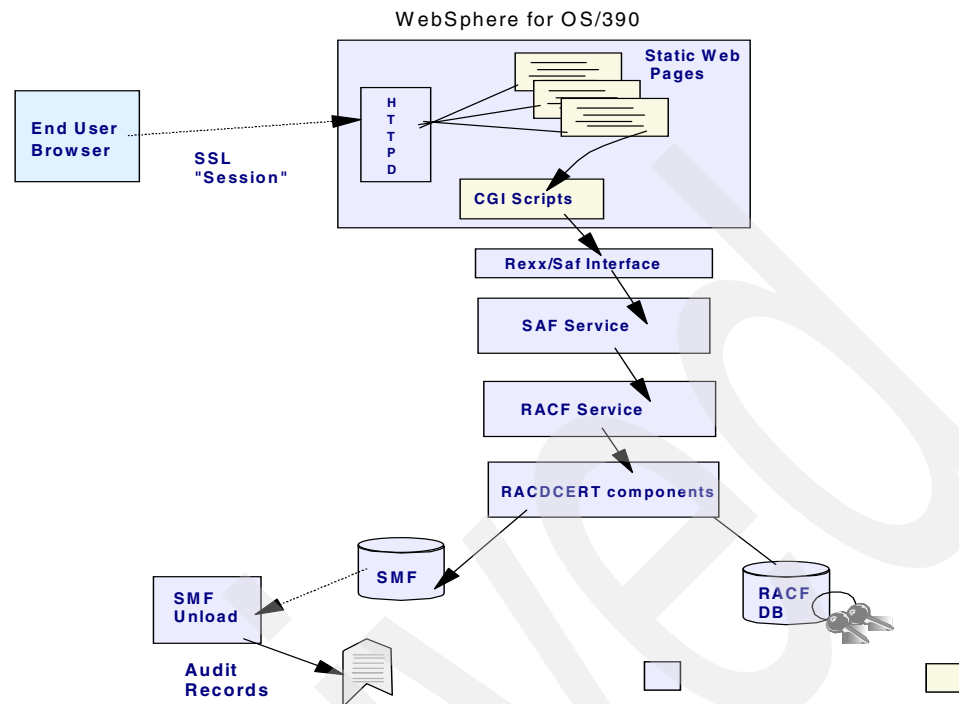


Figure 2-43 Overview of the Web-based process of generating client certificates

The certificate request flow is as follows:

1. The client selects the Web page, either linked or directly, to generate a digital certificate over an SSL established session.

Note: SSL server-side authentication only.

2. The client fills in the Web page with the required information to make up the distinguished name of the client, like common name, organization, and so forth.

Note: The required fields can be tailored according to the options defined in the configuration file, which is discussed in 2.4.3, “The configuration file” on page 51.

3. The client submits the request and new RACF callable services invoke the existing RACF **RACDCERT** facilities to generate a digital certificate request.
4. The client is notified of the successful or unsuccessful certificate request.
The client is issued a transaction identifier, which is used to pick up the approved certificate.
5. The client can now pick up his certificate based on the transaction identifier and install it into the browser. The client also has the option to defer the pickup of his certificate to a later date or time.

Restriction: There is no formal approval process here. The setup assumes the client is authorized by RACF to generate a certificate and have it signed by the Certificate Authority selected. This is explained in the customization section.

2.4.2 Directory structure provided

RACF APAR OW45211 and SAF APAR OW45212 supply the files and separate directories according to the directory structure shown in Figure 2-44.

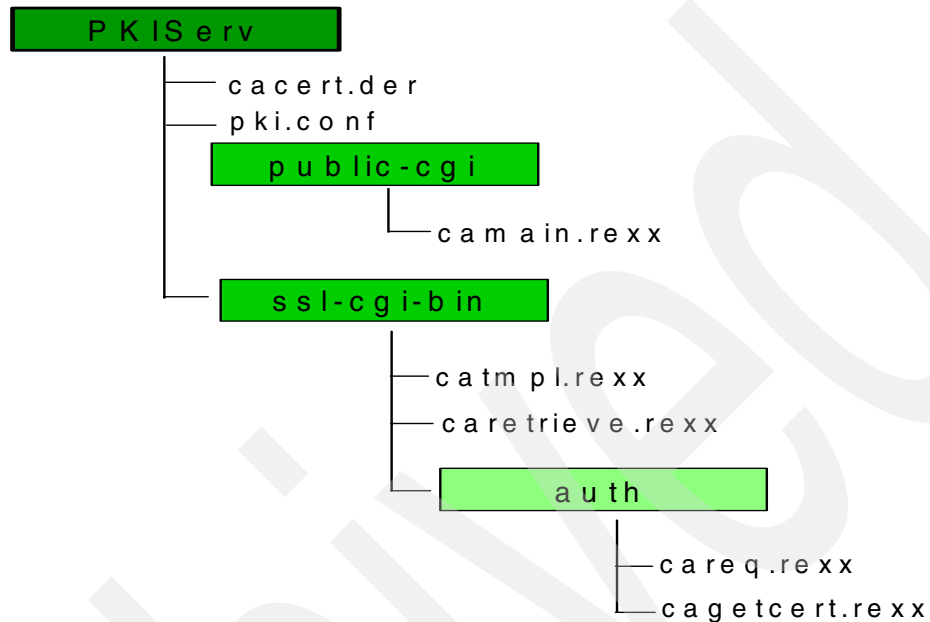


Figure 2-44 Directory structure provided by APAR

The directory structure consists of the following:

PKIServ This is the root directory, which contains the following files:

cacert.der The local certificate authority (CA) signing certificate

pki.conf The main configuration file, which controls the setup of the new functions provided

The PKIServ directory contains two subdirectories, called:

public-cgi This directory contains the main program, called CAMAIN.REXX.

ssl-cgi-bin This directory contains the executables, which need to be executed only when an SSL connection is established between the client and the Web server. The ssl-cgi-bin directory contains one more subdirectory:

auth This directory contains the code required to request and retrieve a digital certificate, and requires both an SSL established session and a RACF user ID and password.

Restriction: The supplied directory structure can be changed if required, but would require changes to the files themselves. Of course, the main directory PRIServ can be installed as a subdirectory structure within your file system if required.

2.4.3 The configuration file

The configuration of the new PKI services is done through a file, called `pki.conf`, located in the root directory. It introduces the concept of certificate templates, which define the fields that comprise a specific certificate request. These templates are examined by the REXX connector logic within the Web server to establish defaults and identify the fields which can be changed on a certificate request. Certificate templates are shipped in pseudo HTML, which enables you to easily modify them to suit your installation's needs. In essence, much of the intelligence with respect to certificate generation is contained within the certificate template, with the new SAF service providing the basic storage and management primitives.

Structure of the configuration file

The file contains a mixture of true HTML and HTML-like tags. The main tags divide the file into sections: `APPLICATION`, `TEMPLATE`, and `INSERT`, where `APPLICATION` and `TEMPLATE` may contain various subsections, named fields, and substitution variables as explained next.

<APPLICATION NAME=appl-name> ... </APPLICATION>

This section identifies the applications that will make use of PKI Services for OS/390. The product ships with one application defined, "PKISERV." This section contains one subsection, `CONTENT`.

1. `<CONTENT> ... </CONTENT>`

This subsection contains the HTML to be presented to the end user requesting and retrieving certificates. The subsection should contain one or more named fields identifying certificate templates to be used for requesting or managing certificates through this application. (See the next section for a description of named fields.) These template names should match the HTML selection values associated with them.

<TEMPLATE NAME=tmpl-name> ... </TEMPLATE>

This section defines the certificate templates referenced in the `APPLICATION` sections. Applicable subsections are:

1. `<CONTENT> ... </CONTENT>`

This subsection contains the HTML to be presented to the end user requesting certificates of this type. Any named fields in this subsection are interpreted as certificate field names defined by `INSERT` sections. For PKISERV, the `INSERT` sections are included as part of the HTML presented to the end user (that is, the end user provides values for these fields.) Named fields in this subsection are considered optional if the named field contains more than one word within the `%%` delimiters, for example, `%%AltName (Optional)%%`. The user need not supply a value for `AltName`.

2. `<APPL> ... </APPL>`

This subsection identifies certificate fields that the application itself should provide values for. This subsection should contain named fields only, one per line. Currently, the only supported named field allowed in this section is "UserId."

3. `<CONSTANT> ... </CONSTANT>`

This subsection identifies certificate fields that have a constant (hard-coded) value for everyone. This subsection should contain named fields only, one per line. The syntax for specifying the values is `%%field-name=field-value%%`, for example `%%KeyUsage=handshake%%`.

4. <SUCCESSCONTENT> ... </SUCCESSCONTENT>

This subsection contains the HTML to be presented to the end user when the certificate request was submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

5. <FAILURECONTENT> ... </FAILURECONTENT>

This subsection contains the HTML to be presented to the end user when the certificate request submit failed. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

6. <RETRIEVECONTENT> ... </RETRIEVECONTENT>

This subsection contains the HTML to be presented to the end user to enable certificate retrieval. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

7. <RETURNCERT> ... </RETURNCERT>

This subsection contains the HTML to be presented to the end user upon successful certificate retrieval. For PKISERV, if the certificate being retrieved is a browser certificate, then this section must contain a single line containing a browser qualified INSERT name, for example %%returnbrowsercert[browser type]%%. Additionally, INSERTs for Netscape (returnbrowsercertNS) and Internet Explorer (returnbrowsercertIE) containing browser-specific HTML for returning certificates must be defined elsewhere in the configuration file. If the certificate being retrieved is a server certificate, this section should contain the HTML necessary to present the certificate to the user as text.

8. <INSERT NAME=insert-name> ... </INSERT>

This section contains HTML that either describes a certificate field or defines other common HTML that may be referenced in the TEMPLATE sections. INSERTs are referenced elsewhere by using a named field of the form %%insert-name%%.

Named fields are delineated with pairs of percent signs (%%), for example %%Label%%. Their meaning is specific to the section they are contained in. Named fields are case sensitive. Named fields are also used to reference common includable HTML. Note, PKISERV treats named fields that begin with a dash as just includable code. Any special meaning a named field may have, given the section it's contained in, is ignored if it begins with a dash. For example, if %%-pagefooter%% was specified in a TEMPLATE CONTENT section, -pagefooter would not be considered a certificate field name. However, the INSERT with the name -pagefooter would be included in the HTML page presented to the end user.

Substitution variables are delineated with square brackets, for example [base64cert]. They represent variables that get replaced with an actual value at run time. Substitution variables are case sensitive. The valid substitution variables are:

transactionid	Unique value returned from a certificate request.
tmplname	Certificate template name. Primed from the HTML tag <SELECT NAME="Template"> in the <APPLICATION NAME=PKISERV> section. This is selected by the end user on the first Web page.
base64cert	The requested certificate base64 encoded.
iecert	The requested certificate in a form the Microsoft Internet Explorer accepts.
browser type	Special substitution variable to be used to qualify named fields only. Its use enables the different browsers, Netscape and Internet Explorer, to perform browser-specific operations, that is, Netscape uses a KEYGEN HTML tag

to generate a public/private key pair while Internet Explorer uses ACTIVEX controls. For example, if `%%PublicKey[browsertype]%%` was specified in a TEMPLATE CONTENT section referenced by a user with the Netscape Navigator browser, then INSERT PublicKeyNS would be included; if the user's browser was the Microsoft Internet Explorer, INSERT PublicKeyIE would be included.

optfield

Special substitution variable that should be placed in any certificate field name INSERT where the value may be supplied by the end user. It enables the field to be displayed as optional if desired.

Attention: Depending on where a substitution variable is used, it may not have a valid meaning. For example, base64cert would be meaningless prior to the certificate being retrieved. The value of [base64cert] would be the empty string (also known as NULL) in this case.

2.4.4 User interface

The end user interface is browser-based static HTML pages. These are produced by the REXX connector CGIs by reading the installation-customized configuration file. The user is required to have a RACF user ID and password for authentication. This is similar in concept to the RACF Auto-Registration (RAR) samples. The user's (or application's) access to new and existing RACF FACILITY class profiles determines if the user is authorized to generate and retrieve certificates through this interface. Installation-customizable certificate templates contained within the configuration file control which fields are user-customizable via the HTML dialogues.

The PKIServ application enables a user to request a digital certificate for a client (browser) or server. Figure 2-45 shows the first screen in a flow to request a digital certificate.

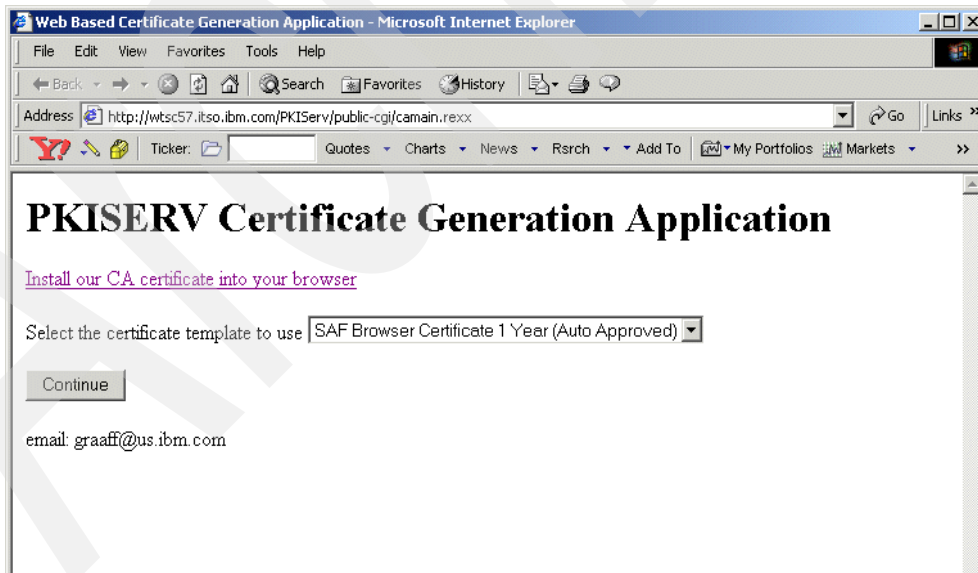


Figure 2-45 PKISERV certificate generation application page

This is the main page where the user can select a certificate template (model). The REXX CGI (CAMAIN.REXX) produces this page by reading the configuration file and answering everything in the <CONTENT> section under <APPLICATION NAME=PKISERV>. This page is not SSL-protected, but subsequent pages in this flow are.

The hypertext link *downloads* and installs the OS/390 Root CA certificate into the user's Web browser.

There are two selection choices to make on this page to request a digital certificate:

1. **SAF Browser Certificate 1 Year (Auto Approved)**
This option is intended to request a digital certificate for an end user/client. The certificate has a validity period of one year (365 days) and is automatically approved.
2. **SAF Server Certificate 1 Year (Auto Approved)**
This option is intended to request a digital certificate for a server, like the HTTP, LDAP, TN3270 or other server. Again the certificate has a validity period of one year (365 days) and is automatically approved.

Tip: These are the options that are distributed as samples. They can be customized to suit the installation's needs, as we discuss later.

SAF Browser Certificate 1 Year

To request a digital certificate for a client (browser), select the option **SAF Browser Certificate 1 Year (auto approved)**.

Click the **Continue** button to proceed to the next dialogue in the sequence, shown Figure 2-46.

Web Based SAF Certificate Generation Application Pg 2 - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites History

Address <https://wtsc57.itso.ibm.com/PKIServ/ssl-cgi-bin/catmpl.rexx?Template=1+Year+SAF+Browser+Certificate> Go Links »

Y! Ticker Quotes Charts News Rsrch Add To My Portfolios Markets »

SAF Browser Certificate 1 Year (Auto Approved)

Choose one of the following:

- **Request a New Certificate**
Enter values for the following field(s)
Label assigned to certificate being requested

Select the following key information
Cryptographic Service Provider
Use Smart Card for private key protection?
- **Pick Up a Previously Issued Certificate**

email: graaff@us.ibm.com

Done Internet

Figure 2-46 PKISERV certificate generation application: Browser certificate page (1)

Once a template is selected, enter the information permitted for that template (in this case the certificate label and key size). The REXX CGI (**CATMPL.REXX**) produces this page (Figure 2-46 on page 54) by reading the configuration file and answering everything in the <CONTENT> section under the <TEMPLATE> selected by the user.

Attention: This page is SSL protected.

There are only two fields generated on this page that require a selection:

1. %%Label%%

This field indicates to specify the label associated with the RACF certificate being generated.

2. %%PublicKey browsertype%%

This field generates a so-called INSERT based on the browser used. In this case, we use Internet Explorer and select the INSERT PublicKeyIE. This provides us with a selection of the Cryptographic Service Providers (CSPs) Internet Explorer supports to generate a public/private keypair.

Restriction: Depending on the operating system you are using and the version of the browser, various CSPs may or may not show.

In our example, we are using Windows 2000 Professional Edition and Microsoft's Internet Explorer 5.5 with a cipher strength of 128 bit.

Again the page can be customized to suit your needs and allow for additional fields used in the certificate to be specified, like:

- ▶ Common Name (CN)
- ▶ Organization (O)
- ▶ Organizational Unit (OU)

This page can also be used to retrieve a certificate from a request that was submitted earlier.

If any required fields are not filled in, the CGI produces an error.

When **Continue** is clicked, you are prompted to enter a valid user ID/password, as shown in Figure 2-47.



Figure 2-47 RACF user ID and password prompt window

To be allowed to request a digital certificate, a user ID requires authorization to various RACF FACILITY class profiles. These profiles are discussed in “Determine RACF access to the RACDCERT services” on page 70.

Enter your user ID and password; if you are authorized to personal certificate services, the next page is displayed, as shown in Figure 2-48.

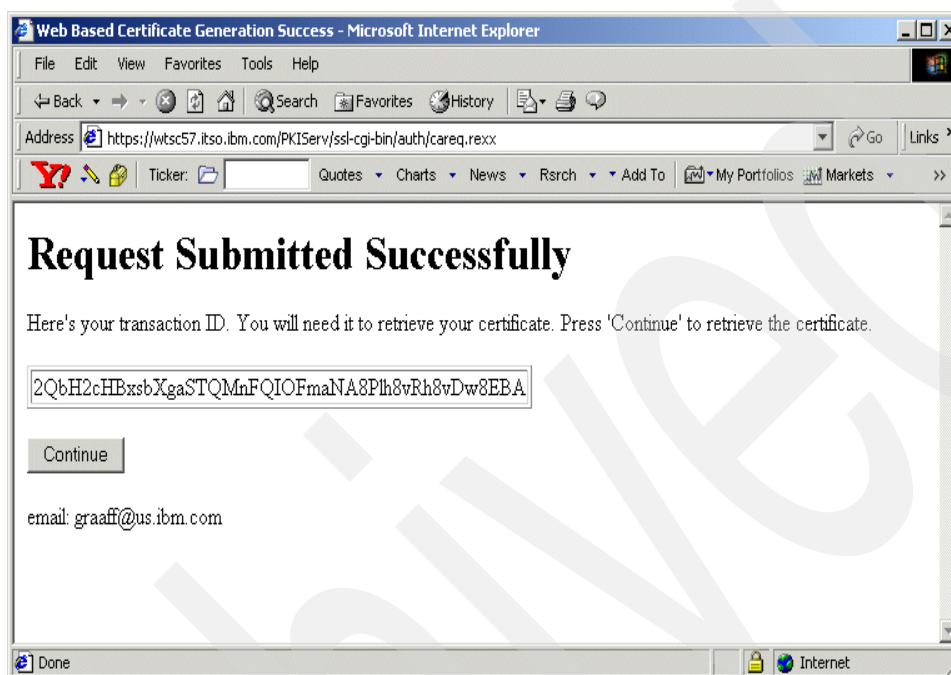


Figure 2-48 Certificate request submitted successful page

The user-provided fields are gathered, along with fields specified in the <APPL> and <CONSTANT> sections under the <TEMPLATE> selected by the user. All field values become parameters and the request is submitted to RACF (**IRRSPXGL** is called). If successful, a *transaction ID* (also called a Certificate ID) is returned. The *Certificate ID* uniquely identifies the newly created certificate in RACF. The REXX CGI (**CAREQ.REXX**) produces this page by reading the configuration file and answering everything in the <SUCCESSCONTENT> section under the <TEMPLATE> selected by the user. The digital certificate is now generated and can be picked up using the so-called transaction ID. If you take a look at RACF, you see the certificate as well, using the **RACDCERT LIST** command, as shown in Figure 2-49 on page 57.

```
raccert list(label('Paul IE cert 09/24/2000'))

Digital certificate information for user GRAAFF:

Label: Paul IE cert 09/24/2000
Certificate ID: 2QbH2cHBxsbXgaSTQMnFQIOFmaNA8P1h8vRh8vDw8EBA
Status: TRUST
Start Date: 2000/09/24 00:00:00
End Date: 2001/09/24 23:59:59
Serial Number:
    >02<
Issuer's Name:
    >CN=IBM-ITSO PDG CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=New York.C=US<
Subject's Name:
    >OU=ITSO.O=IBM.C=US<
Key Usage: HANDSHAKE
Private Key Type: None
Ring Associations:
*** No rings associated ***
```

Figure 2-49 RACDCERT LIST output from the generated certificate using PKISERV

As you can see from the **RACDCERT LIST** output, it shows the LABEL name you requested, the transaction ID (Certificate ID), and it indicates that there is no private key.

Note: The private key is on the workstation, not on the host (RACF) side.

To complete the digital certificate request process, you need to retrieve the certificate from RACF. To do this, click the **Submit** button to progress to the next page.

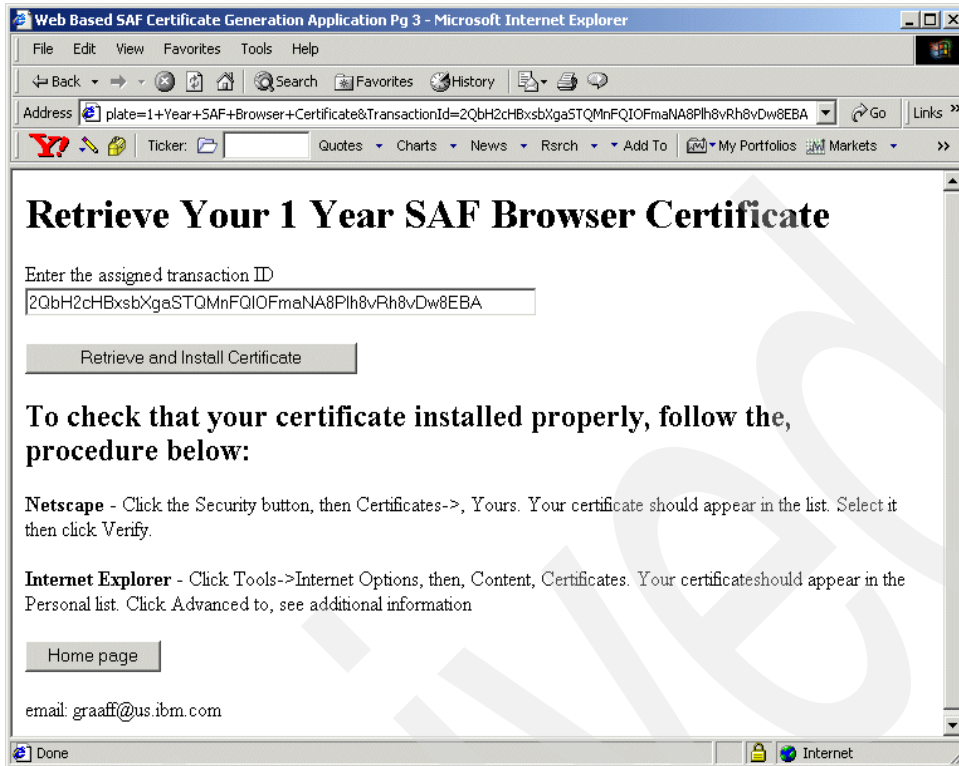


Figure 2-50 Retrieve browser certificate page

Clicking the **Submit** button presents you with the page shown in Figure 2-50, to retrieve the certificate. The REXX CGI (**CARETRIEVE.REXX**) produces this page by reading the configuration file and answering everything in the <RETRIEVECERT> section under the <TEMPLATE> selected by the user.

Figure 2-50 also explains how you can check that the certificate is properly installed.

To retrieve the certificate, click the **Retrieve and Install Certificate** button. Figure 2-51 on page 59 shows the install page for Internet Explorer.

If the certificate being retrieved is a browser certificate, clicking the submit button will download the certificate directly into the browser certificate store. If the certificate being retrieved is a server certificate, you will be presented with a page that enables you to cut and paste the certificate to a file. The REXX CGI (**CAGETCERT.REXX**) produces this page by reading the configuration file and answering everything in the <RETURNCERT> section under the <TEMPLATE> selected by the user.

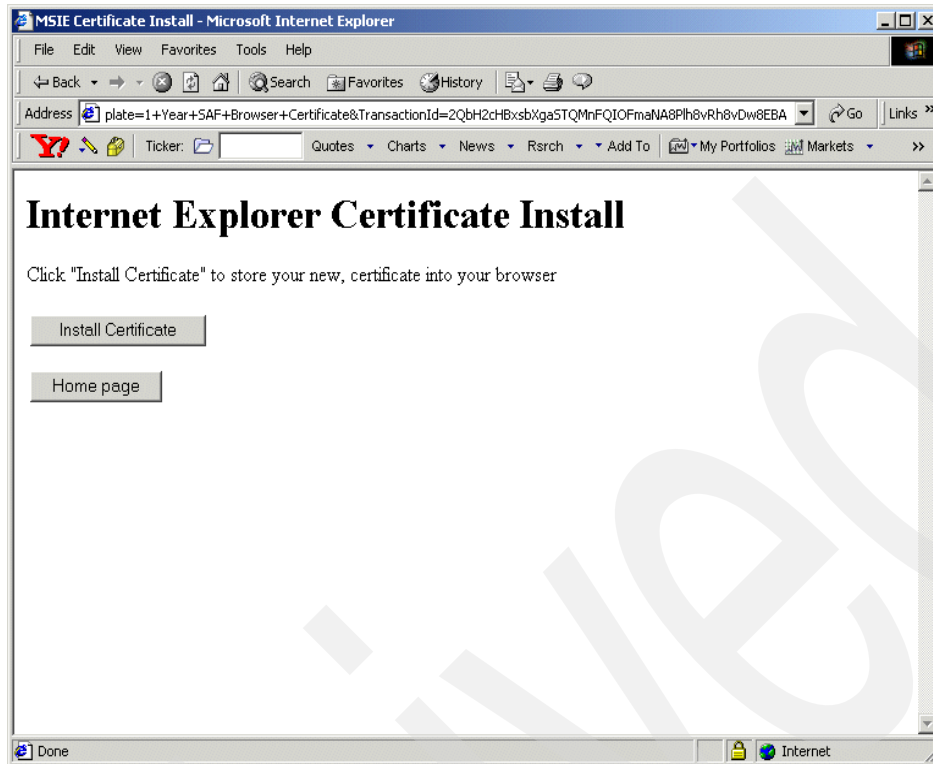


Figure 2-51 Internet Explorer certificate install page

Internet Explorer requires one more step than Netscape Navigator does. Again, click the **Continue** button to install the certificate into the browser (IE). A conformation message that the install was successful is returned, as shown in Figure 2-52.

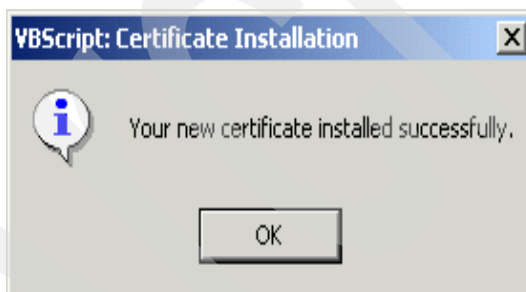


Figure 2-52 Certificate Installation successful message window

To check whether you have successfully installed your certificate, for Internet Explorer you have to take the following steps:

1. Select **Tools** from the main menu.
2. Select **Internet options** from the pull-down list.
3. Select the **Content** tab, then select **Certificates**, as shown in Figure 2-53 on page 60.

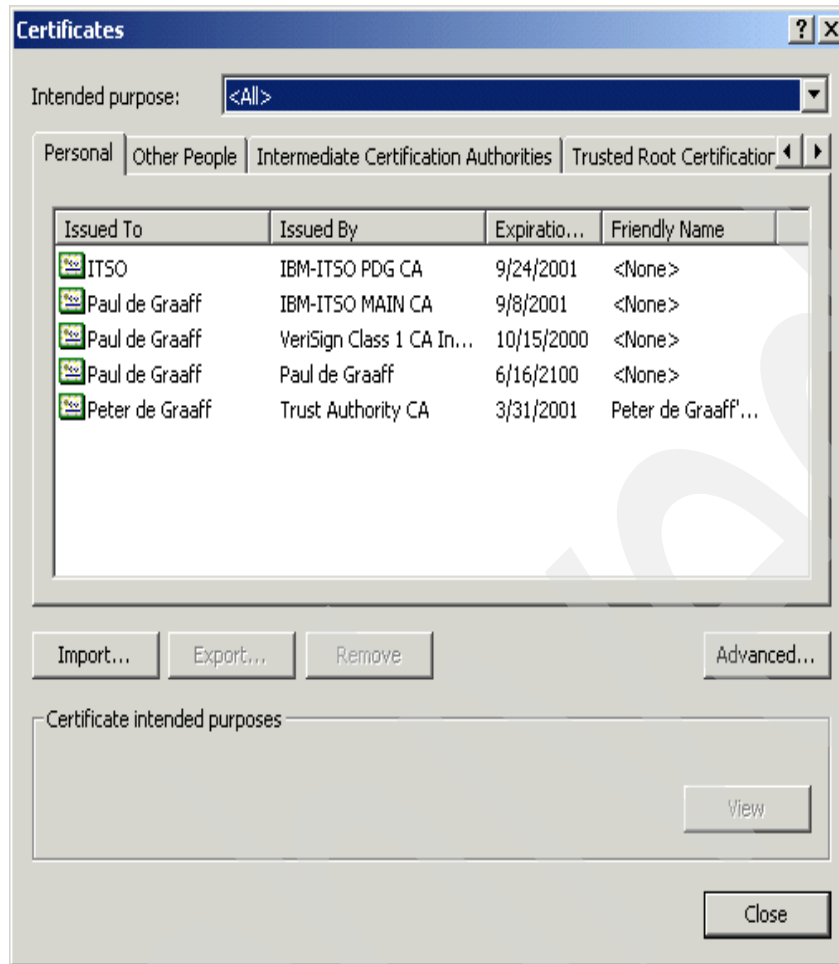


Figure 2-53 Internet Explorer Certificates window

When you select a certificate by double clicking on it, you can see the details about the certificate, as shown in Figure 2-54 on page 61.

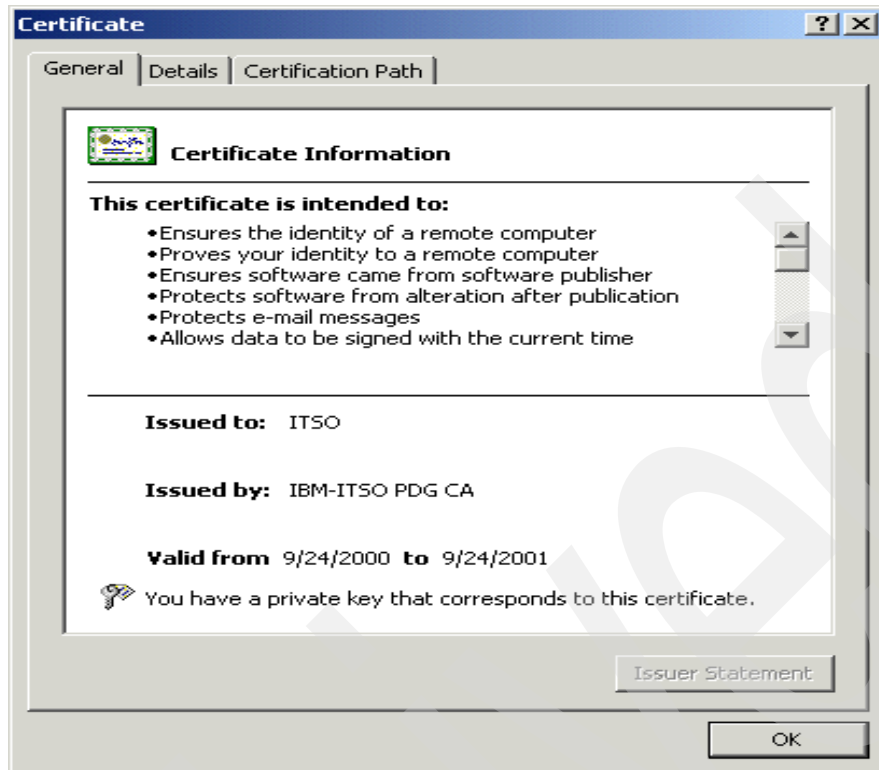


Figure 2-54 Certificate details window

You can see that the certificate was signed by the Certificate Authority (CA) IBM-ITSO PDG CA and issued to ITSO. Where did it get that information from? You did not specify any of that information when you submitted your request for a certificate. The configuration file PKI.CONF holds the key to that information. In 2.4.5, “Installation and configuration” on page 69, we discuss the actual installation of the application and the configuration of PKI.CONF to make it better suit your needs.

SAF Server Certificate 1 Year

To request a digital certificate for a server (like the IBM HTTP server) select the option **SAF Server Certificate 1 Year (auto approved)**, as shown in Figure 2-55 on page 62.

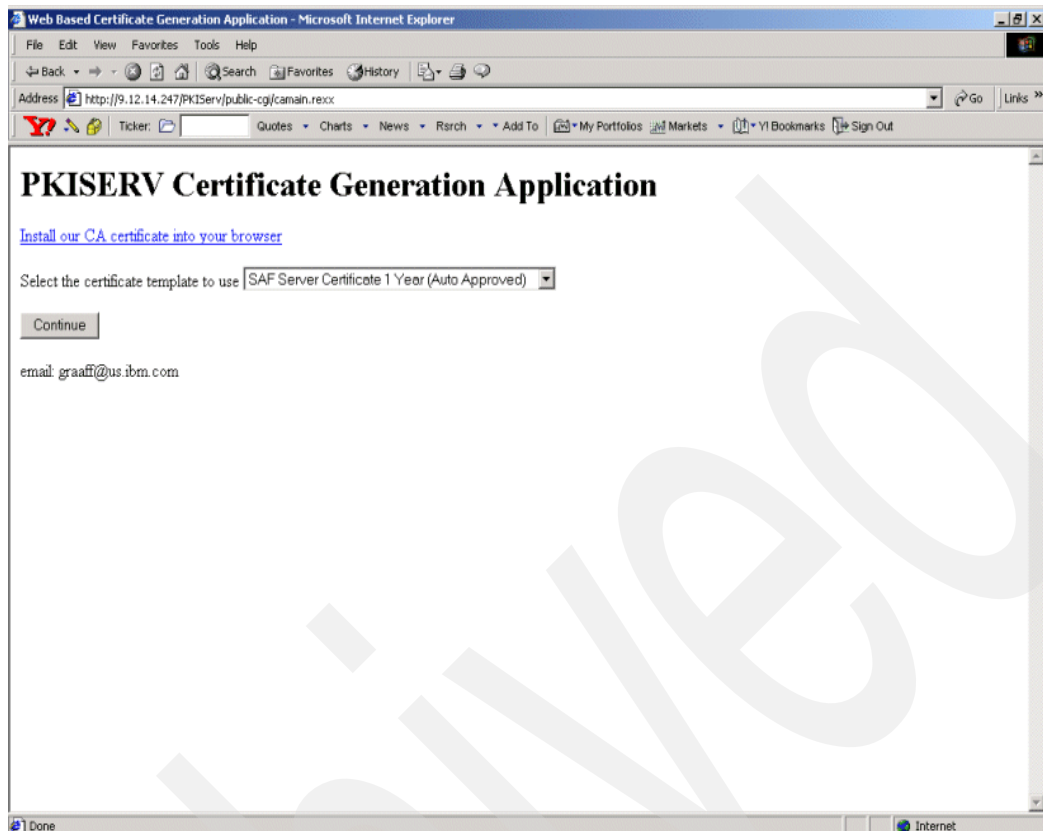


Figure 2-55 PKISERV application to request a server certificate (1)

Click **Continue** to proceed to the next dialogue in the sequence, as shown Figure 2-56 on page 63.

Web Based SAF Certificate Generation Application Pg 2 - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Search Favorites History

Address <https://jvsc57.itso.ibm.com/PKIServ/ssl-cgi-bin/catmpl.rexx?Template=1+Year+SAF+Server+Certificate> Go Links

Y Ticker Quotes Charts News Rsrch Add To My Portfolios Markets Y1 Bookmarks Sign Out

SAF Server Certificate 1 Year (Auto Approved)

Choose one of the following:

- **Request a New Certificate**

Enter values for the following field(s)

Common Name (Optional)

Organizational Unit

Organization

Country

Label assigned to certificate being requested

Base64 encoded PKCS#10 certificate request

Done Internet

Figure 2-56 PKISERV application to request a server certificate (2)

The request for a digital certificate for a server is slightly different: you first create a certificate request with the key management software of your server, like **gskkyman**. The request file is then used in the certificate request to RACF.

Important: This interface is not generally used for 390-based servers, but instead for servers running on a distributed platform and where you want RACF to become your Certificate Authority.

In our example, we created a certificate request using **gskkyman** and option 3 from the main menu, **Create new key pair and certificate request**, as shown in Figure 2-57 on page 64.

```
Key database menu

Current key database is /u/graff/racf.kdb

1 - List/Manage keys and certificates
2 - List/Manage request keys
3 - Create new key pair and certificate request
4 - Receive a certificate issued for your request
5 - Create a self-signed certificate
6 - Store a CA certificate
7 - Show the default key
8 - Import keys
9 - Export keys
10 - List all trusted CAs
11 - Store encrypted database password

0 - Exit program

Enter option number (or press ENTER to return to the parent menu):
==> 3
```

Figure 2-57 gskkyman menu to request a new key pair and certificate request

On the next screen, enter the required information about the server, as shown in Figure 2-58.

```
Enter certificate request file name or press ENTER for "certreq.arm":
racf.arm
Enter a label for this key.....> racf cert
Select desired key size from the following options (512):
1: 512
2: 1024
Enter the number corresponding to the key size you want: 2
Enter certificate subject name fields in the following.
Common Name (required).....> wtsc57.itso.ibm.com
Organization (required).....> ibm
Organization Unit (optional).....>
City/Locality (optional).....>
State/Province (optional).....>
Country Name (required 2 characters)..> US

Please wait while key pair is created...

Your request has completed successfully, exit gskkyman? (1 = yes, 0 = no):
==> 1
```

Figure 2-58 gskkyman request output for a new key pair and certificate request

The output file RACF.ARM contains the certificate request. The content of this file is the input for the PKISERV application to request a server certificate.

To use the contents of the RACF.ARM file is to use a cut and paste operation. First browse the RACF.ARM file; in our example, we used OBROWSE to display the content of the RACF.ARM file.

The certificate request information is in Base64 encoded format (PKCS#10).

You are now able to paste the certificate request into the window shown in Figure 2-56 on page 63 to request a server certificate.

Figure 2-59 shows the PKISERV application window with the paste operation completed and the other required information to generate the certificate.

Attention: We changed the default PKI.CONF file to remove the optional fields.

Enter values for the following field(s)

Common Name (Optional)
wtsc57.itso.ibm.com

Organizational Unit
itso

Organization
ibm

Country
us

Label assigned to certificate being requested
test cert2

Base64 encoded PKCS#10 certificate request

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBEDCB4gIBADASMQswCQYDVQQGEwJVUzEMMAoGA1UEChMDaWJtMRwwGgYDVQQD
ExN3dHNjNTcuaXRoby5pYm0uY29tLmIGfMAOGCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDkVY87wX0AfgXq1z3Mo4sX0k910HEHjHF22X838hXwr8zSkmy7+CulDvKeAtcj
Mx0Ah3CKg3njfifkMXpR9SVcUe40+usNQsKJX+1QgOXfjoptb1lrThvSWWuU/R+j
Bke+7ROAMuOeOP1gRe6JbZEK3OFZ/71L1Y5y+mH53+JnpQIDAQABoAAwDQYJKoZI
hvcNAQEEBQADgYEAg4xrt1c3hXTToYBsgUOxf/5mKdVIGNsHqnOmE1S8uwKcRXiQn
yIdUaaf17zwS9Mc11Y17Pd11UJ71UIITqY6F6eagwdO7HSSVuYJafP7HDfCQCjBC
S1eBB4JZR1mmHQ1v7VDA7gSxOvKBNR5tQQFV+Arb+/L69VFPNe2SoX7vMc=
-----END NEW CERTIFICATE REQUEST-----
```

Figure 2-59 PKISERV application to request a server certificate (3)

Attention: Another thing to note is that when you paste the certificate request information from the file into this window, you have pasted blanks after the **begin new certificate request** line and the **end new certificate request** line. You have to remove these blanks manually, otherwise you receive the following error:

SAF RC = 8 RACF RC = 8 RACF RSN = 60
Certificate generation provider indicated the following error: DiagInfo="PublicKey"
encoding is not valid (IRRD104I)

Click **Submit certificate request** to request your certificate. The next window appears to confirm the successful request, as shown in Figure 2-60 on page 66.

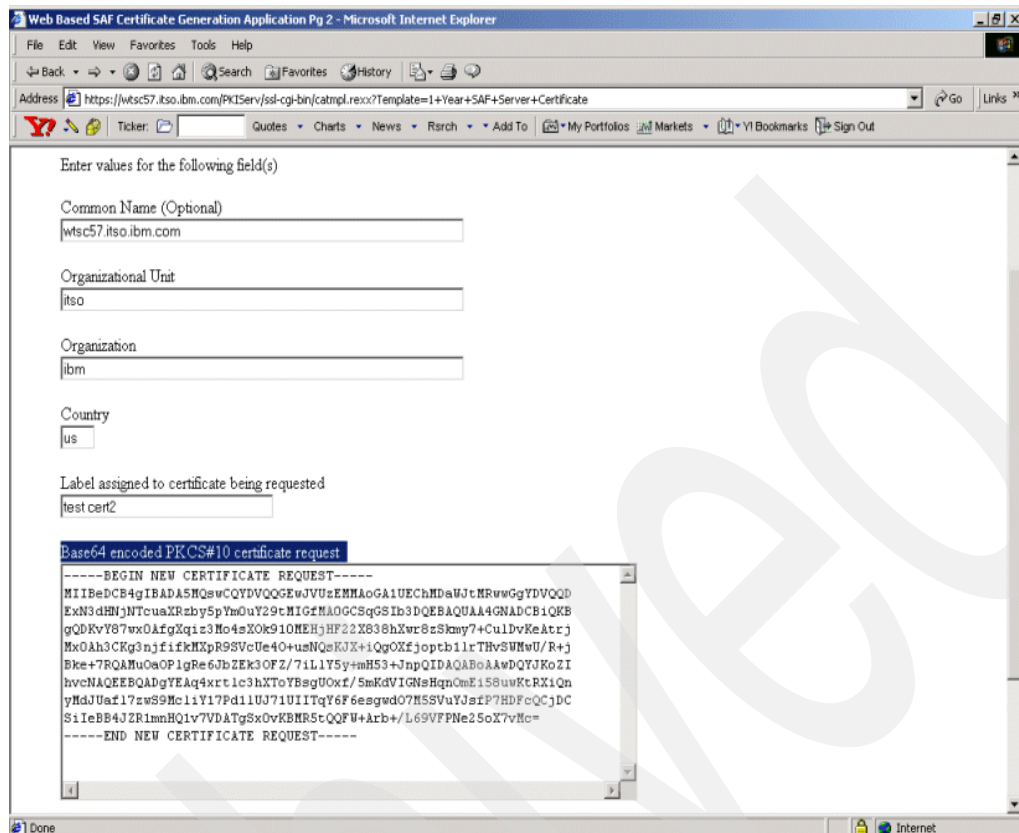


Figure 2-60 PKISERV application to request a server certificate (4)

After the successful submission, you have to retrieve the certificate from RACF. Click **Continue**; the window shown in Figure 2-61 on page 67 is displayed.

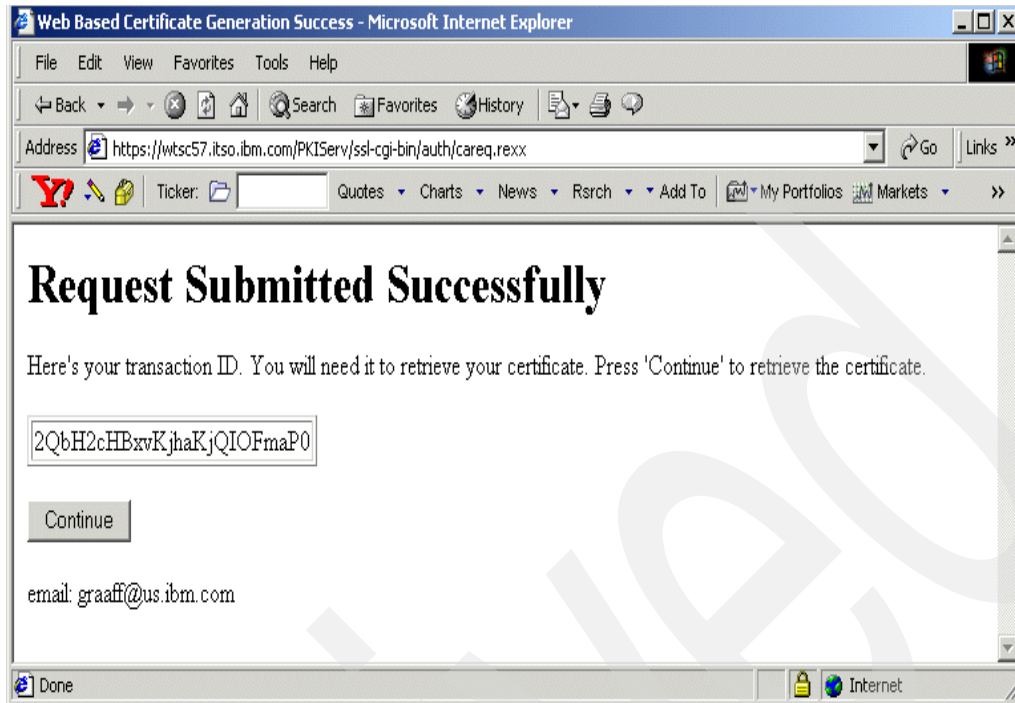


Figure 2-61 PKISERV application to request a server certificate (5)

This window allows you to retrieve the certificate based on the transaction ID (certificate ID). The transaction ID is maintained between windows, so all you have to do is click **Continue** to retrieve the certificate.

Figure 2-62 shows the final window that allows you to pick up the certificate for the server.

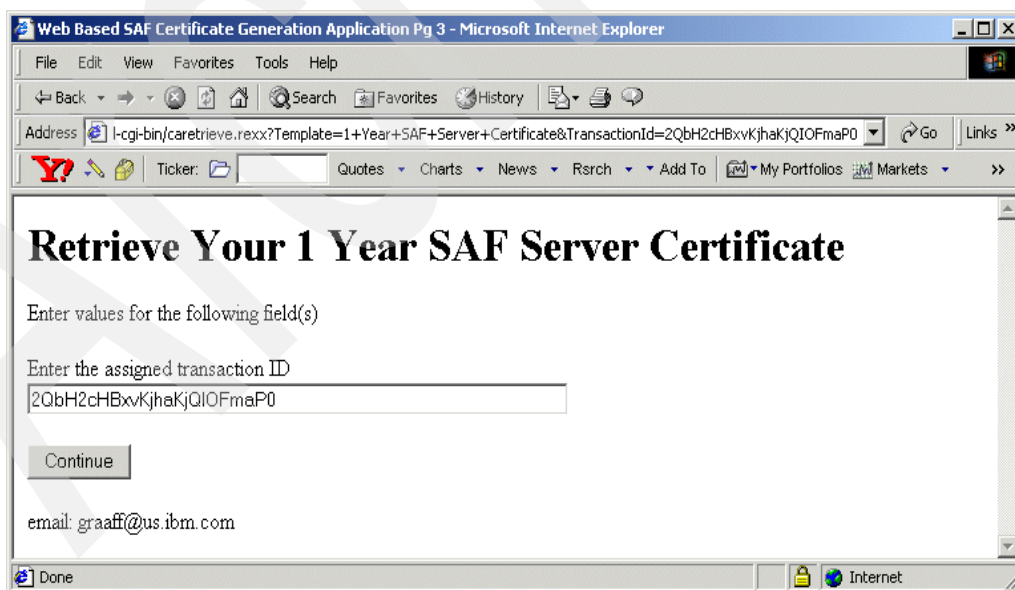


Figure 2-62 PKISERV application to request a server certificate (6)

You have to cut the certificate information from this window and paste it into a file that can be used by the key management function of your server, in our example gskkyman.

We create a file called RACF.CERT using the OEDIT function in UNIX System Services and pasted the certificate information into the file, as shown in Figure 2-63. The RACF.CERT file is now used as input to gskkyman to store the certificate into the key database for use by the server.

```

EDIT          /u/graaff/racf.cert          Columns 00001 00072
***** ***** Top of Data*****
000001 -----BEGIN  CERTIFICATE-----
000002 MIICozCCAgygAwIBAgIBFTANBgkqhkiG9w0BAQUFADBEMQswCQYDVQQGEwJVUzEM
000003 MAoGA1UEChMDSUJNMQowCwYDVQQLewRJVFNPMRgwFgYDVQQDEw9JQk0tSVRTTyBQ
000004 REcgQ0EwHhcNMDAxMDIzMDQwMDAwHcNMDEXMDIOMDM1OTU5WjBIMQswCQYDVQQG
000005 EwJ1czEMMAoGA1UEChMDaWJtMQowCwYDVQQLewRpdHNvMRwwGgYDVQQDEwN3dHNj
000006 NTcuaXRzby5pYmOuY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDKvY87
000007 wx0AfgXqiz3Mo4sX0k910MEHjHF22X838hXwr8zSkmy7+Cu1DvKeAtrjMx0Ah3CK
000008 g3njfifkMXpR9SVcUe40+usNQsKJX+iQgOXfjoptb1lrThvSWMwU/R+jBke+7RQA
000009 Mu0aOP1gRe6JbZEK30FZ/7iL1Y5y+mH53+JnpQIDAQABo4GgMIGdMEsGCVUdDwGG
000010 +EIBDQ+ExHZW51cmF0ZWQgYnkgdGhlIFN1Y3VyZVdheSBTZWN1cm10eSBTZXJ2
000011 ZXIgZm9yIE9TLzM5MCAoUkFDRikwDgYDVROPAQH/BAQDAgCGMBOGA1UdDgQWBBQc
000012 yjPYCe4cmcIqoHVVSIEGPhk+vzAfBgNVHSMEGDAWgBTOIyLpiQ8XZfRJE/LJrSN3
000013 SFONWjANBgkqhkiG9w0BAQUFAA0BgQCac/squ911K6MYpM8GaDeMpu7jTOVVjNjE
000014 TbOMwHj/G5swjzVJeAaqDaXeScvF7e6n8P9yubp3AXOZsiR3L6x361XTApbBC9wG
000015 xb3gW1JdtG060YkbXY3Dq1gmGc59We3Zgce/8ha9ziD14iW1GqIF1/WazAHT+QTY
000016 StUZYH1sXU==
000017 -----END  CERTIFICATE-----
***** ***** Bottom of Data*****

```

Figure 2-63 RACF.CERT contents

We used option 4 from the gskkyman main menu to store the certificate, as shown in Figure 2-64 on page 69.

```

Key database menu

Current key database is /u/graafl/racf.kdb

    1 - List/Manage keys and certificates
    2 - List/Manage request keys
    3 - Create new key pair and certificate request
    4 - Receive a certificate issued for your request
    5 - Create a self-signed certificate
    6 - Store a CA certificate
    7 - Show the default key
    8 - Import keys
    9 - Export keys
   10 - List all trusted CAs
   11 - Store encrypted database password

    0 - Exit program

Enter option number (or press ENTER to return to the parent menu): 4
Enter certificate file name or press ENTER for "cert.arm": racf.cert
Do you want to set the key as the default in your key database? (1 = yes, 0
= no:
-->1
Please wait while certificate is received.....
Your request has completed successfully, exit gskkyman? (1 = yes, 0 = no) :
-->1
GRAAFF @ SC57:/u/graafl>

```

Figure 2-64 gskkyman - receive certificate overview

2.4.5 Installation and configuration

Following is a high-level overview of the steps necessary to install and configure the PKIServ application. Complete details for each step are presented in the subsequent sections.

1. SMP/E Install the PTFs related to RACF APAR 45211 and SAF APAR 45212.
2. Determine and set up the RACF access control needed to protect the R_PKIServ callable service through profiles in the RACF FACILITY class.
3. Determine and set up the RACF access control needed to protect the RACDCERT services through profiles in the RACF FACILITY class.
4. Create your Certificate Authority certificate and make it available to your end users.
5. Determine the server root directory for the Web application and install the sample REXX scripts and PKI.CONF configuration file.
6. Configure the IBM HTTP Server (IHS) for OS/390 to use SSL.
7. Add the directives necessary to the HTTPD.CONF file to enable the PKISERV application.
8. Customize the sample PKI.CONF configuration file and/or REXX scripts as needed for your organization's use.

SMP/E install the PTF

Depending on your maintenance level of OS/390 Version 2 Release 10, you may or may not need to install the PTFs related to APAR 45211 and 45212.

Determine RACF access to the R_PKIServ callable service

The R_PKIServ callable service is protected by new profiles in the RACF FACILITY class called IRR.RPKISERV.<function>.

The function we are securing is called **GENCERT** and **EXPORT** like a **RACDCERT GENCERT**. The authority given to a user ID controls subsequent **RACDCERT** access checks, as follows:

- | | |
|---------------|--|
| NONE | Access to the callable service is denied.
This is also true if no profile is defined (RC4). |
| READ | Access is permitted based on subsequent RACDCERT access checks against the client's (end user's) user ID. |
| UPDATE | Access is permitted based on subsequent RACDCERT access checks against the daemon's user ID. |

Restriction: This does not apply to the Web server daemon. This is intended for future use.

- | | |
|----------------|--|
| CONTROL | Access is permitted with no subsequent RACDCERT access checks made.
If you user ID is RACF SPECIAL then no subsequent checks are made as well. |
|----------------|--|

The subsequent access control checks are made against the RACF FACILITY class profile IRR.DIGTCERT.<function>, as discussed in the next section.

The following authorizations are needed when you are not using a surrogate user ID for digital certificate processing:

```
PERMIT IRR.RPKISERV.GENCERT CLASS(FACILITY) ID(userid) ACC(READ)
PERMIT IRR.RPKISERV.EXPORT CLASS(FACILITY) ID(userid) ACC(READ)
```

Note: *userid* is any RACF user ID that is allowed to generate a certificate. The export function allows the user to retrieve the certificate.

If you have chosen to use a surrogate user ID as suggested, then you need the following RACF authorizations:

```
PERMIT IRR.RPKISERV.GENCERT CLASS(FACILITY) ID(pkiserv) ACC(READ)
PERMIT IRR.RPKISERV.EXPORT CLASS(FACILITY) ID(userid) ACC(READ)
```

Note:

pkiserv is our surrogate user ID.

userid is any RACF user ID that is allowed to generate a certificate. The export function allows the user to retrieve the certificate.

Determine RACF access to the RACDCERT services

To be able to generate a digital certificate with the RACF **RACDCERT** command, you need RACF access to the following profiles in the RACF FACILITY class:

1. IRR.DIGTCERT.ADD
2. IRR.DIGTCERT.GENCERT

The PKISERV application uses a Certificate Authority certificate to sign all digital certificates indicated by the %%SignWith%% field in the configuration file PKI.CONF. The CA certificate is likely to be owned by a user ID that is not used for signon purposes. So, to allow somebody to sign his or her certificate with that CA certificate, the following authorities are needed for the PKISERV application to work:

1. READ access to the IRR.DIGTCERT.ADD profile in the RACF FACILITY class
2. CONTROL access to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class

As an alternative, you can choose to use a surrogate user ID, like PKISERV, to be authorized to this service. Do this, for instance, if you do not want everybody to have CONTROL access to the IRR.DIGTCERT.GENCERT FACILITY profile. User ID PKISERV requires the following authorities:

1. UPDATE access to the IRR.DIGTCERT.ADD profile in the RACF FACILITY class
2. CONTROL access to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class

Depending on your security requirements and policy, access to these profiles can be allowed to all user IDs defined on the system.

Important: A RACF user ID with the SPECIAL attribute is exempt from any access control checks to these profiles.

Example 2-11 shows the authorizations needed for the HTTP server when using a surrogate user ID like PKISERV.

Example 2-11 Sample definitions for PKISERV surrogate processing

```
RDEFINE SURROGAT BPX.SRV.PKISERV UACC(NONE)
PERMIT BPX.SRV.PKISERV CLASS(SURROGAT) ID(websrv) ACC(READ)
SETOPTS RACLIST(SURROGAT) REFRESH
```

Example 2-12 shows our definition of our *surrogate* user ID PKISERV.

Example 2-12 PKISERV definition example

```
addgroup pki supgroup(sys1) owner(sys1)

adduser pkiserv dfltgrp(pki) owner(pki) name('PKISERV RACF APPL')

lu pkiserv

USER=PKISERV NAME=PKISERV RACF APPL OWNER=PKI CREATED=00.270
DEFAULT-GROUP=PKI PASSDATE=00.000 PASS-INTERVAL=180
ATTRIBUTES=NONE
REVOKE DATE=NONE RESUME DATE=NONE
LAST-ACCESS=00.273/12:05:55
CLASS AUTHORIZATIONS=NONE
NO-INSTALLATION-DATA
NO-MODEL-NAME
LOGON ALLOWED (DAYS) (TIME)
-----
ANYDAY ANYTIME
GROUP=PKI AUTH=USE CONNECT-OWNER=PKI CONNECT-DATE=00.273
CONNECTS= 01 UACC=NONE LAST-CONNECT=00.273/12:05:55
CONNECT ATTRIBUTES=NONE
REVOKE DATE=NONE RESUME DATE=NONE
SECURITY-LEVEL=NONE SPECIFIED
CATEGORY-AUTHORIZATION
NONE SPECIFIED
SECURITY-LABEL=NONE SPECIFIED
```

Create your Certificate Authority certificate

In order to create and sign digital certificates for others you need to define a Certificate Authority certificate and associated private key. This is done using the RACF **RACDCERT GENCERT** command. Before issuing the command, decide what the Certificate Authority's distinguished name will be. Typically, Certificate Authorities have distinguished names in the following form:

OU=<your-CA's-friendly-name>.O=<your-organization>.C=<your-2-letter-country-abbreviation>

The **RACDCERT GENCERT** command to create our CERTAUTH certificate for use in our examples is shown in Example 2-13.

Example 2-13 RACDCERT GENCERT command to generate CA certificate

```
RACDCERT GENCERT CERTAUTH SUBJECTSDN(DN('IBM-ITSO PDG CA') OU('ITSO') O('IBM') C('US'))
NOTBEFORE(DATE(2000-09-25)) NOTAFTER(DATE(2005-09-24))
WITHLABEL('ITSO PDG CA'))
```

Important: Set the validity date of the CA certificate to more years than what you will use for your personal certificates to avoid date inconsistencies.

You now need to export the CA certificate and publish it to your root directory, so it is available for download. Use the following RACF command to do this:

```
RACDCERT CERTAUTH EXPORT(LABEL('ITSO PDG CA')) FORMAT(CERTDER) DSN('GRAAFF.EXPORT.CADER')
```

Next, copy it from the dataset created with the **RACDCERT** command to the HFS root directory you choose for the PKISERV application, like this:

```
OPUT 'graaff.export.cader' '/usr/lpp/internet/etc/cacert.der' BINARY
```

For more information on the **RACDCERT** command, defining profiles, and granting access, see the *SecureWay Security Server for OS/390 (RACF) Security Administrator's Guide*, SC28-1915 and *SecureWay Security Server for OS/390 (RACF) Command Language Reference*, SC28-1919.

Installing the sample code

We assume the IBM HTTP Server is installed and functional for at least normal non-SSL mode before going on to this step. The default location for the server root is:

```
/usr/lpp/internet/server_root
```

Depending where you want to install the PKISERV application, the root might be:

```
/usr/lpp/internet/server_root/PKIServ
```

Create the directory structure for PKIServ, as defined in 2.4.2, "Directory structure provided" on page 50.

Obtain the sample tar file from the RACF Web site:

```
http://www.s390.ibm.com/products/racf/webca.html
```

Next, do a file transfer in binary mode using FTP to OS/390. Then untar the file using the following command:

```
cd /usr/lpp/internet/server_root
```

```
tar -xvf PKISERV.tar
```

Remember to set the permission bits on all files. The permission bits for the executables can be the octal value of “644.” Permissions bits can be changed using the **chmod** command, as shown:

```
CHMOD 644 CAMAIN.REXX
```

The permissions bits for the configuration file PKI.CONF and the Certificate Authority certificate CACERT.DER can be set according to your installation’s security requirements.

Configure the IBM HTTP Server for SSL mode

The PKISERV application requires that the IBM HTTP Server operate in both normal and SSL mode. To be able to use SSL, your server will need to obtain a digital certificate. You can choose to purchase one from an external Certificate Authority (for example, Verisign) or create one using RACF.

Attention: If your server is already operating in SSL mode you can skip “Obtaining a certificate for your Web server” on page 73.

What utility you use to create a server certificate with is determined by your level of IBM HTTP Server for OS/390. Prior to IHS Release 5.3, you use **ikeyman**; with IHS Release 5.3 you can use **gskkyman** or a RACF keyring using the **RACDCERT ADDRING** command.

Obtaining a certificate for your Web server

The procedure we describe next applies to the IBM HTTP Server (IHS) Release 5.3 that ships with OS/390 Version 2 Release 10. To be able to use SSL, your server needs a digital certificate to identify the server, like the one we created earlier for you (an individual). IHS allows you to choose between two options:

1. Create a digital certificate using a utility called **gskkyman**
2. Create a digital certificate using RACF, using the **RACDCERT GENCERT** command

Next, the digital certificate of your server needs to be installed in a “key ring,” which is used by your server. The **gskkyman** utility creates a key database or key ring for you. The RACF **RACDCERT ADDRING** and **RACDCERT CONNECT** commands can do similar functions. The difference between the two key rings is the location of the key ring and the security of the key ring. The key ring created by **gskkyman** is a UNIX file and the security relies on permission bits and a password to access the file. The RACF key ring, as the name implies, is stored in the RACF database, and requires RACF authorizations to be read and updated.

You need to decide who (what Certificate Authority certificate) is signing the server’s certificate, as discussed later in this section. You can use the same CA certificate that signs the browser’s (user) digital certificates.

Using gskkyman to obtain a server certificate

1. Determine the location for your key database (key ring) and change to that directory, for example **CD** to **/usr/lpp/internet/etc** and invoke **gskkyman**.
2. Choose option **1** to create a key database. Type in a name or let it default to **key.kdb** and enter the desired password. When asked to “Work with the database now?” enter **1** for yes.
3. Choose **3 - Create new key pair and certificate request**. Answer the prompts for file name, label, key size (1024 recommended), and subject name fields. Exit **gskkyman** when done.

Note: Common Name should be your server’s symbolic IP address (for example, **www.ibm.com**).

4. From TSO, **0GET** the certificate request file to an OS/390 data set, for example:

```
0GET certreq.arm '/usr/lpp/internet/etc/certreq.arm'
```
5. Use the RACF **RACDCERT** command to read the request and generate the server certificate, for example:

```
RACDCERT GENCERT(certreq.arm) ID(websrv) SIGNWITH(CERTAUTH LABEL('ITSO PDG CA'))  
WITHLABEL('SSL Cert')
```
6. Export the new server certificate to a data set, then do an **0PUT** to the HFS file, for example:

```
RACDCERT EXPORT(LABEL('SSL Cert')) ID(websrv) DSN(cert.arm) FORMAT(CERTB64)  
  
0PUT cert.arm '/usr/lpp/internet/etc/cert.arm'
```
7. **CD** to /usr/lpp/internet/etc and invoke **gskkyman** again.
8. Choose option **2** to open the key database you created before. Reply to the name and password prompts.
9. Choose option **6** to store a CA certificate and specify the /usr/lpp/internet/etc/cacert.der file. When prompted to “Exit gskkyman?” enter **0** for No.
10. Choose option **4** to receive a certificate issued for your request and specify the /usr/lpp/internet/etc/cert.arm file. Again enter **0** when asked “Exit ikeyman?”
11. Choose option **11** to store the encrypted database password.

Using RACF to obtain a server certificate

The following steps detail the setup of a RACF key ring and a server certificate for your Web server. This requires Release 5.3 of the IBM HTTP Server (IHS) for OS/390 and APAR PQ39311.

1. The user ID associated with the IHS started task needs to be authorized to the following RACF FACILITY class profiles:

```
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(websrv) ACCESS(CONTROL)  
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ID(websrv) ACCESS(CONTROL)
```

2. Set up the RACF key ring for use by the Web server with the following command:

```
RACDCERT ID(websrv) ADDRING(WEB57)
```

Note: You need RACF SPECIAL to be able to define the RACF key ring.

3. Generate a server certificate for your HTTP server using the RACF **RACDCERT** command, as follows:

```
RACDCERT ID(websrv) GENCERT SUBJECTSDN(CN('WTSC57.ITS0.IBM.COM') O('IBM') OU('ITS0')  
L('Poughkeepsie') C('US') SP('New York')) SIZE(1024) WITHLABEL('WEB57CERT')  
SIGNWITH(CERTAUTH LABEL('ITSO PDG CA'))
```

Note: *websrv* is the user ID associated with the Started Task running our HTTP Server. This is a hard requirement.

4. Connect the server certificate and the Certificate Authority certificate to the RACF key ring created in step 2 using the RACF **RACDCERT CONNECT** command, as follows:

```
RACDCERT ID(websrv) CONNECT(ID(websrv) LABEL('WEB57CERT') RING(WEB57) DEFAULT  
USAGE(PERSONAL)  
RACDCERT ID(websrv) CONNECT(CERTAUTH LABEL('ITSO PDG CA') RING(WEB57) USAGE(CERTAUTH))
```

5. To configure your HTTP server to use the RACF key ring, update the HTTPD.CONF file with the following KEYFILE directive:

keyfile "WEB57" SAF

Turning on SSL mode

To turn on SSL in your HTTP server, make the following changes in your configuration file HTTPD.CONF:

1. Set SSLMODE ON
2. Designate a port for SSL usage, like SSLPORT 443
3. Set the parameter KEYFILE to indicate either the UNIX file name for the key ring to be used (such as KEYFILE /usr/lpp/internet/etc/key.kdb) or indicate the label name of the RACF keyring to be used.

Note: Step 3 is just a reminder. If you followed the procedure outline in "Using RACF to obtain a server certificate" on page 74 you already made that change.

Other changes to make to the HTTPD.CONF file

Figure 2-65 on page 76 shows the directives that are added to the HTTPD.CONF file to be able to use the PKISERV application. The setup requires three protection setups:

- | | |
|------------------------------|--|
| PublicUser (1) | Used for the "public" resources |
| AuthenticatedUser (2) | Used to determine the client requesting the certificate |
| SurrogatUser (3) | Used for the actual generation of the certificate because we do not want every user to be authorized with CONTROL authority to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class |

Tip: If you choose not to use a surrogate user ID to issue certificates, then replace *SurrogatUser* with the *AuthenticatedUser*.

The **PROTECT**, **PASS** and **EXEC** directives are there to insure the proper protection and execution of the PKISERV application.

There is one **REDIRECT** directive to insure that SSL is invoked for the PKISERV application.

```

Protection PublicUser {
    ServerId      PublicUser
    UserID        PUBLIC
    Mask          Anyone
}

Protect /PKIServ/public-cgi/*      PublicUser
Protect /PKIServ/ssl-cgi-bin/*     PublicUser
Protect /PKIServ/*                 PublicUser

Protection AuthenticatedUser {
    ServerId      AuthenticatedUser
    AuthType      Basic
    PasswdFile    %%SAF%%
    UserID        %%CLIENT%%
    Mask          All
}

Protection SurrogateUser {
    ServerId      SurrogateUser
    AuthType      Basic
    PasswdFile    %%SAF%%
    UserID        PKISERV
    Mask          All
}

Protect /PKIServ/ssl-cgi-bin/auth/*      AuthenticatedUser
Protect /PKIServ/ssl-cgi-bin/auth/careq.rexx SurrogateUser

Redirect /PKIServ/ssl-cgi/*      https://<server-domain-name>/PKIServ/ssl-cgi-bin/*
Exec /PKIServ/public-cgi/*      <server-root>/PKIServ/public-cgi/*
Exec /PKIServ/ssl-cgi/*         <server-root>/PKIServ/ssl-cgi-bin/*
Exec /PKIServ/ssl-cgi-bin/*     <server-root>/PKIServ/ssl-cgi-bin/*
Pass /PKIServ/*                 <server-root>/PKIServ/*

AddType .cer application/x-x509-user-cert ebcidc 0.5 # Browser Certificate
AddType .der application/x-x509-ca-cert binary 1.0 # CA Certificate

```

Figure 2-65 HTTPD.CONF file required changes

Customizing the PKI.CONF file

Depending on your installation's needs and security requirements, you may decide to change your configuration file PKI.CONF. One change that is definitely required is to indicate the label of the Certificate Authority certificate that is going to sign the certificates requested.

In the <constant> section of each <template>, the %%SignWith=%% parameter indicates the label of the CERTAUTH certificate that signs the certificate issued with the PKISERV application. In our example we use the CERTAUTH certificate generated earlier, like %%SignWith=SAF:CERTAUTH/ITSO PDG CA%%.

Another area where you may want to change things is the information necessary to build the browser's certificate. The only two fields displayed today are the label and public key type (CSP) on the certificate request page, as shown in Figure 2-46 on page 54. The following fields can be added to the request page to provide more meaningful information:

CommonName Name of the individual, like Paul de Graaff

Important: If you leave the %%CommonName%% field in the constant section, it uses the NAME field from your RACF User profile as the Common Name.

OrgUnit Organizational unit the individual belongs to, like ITSO

Org Organization the individual works for, like IBM

Locality Locality, like Poughkeepsie

StateProv State or Province you live in, like New York

Country Country you live in, abbreviated to two characters, like US

AltEmail Your E-mail address, like graaff@us.ibm.com

AltDomain Your internet domain, like www.ibm.com

AltURI Your universal resource identifier, like wtsc57.itso.ibm.com

AltIPAddr Numeric IP address, like 9.12.14.247

Depending on your security requirements, you can have the user enter his or her CommonName, Org, OrgUnit and Locality. To make these changes, you need to edit the PKI.CONF file.

Figure 2-66 shows part of the PKI.CONF file before we made the changes to include the new fields on the certificate request page.

```
TEMPLATE NAME=1 Year SAF Browser Certificate>
.....
<p> Enter values for the following field(s)
%%Label%%
%%PublicKeyYbrowsertype""%%
.....
<CONSTANT>
%%Org=IBM%%
%%OrgUnit=ITSO%%
%%KeyUsage=handshake%%
%%NotAfter=365%%
%%Country=US%%
%%SignWith=SAF:CERTAUTH/ITSO PDG CA%%
</CONSTANT>
```

Figure 2-66 PKI.CONF file before changes

To make the changes, move the fields you want to include in the certificate request page from the <constant> section to the section under “Enter values for the following Field(s)” and remove the value specified, as shown in Figure 2-67.

```

<TEMPLATE NAME=1 Year SAF Browser Certificate>
.....
<p> Enter values for the following field(s)
%%Label%%
%%CommonName%%
%%Org%%
%%OrgUnit%%
%%PublicKeyYbrowserType%%
<CONSTANT>
%%KeyUsage=handshake%%
%%NotAfter=365%%
%%Country=US%%
%%SignWith=SAF:CERTAUTH/ITSO PDG CA%%
</CONSTANT>

```

Figure 2-67 PKI.CONF file after changes

When you have made your changes, save them and invoke the PKISERV application again. When you make the request for a browser certificate, you will see the screen shown in Figure, 2-68, including the new fields added.

Figure 2-68 Certificate request page after PKI.CONF updates

2.5 OS/390 UNIX Superuser granularity support

This enhanced support verifies that an OS/390 UNIX user has the necessary privilege to perform a **chmount**. OS/390 UNIX checks the authority information by calling the check privilege (**ck_priv**) callable service (IRRSKP00), which determines if the user requesting the **chmount** has superuser privileges, or the authority to the appropriate resource in the UNIXPRIV class.

Important: This support is available on OS/390 Version 2 Release 9 with APAR OW39896 applied.

The **ck_priv** callable service (IRRSKP00) checks authority to the SUPERUSER.FILESYS.MOUNT resource in the UNIXPRIV class if the user does not have superuser authority. UPDATE authority is required for **chmount** when **setuid** is specified; READ authority is required when **setuid** is not specified.

2.6 Service Updates

This section describes changes from authorized program analysis reports (APARs) or other service updates that have been incorporated into this release.

2.6.1 APAR OW39128

Customers need to audit the use of programs. When they use a program class profile that includes more than one library in the member list, the auditing is not effective since the data set name is not included in the SMF record. Without the data set name the customer cannot determine which library the program was loaded from.

The problem resolution is that if a successful or failed attempt to load a controlled program is audited, RACF builds SMF record type 80, EVENT 2 documenting the RESOURCE ACCESS attempt. The program name is contained in Relocate Section 1 (x'01'), which is included in the SMF record. To further identify the controlled program, the library (the partitioned data set) and the volser are added to the record. The partitioned data set name is contained in a new relocate section, 66 (x'42'), and the volume serial is contained in an existing relocate section, 15 (X'0F') that had not previously been included in the EVENT 2 record for this case. Changes are also made to the SMF UNLOAD utility and SAMPLIB members IRRADUTB and IRRADULD to allow the viewing of this new information. The additional information makes program control much easier to audit.

2.6.2 APAR OW38799

The SMF records written by the **SETOPTS** command processor have no indication that the **LIST** operand was requested. The output in the SMF records is the same for **SETOPTS** (with no operands) as it is for **SETOPTS LIST**. RACF should indicate the **LIST** operand was specified on the **SETOPTS** command.

The problem resolution is that the SMF Type 80 record has been modified to include **LIST** among the options specified or ignored in the **SETOPTS** command. **LIST**-specified is indicated by a bit positioned immediately after the **NOADDCREATOR**-specified bit. Likewise, **LIST**-specified-but-ignored is indicated by a bit positioned immediately after the **NOADDCREATOR**-specified-but-ignored bit. Also, the SMF Data Unload Utility (IRRADU00) was modified such that if **LIST** had been specified in the **SETOPTS** command, it will also be included in the list of options in its corresponding formatted output record.

2.6.3 APAR OW42092

In some circumstances the `ls -l` and `whoami` commands may return UID numbers instead of UID names. Specifically, both of these functions rely on a `getpwuid()` call to provide a name and group name for the associated UID. Currently, `getpwuid` will translate the UID number to a UID name and then use that name to determine a group name. If the group name found does not have an associated GID number, then `getpwuid` will return with a bad return code, causing the UID number not to be translated in the output of the command. The intention of this APAR is to change the `getpwuid` function to return the group name from BPX.DEFAULT.USER in this circumstance. This makes the function of `getpwuid` consistent with `getpwnam`. For example, a user that did a `chown` user name filename will not be failed if the default group of user name has no GID (provided BPX.DEFAULT.USER has a group with a valid GID) yet the corresponding `ls -l` command will show the owner of the file as the UID number of the user name instead of the user name. This APAR would cause `ls -l` to show a user name.

The problem resolution is that RACF will always audit when a default-UID-user gets dubbed. This facilitates an installation's ability to monitor usage of default UID, and to prevent unintended use of the default UID. Also, RACF will issue a warning message if a user is assigned an OMVS UID, but the user's default group does not have an OMVS GID.

Two new RACF warning messages have been documented for an **ADDUSER** and **ALTUSER** command:

ICH01019I This warning message is issued if a user with an OMVS UID gets added and has a default group which does not have a GID. The message is as follows:

```
ICH01019I USER UGO IS ASSIGNED AN OMVS UID,  
BUT DEFAULT GROUP CICSJB DOES NOT HAVE A GID.PROCESSING CONTINUES.
```

ICH21035I This warning message is issued if a user with an OMVS UID gets changed and has a default group which does not have a GID. The message is as follows:

```
ICH21035I USER UGO IS ASSIGNED AN OMVS UID, BUT DEFAULT GROUP  
CICSJB DOES NOT HAVE A GID. PROCESSING CONTINUES.
```

To fix either of these problems, the default group should be assigned a GID, or the UID should be removed from the user profile.

2.7 RVARY command enhancement

This section describes the enhancement made to the password processing of the **RVARY** command.

Important: This enhancement depends greatly on physical security on the Master Console, we recommend that the Master Console is always protected from unauthorized access. Only Operational personnel should have access to the Master Console, which should reside in a secure controlled area.

2.7.1 Introduction

To improve the availability of RACF, the default password *YES* is accepted by the **RVARY** command in certain scenarios. This is in addition to any password set by the installation with **SETRPTS RVARYPW**. This is limited to the **RVARY** command functions normally required in recovery situations, namely:

- ▶ **RVARY NODATASHARE**

- ▶ **RVARY SWITCH**
- ▶ **RVARY ACTIVE**

If one of these **RVARY** command functions is requested, either from someone who has access to the **RVARY** command or from an operator who has physical access to the master console, an operator reply of *YES* or the installation-defined password will be accepted, but only from the master console. This allows faster recovery from RACF database errors, DASD failures, and so forth. The *YES* password may be used in these critical situations:

- ▶ The person who knows the **RVARY** command password is unavailable.
- ▶ The **RVARY** command password has been forgotten and no administrator is available to reset it with the **SETROPTS RVARYPW** command, or the system is not in a state that allows a reset.
- ▶ The system has unknowingly been IPLed with the wrong RACF databases, which have different passwords than expected.
- ▶ An error is encountered in encrypting the password and comparing it to the installation-defined password.

2.7.2 Logical console security

During a planned outage for a RACF database change, it is possible for the RACF administrator to amend their own user ID or the driver of the change user ID so they have **OPER** and **CONSOLE** authority under the class **TSOAUTH**, along with amendments to the **OPERPARM** segment within the user ID.

Important: While this makes the change easier to implement, it is not recommended that any user ID has this authority on a permanent basis. As we stated earlier in this discussion of the **RVARY** command, we recommend that physical security is in place.

With a few RACF commands you can add an **OPERPARM** segment to the user ID and permit access to the **OPER** and **CONSOLE** resources in the **TSOAUTH** class.

1. Give the user ID read access to the profiles **OPER** and **CONSOLE** within the class **TSOAUTH**, as follows:

```
PERMIT OPER CLASS(TSOAUTH) ID(BRIGGS) ACCESS(READ)
PERMIT CONSOLE CLASS(TSOAUTH) ID(BRIGGS) ACCESS(READ)
```

2. To refresh the **RACLIST**ed class **TSOAUTH** for the changes made in step 1, use the **SETROPTS RACLIST REFRESH** command:

```
SETR RACLIST(TSOAUTH) REFRESH
```

3. Add an **OPERPARM** segment to the user ID using the **ALTUSER** command:

```
ALU BRIGGS OPERPARM(AUTH(MASTER) LEVEL(ALL) CMDSYS(*) MSCOPE(*ALL)
ALTGRP(MASTER))
```

This now enables the user ID to enter the **OPER** and **CONSOLE** commands via their own workstation. This authority, while useful for the individual user, potentially opens up various security exposures to your environment.

2.7.3 RVARY console samples

Now that you have a user ID with an **OPERPARMS** segment correctly defined, you can issue the **CONSOLE** and **OPER** commands via your workstation. This section shows the main uses of the **RVARY** command using the password *YES*, with the locally defined RACF subsystem '#'.

1. By entering the **CONSOLE** command from any ISPF screen, you obtain Master Console authority. Use the **RVARY LIST** command to check the status of the RACF databases, as shown in Figure 2-69.

```

CONSOLE
#rvary list
CONSOLE

IRRA011I (#) OUTPUT FROM RVARY:
ICH15013I RACF DATABASE STATUS:
ACTIVE  USE    NUMBER  VOLUME    DATASET
-----  ---    -
YES     PRIM    1      PDGCAT    SYS1.RACFESA
YES     BACK    1      PDGSY1    SYS1.RACF.BKUP1
ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.
CONSOLE

```

Figure 2-69 RVARY LIST command and output

2. Issue the **RVARY SWITCH** command to switch from the RACF Primary database to the RACF Backup database. After entering the **RVARY SWITCH** command, you need to determine the outstanding WTOs reply, to find out the correct WTO associated with the **RVARY SWITCH** before you can reply to the **SWITCH** command, as shown in Figure 2-70.

```

#rvary switch
CONSOLE
d r,1
CONSOLE

IEE112I 14.01.21 PENDING REQUESTS 558
RM=1  IM=0  CEM=0  EM=0  RU=0  IR=0  NOAMRF
ID:R/K  T MESSAGE TEXT
008 R *008 ICH703A ENTER PASSWORD TO SWITCH RACF DATASETS
        JOB=RACF  USER=BRIGGS

CONSOLE
r 08,yes

```

Figure 2-70 RVARY SWITCH process

3. To activate the RACF Primary database, use the **RVARY ACTIVE** command. Like in step 2, find the WTO and reply to the outstanding WTO number, as shown in Figure 2-71 on page 83.


```

#rvary active,dataset(sys1.racfesa)
CONSOLE
d r,1
CONSOLE

IEE112I 14.03.51 PENDING REQUESTS 661
RM=1    IM=0    CEM=0    EM=0    RU=0    IR=0    NOAMRF
ID:R/K   T MESSAGE TEXT
          009 R *009 ICH702A ENTER PASSWORD TO ACTIVATE RACF JOB=RACF
          USER=BRIGGS

CONSOLE
r 09,yes
CONSOLE
IEE600I REPLY TO 009 IS;SUPPRESSED

IRRA011I (#) OUTPUT FROM RVARY:
ICH15013I RACF DATABASE STATUS:
ACTIVE   USE    NUMBER  VOLUME   DATASET
-----
YES      PRIM    1      PDGSY1   SYS1.RACF.BKUP1
YES      BACK    1      PDGCAT   SYS1.RACFESA
ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.

```

Figure 2-71 RVARY ACTIVE command

Restriction: The **RVARY** command should not be protected using RACF program control. During recovery, RACF must not be allowed to attempt to access the database. The **RVARY** command is always protected by an operator prompt, regardless of whether it is entered from TSO or as an operator command.

Archived

Network Authentication and Privacy Service

A new component has been added to the SecureWay Security Server for OS/390. Called Network Authentication and Privacy Service (NAPS), it is the IBM OS/390 implementation of Kerberos Version 5 from the Massachusetts Institute of Technology (MIT).

This chapter describes the installation and implementation of the Network Authentication and Privacy Services on OS/390.

It also describe the use of Kerberos (NAPS) by IBM's DB2 Version 7 product on OS/390 and DB2 Connect Version 7.1 running on Windows 2000, as well as how to configure these components to use Kerberos technology.

The Network Authentication and Privacy Service (NAPS) support is rolled back to OS/390 Version 2 Release 8. The examples in this chapter use OS/390 Version 2 Release 10.

3.1 Introduction to Kerberos

Kerberos is a network authentication protocol that was developed in the 1980s by Massachusetts Institute of Technology, in cooperation with IBM and Digital Equipment Corporation. Data Encryption Standard (DES) cryptography is used to provide data privacy, especially for the sensitive data such as password to log into a server.

Kerberos Version 5 is the latest release. It has been implemented in SecureWay Security Server Network Authentication and Privacy Service for OS/390, and chosen by Microsoft Corporation as their preferred authentication technology in Windows 2000 and by SUN for Solaris 8.

Important: IBM OS/390 implementation of Kerberos Version 4 provided by Communications Server for OS/390 IP is no longer supported in z/OS V1R2.

Kerberos is an encryption-based security system that provides mutual authentication between the users and the servers in a network environment. The assumed goals for this system are:

- ▶ Authentication to prevent fraudulent requests/responses between users and servers that must be confidential and on groups of at least one user and one service.
- ▶ Authorization can be implemented independently from the authentication by each service that wants to provide its own authorization system. The authorization system can assume that the authentication of a user/client is reliable.
- ▶ Message confidentiality may also be used that provides assurance to a data sender that the message's content is protected from access by entities other than the context's named peer.

The Kerberos authentication is heavily based on shared secrets, which are passwords stored on the Kerberos server. Those passwords are encrypted with a symmetrical cryptographic algorithm, which is DES in this case, and decrypted when needed. This fact implies that a decrypted password is accessed by the Kerberos server, which is not usually required in an authentication system that exploits public key cryptography. Therefore, the servers must be placed in locked rooms with physical security to prevent an attacker from stealing a password.

For a complete description about the Kerberos Version 5 protocol, refer to *RFC 1510 - The Kerberos Network Authentication Service (V5)*.

3.1.1 Kerberos protocol overview

The Kerberos system consists of three components: a client, a server, and a trusted third party, which is also known as a *Key Distribution Center* (KDC). KDC interacts with both a client and server to accept the client's request, authenticate its identity, and issue tickets to it.

The domain served by a single KDC is referred to as a *realm*. A *principal identifier* is used to identify each client and server in a realm. The principal name is uniquely assigned for all clients and servers by the Kerberos administrator. All principals must be known to the KDC.

Although the Kerberos protocol consists of several sub-protocols, three exchanges are particularly interesting to most readers. The first phase exchange takes place between a client and the authentication server. In this phase, a client asks the authentication server that knows the secret keys of all clients in the realm to authenticate himself and give the client a ticket (called a *ticket-granting ticket*) to be used to get a secret key which is then shared with an application server the client wants to access.

Upon receiving the ticket-granting ticket, the client sends a request that contains the ticket-granting ticket, for a service ticket to the ticket-granting server, and waits for a service ticket to be returned. Having the session ticket ready, the client is allowed to communicate with the server that is providing a service he wants to use. Optionally, the application server can perform further authentication processing against the client.

Figure 3-1 shows an overview of the Kerberos protocol.

Message encoding defined in Kerberos Version 5 is described using the Abstract Syntax Notation 1 (ASN.1) syntax, in accordance with ISO standards 8824 and 8825.

In the following sections, we discuss the interactions in more detail using the following notations:

- ▶ K_x : X's symmetric encryption key
- ▶ $K_{x,y}$: Encryption key shared by X and Y (for example, a session key)
- ▶ $K_x\{\text{data}\}$: A message that contains data encrypted with X's key

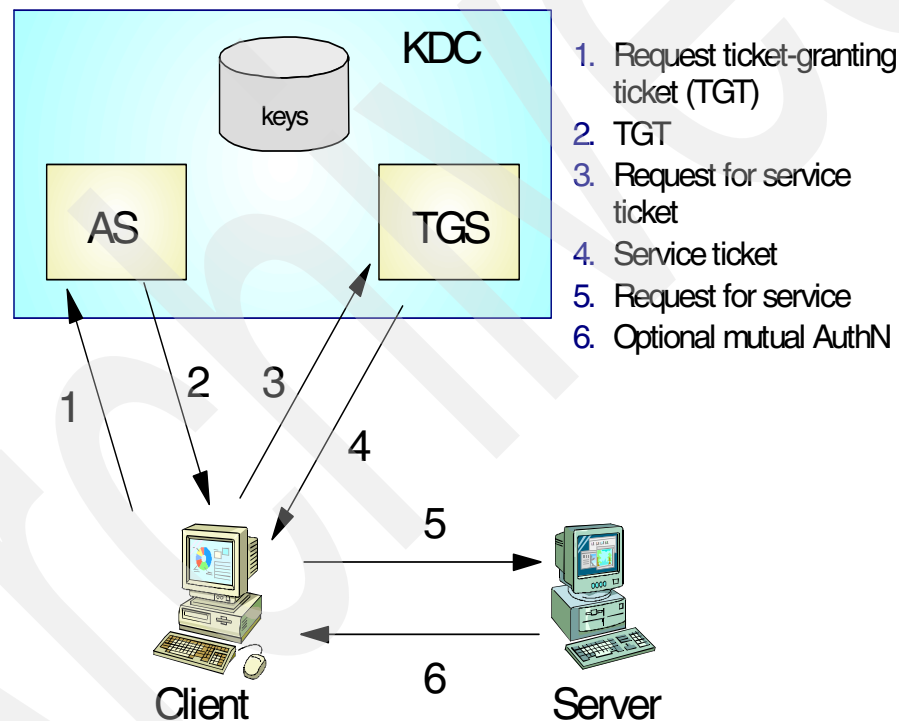


Figure 3-1 Kerberos protocol overview

Phase 1: Authentication service exchange

The authentication service exchange is initiated by a client when it wants to get authentication credentials for an application server but currently holds no credentials. Two messages are exchanged between the client and the Kerberos authentication server, then credentials for a ticket-granting server are given to the client. These credentials are the so-called ticket-granting ticket, which will subsequently be used to obtain credentials for other services.

This exchange is used for other services, such as the password-changing service, as well. Note that the client's secret key is used exclusively in this phase.

When a user logs into a client system and enters her password, a client sends the Kerberos authentication server a message that includes a user name in plain text ("Alice"), the current time encrypted with her secret key, and the identity of the server for which the client is requesting credentials (*ts* in Figure 3-2).

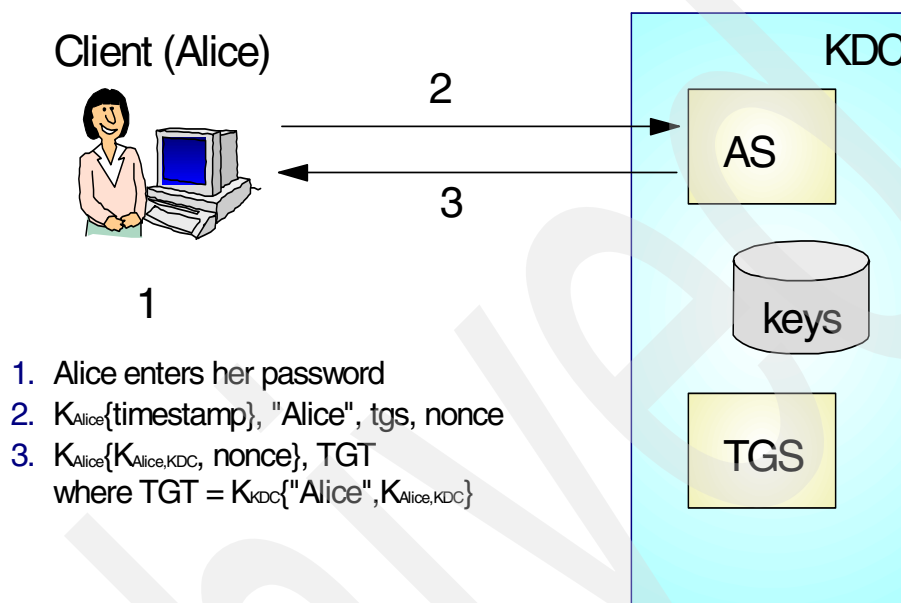


Figure 3-2 Simplified authentication service exchange

Upon receiving the request from the client, the authentication server looks up the client name and the service name (the ticket-granting service in this case) in the Kerberos database, and obtains an encryption key for each of them, K_{Alice} and K_{KDC} .

The authentication server then generates a response back to the client, which contains the ticket-granting ticket and a session key $K_{\text{Alice}, \text{KDC}}$, which is used in the subsequent secure communication between the client and KDC. The ticket-granting ticket includes the session key $K_{\text{Alice}, \text{KDC}}$, the identities of the server and the client, lifetime, and some other information. The authentication server then encrypts the ticket using its own key K_{KDC} . This produces a *sealed ticket*. The session key $K_{\text{Alice}, \text{KDC}}$ is also encrypted using the client's key K_{Alice} with some other information, such as nonce.

The encrypted current time is also known as the *authenticator*, since the receiver can assure that the sender knows the correct shared secret K_{Alice} , which is the client's encryption key derived from her password (this key is also referred as Alice's *long-term key*), by decrypting it and validating what is inside. Because the authentication server knows Alice's secret key, it can evaluate the time decrypted from the received authenticator. As you might have noticed, the clocks on the client system and the KDC must be reasonably synchronized with each other. A network time service may be used for this purpose.

An authenticator is also used to help the server detect message replays.

Nonce is information used to identify a pair of the Kerberos request and response. A timestamp or a random number generated by a client may be used.

Tgs is the server's identification, which is the Kerberos ticket-granting server in this case.

Since K_{Alice} is known exclusively by Alice and the KDC, no one but Alice can extract the critical information from the response message, such as the session key $K_{\text{Alice},\text{KDC}}$ to be used in the next phase.

When the client receives the authentication server's response, it decrypts it using its secret key K_{Alice} and checks to see if the nonce matches the specific request. If the nonce matches, the client caches the session key $K_{\text{Alice},\text{KDC}}$ for future communications with the ticket-granting server.

Phase 2: Ticket-granting service exchange

The next phase is used for a client to obtain credentials for services it wants to use. This exchange is also initiated by the client, and two messages are exchanged between the client and the ticket-granting server. The protocol and message format used in this exchange is almost identical to those for the authentication server exchange. The primary difference is that the client's key is never used in this exchange, but the session key obtained from the preceding authentication server exchange is used.

The request message the client send to the ticket-granting server contains several pieces of information, including:

- ▶ Information to authenticate the client, which includes a new authenticator and the ticket-granting ticket obtained in the preceding authentication server exchange
- ▶ Identity of the service for which the client is requesting credentials
- ▶ Nonce to identify this request

Figure 3-3 shows a simplified ticket-granting service exchange.

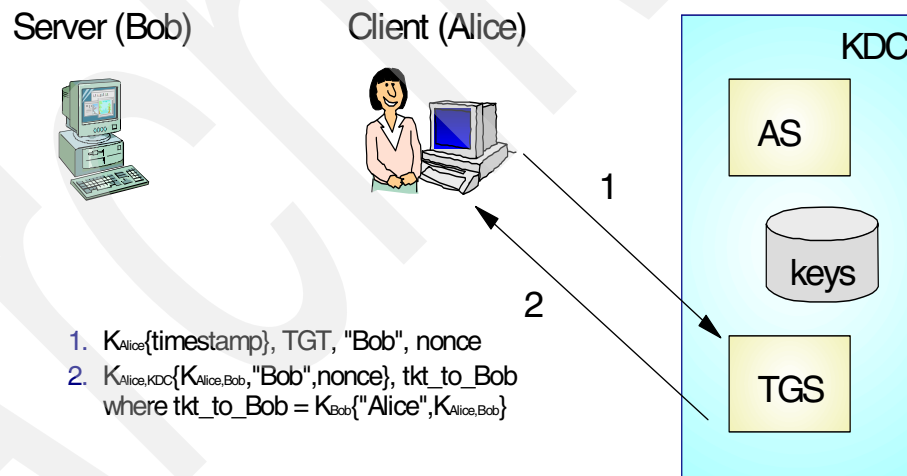


Figure 3-3 Simplified ticket-granting service exchange

When the ticket-granting server receives the message from the client, it first decipheres the sealed ticket using its encryption key K_{KDC} . From the deciphered ticket, the ticket-granting server obtains the session-key $K_{\text{Alice},\text{KDC}}$. It uses this session key to decipher the authenticator.

The validity checks performed by the ticket-granting server include verifying the following:

- ▶ The client name and its realm in the ticket match the same fields in the authenticator.
- ▶ The address from which this message originates is found in the address field in the ticket, which specifies addresses from which the ticket can be used.

- The user-supplied checksum in the authenticator matches the contents of the request. This procedure guarantees the integrity of the message.

Finally, it checks the current time in the authenticator to make certain the message is recent. Again, this requires that all the clients and servers maintain their clocks within some prescribed tolerance.

Important: By checking the timestamp in the nanoseconds scale, replay attacks can be detected.

The ticket-granting server now looks up the server name from the message in the Kerberos database, and obtains the encryption key K_{Bob} for the specified service.

The ticket-granting server forms a new random session key $K_{Alice,Bob}$ for the benefit of the client (Alice) and the server (Bob), and then creates a new ticket tk_{to_Bob} containing:

- The session key $K_{Alice,Bob}$
- Identities of the service and the client
- Lifetime

Note: The format of the ticket for a particular service is identical to one of the ticket-granting tickets.

The ticket-granting server then assembles and sends a message to the client.

Phase 3: The client/server authentication exchange

The client/server authentication exchange is performed by the client and the server to authenticate each other. The client has been issued credentials for the server using the authentication service or ticket-granting service exchange before the client/server exchange is initiated.

After receiving the ticket-granting server exchange response from the ticket-granting server, the client deciphers it using the ticket-granting server session key $K_{Alice,KDC}$ that is exclusively known by the client and the ticket-granting server. From this message it extracts a new session key $K_{Alice,Bob}$ that is shared with the server (Bob) and the client (Alice). The sealed ticket included in the response from the ticket-granting server cannot be deciphered by the client because it is enciphered using the server's secret key K_{Bob} .

Then the client builds an authenticator and seals it using the new session key $K_{Alice,Bob}$. Finally, it sends a message containing the sealed ticket and the authenticator to the server (Bob) to request its service.

When the server (Bob) receives this message, it first deciphers the sealed ticket using its encryption key K_{Bob} , which is kept in secret between Bob and the KDC. It then uses the new session key $K_{Alice,Bob}$ contained in the ticket to validate the authenticator in the same way as the ticket-granting server does in the ticket-granting server exchange.

Figure 3-4 on page 91 shows a simplified client/server authentication exchange.

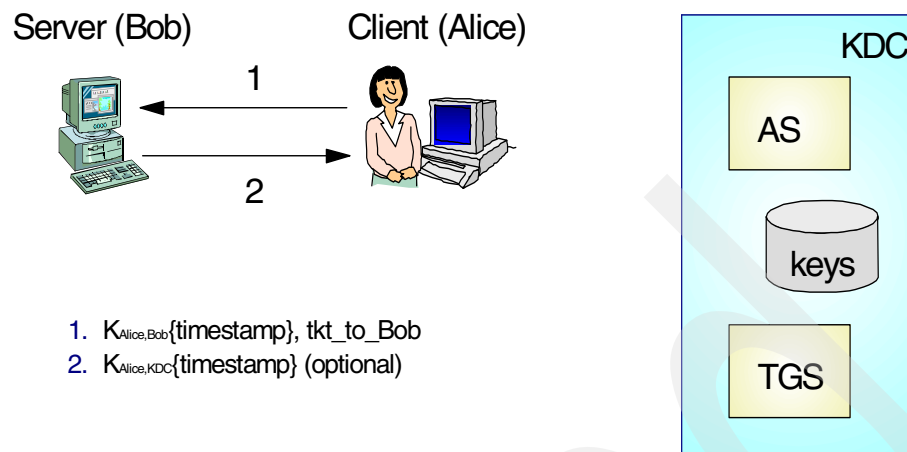


Figure 3-4 Simplified client/server authentication exchange

Once the server has authenticated a client, an option exists for the client to validate the server (this procedure is called *mutual authentication*). This prevents an intruder from impersonating the server.

If mutual authentication is required by the client, the server has to send a response message back to the client. The message has to contain the same timestamp value as one in the client's request message. This message is enciphered using the session key $K_{\text{Alice,Bob}}$ that was passed from the client to the server.

If the response is returned, the client decrypts it using the session key $K_{\text{Alice,Bob}}$ and verifies that the timestamp value matches one in the authenticator that was sent by the client in the preceding client/server exchange. If it matches, then the client is assured that the server is genuine.

Once the client/server exchange has completed successfully, an encryption key is shared by the client and server and can be used for the on-going application protocol to provide data confidentiality.

3.1.2 Inter-realm operation

The Kerberos protocol is designed to operate across organizational boundaries. Each organization wishing to run a Kerberos server establishes its own realm. The name of the realm in which a client is registered is part of the client's name and can be used by the application server to decide whether to honor a request.

By establishing inter-realm keys, the administrators of two realms can allow a client authenticated in one realm to use its credentials in the other realm. The exchange of inter-realm keys registers the ticket-granting service of each realm as a principal in the other realm. A client is then able to obtain a ticket-granting ticket for the remote realm's ticket-granting service from its local ticket-granting service. Tickets issued to a service in the remote realm indicate that the client was authenticated from another realm.

This method can be repeated to authenticate throughout an organization across multiple realms. To build a valid authentication path to a distant realm, the local realm must share an inter-realm key with the target realm or with an intermediate realm that communicates with either the target realm or with another intermediate realm.

Realms are typically organized hierarchically. Each realm shares a key with its parent and a different key with each child. If an inter-realm key is not directly shared by two realms, the hierarchical organization allows an authentication path to be easily constructed. If a hierarchical organization is not used, it may be necessary to consult some database in order to construct an authentication path between realms.

Although realms are typically hierarchical, intermediate realms may be bypassed to achieve cross-realm authentication through alternate authentication paths. It is important for the end-service to know which realms were transited when deciding how much faith to place in the authentication process. To facilitate this decision, a field in each ticket contains the names of the realms that were involved in authenticating the client.

3.1.3 Some assumptions

The following limitations apply to the Kerberized security environment:

- ▶ *Denial-of-service (DoS)* attacks are not addressed by Kerberos. There are places in these protocols where an intruder can prevent an application from participating in the proper authentication steps. Detection and solution of such attacks (some of which can appear to be “usual” failure modes for the system) is usually best left to human administrators and users.

Important: OS/390 (and z/OS also) has the built-in ability to limit the possibility of DoS attacks. In addition, some services provided by z/OS CS/390, such as TRMD, may be used to protect against attacks like resource hogging.

- ▶ The secret key must be kept secret by each principal (each client and server). If an attacker steals a principal's key, it can then masquerade as that principal or impersonate any server of the legitimate principal.
- ▶ Kerberos does not address *password guessing* attacks. If a poor password is chosen, an attacker may be able to mount an offline dictionary attack by repeatedly attempting to decrypt messages that are encrypted with a key derived from the user's password.
- ▶ Kerberos assumes a loosely synchronized clock in the whole system. Workstations may be required to have a synchronization tool such as the time server provided.
- ▶ Principal identifiers should not be reused on a short-term basis. Access control lists (ACLs) may be used to grant permissions to particular principals.

3.2 Implementing NAPS

The implementation of NAPS introduces a new UNIX daemon to provide the KDC services (authentication server and ticket-granting server). RACF has been enhanced to provide KDC registry functions, to store principals and keys.

This section details the setup of the NAPS UNIX daemon and the required configuration files.

3.2.1 SKRBDKDC daemon setup

The following steps describe the setup of the NAPS UNIX daemon called SKRBDKDC:

1. Copy the SKRBDKDC started task procedure (JCL) from EUVF.SEUVSAM to SYS1.PROCLIB or the procedure library you use in your installation for started tasks. An example of the SKRBDKDC started task procedure is shown in Figure 3-5 on page 93.

```

/ SKRBKDC PROC REGSIZE=256M,OUTCLASS='A'
//*****
//*
//* Procedure for starting the Kerberos Security Server
//*
//*****
//GO      EXEC PGM=EUVFSKDC,REGION=&REGSIZE,TIME=1440,
// PARM=('ENVAR("LANG=En_US.IBM-1047"),TERM(DUMP)           X
//          / 1>DD:STDOUT 2>DD:STDERR')
//STDOUT   DD SYSOUT=&OUTCLASS,DCB=LRECL=250
//STDERR   DD SYSOUT=&OUTCLASS,DCB=LRECL=250
//SYSOUT   DD SYSOUT=&OUTCLASS
//CEEDUMP  DD SYSOUT=&OUTCLASS

```

Figure 3-5 SKRBKDC started task procedure

2. Add the data set EUVF.SEUVFLPA to your LPALSTxx member of SYS1.PARMLIB.
Add the data set EUVF.SEUVFLNK to your LNLSTxx member of SYS1.PARMLIB.

Important: An IPL is required to make the changes to LPALST and LNKLST permanent. We are using a TEST environment, so we can issue the **SETPROG** commands to dynamically add the data sets. We do not recommend that these commands be issued within a Production environment.

The **SETPROG** command through SDSF, shown in Figure 3-6, will dynamically add the LPALST data set.

```

System Command Extension

Type or complete typing a system command, then press Enter.

==> SETPROG LPA,ADD,MASK(*),DSNAME=EUVF.SEUVFLPA
==>
==>

F1=Help    F2=Split    F3=Cancel    F9=Swap    F12=Cancel

```

Figure 3-6 SETPROG command to dynamically add the LPALST data set

3. You can also add the LNKLST data set dynamically through SDSF with a series of **SETPROG** commands. The first command, shown in Figure 3-7, defines a temporary LNKLST name set of data sets for the LNKLST concatenation using the current LNKLST member.

```

System Command Extension

Type or complete typing a system command, then press Enter.

==> SETPROG LNKLST,DEFINE,NAME=LNKLSTXX,COPYFROM=CURRENT
==>
==>

F1=Help    F2=Split    F3=Cancel    F9=Swap    F12=Cancel

```

Figure 3-7 SETPROG command to define a temporary LNKLST name

The second **SETPROG** command, shown in Figure 3-8, connects the new LNKST name with the data set name and volume.

```
System Command Extension

Type or complete typing a system command, then press Enter.

====> SETPROG LNKST,ADD,NAME=LNKLSTXX,DSNAME=
====> EUVF.SEUVFLNK,VOLUME=LEMON1
====>

F1=Help    F2=Split    F3=Cancel    F9=Swap    F12=Cancel
```

Figure 3-8 *SETPROG* command to connect new LNKST data set

The last **SETPROG** command for the LNKST dynamic update is shown in Figure 3-9; it will activate the temporary LNKST member.

```
System Command Extension

Type or complete typing a system command, then press Enter.

====> SETPROG LNKST,ACTIVATE,NAME=LNKLSTXX
====>
====>

F1=Help    F2=Split    F3=Cancel    F9=Swap    F12=Cancel
```

Figure 3-9 *SETPROG* command to active new LNKST data set.

4. Define a group for your started task User ID, with an OMVS segment with a GID value, as shown in Figure 3-10. The owner specified in our examples is for our installation only, you may want change this according to your installation standards.

```
ADDGROUP SKRBGRP OWNER(STCGROUP) SUPGROUP(STCGROUP) DATA('GROUP FOR KERBEROS
SKRBKDC User ID') OMVS(GID(20))
```

Figure 3-10 *ADDGROUP* command example to define SKRBGRP

To verify the group is indeed defined correctly, display the group with all the attributes (OMVS), as shown in Figure 3-11.

```
LISTGRP SKRBGRP OMVS

INFORMATION FOR GROUP SKRBGRP
  SUPERIOR GROUP=STCGROUP    OWNER=STCGROUP
  INSTALLATION DATA=GROUP FOR KERBEROS User ID SKRBKDC
  NO MODEL DATA SET
  TERMUACC
  NO SUBGROUPS
OMVS INFORMATION
-----
GID= 0000000020
```

Figure 3-11 *Output from LISTGRP SKRBGRP command*

5. Define a started task User ID with an OMVS segment with the following values:
 - UID value: 0
 - HOME (directory) value: /etc/skrb/home/kdc

- PROGRAM value: /bin/sh

Attention: Both the HOME and PROGRAM values are case sensitive and need to be defined in lower case.

An example definition is shown in Figure 3-12.

```
ADDUSER SKRBKDC OW(SKRBGRP) DEFLTGRP(SKRGPR) NAME('KERBEROS User ID')
OMVS(UID(0) HOME(/etc/skrb/home/kdc) PROG(/bin/sh))
```

Figure 3-12 ADDUSER command example to define user SKRBKDC

Use the RACF **LISTUSER** command to check that the User ID is correctly defined, as shown in Figure 3-13.

```
LISTUSER SKRBKDC OMVS

USER=SKRBKDC  NAME=KERBEROS User ID      OWNER=SKRBGRP  CREATED=00.152
DEFAULT-GROUP=SKRBGRP  PASSDATE=00.154  PASS-INTERVAL=180
ATTRIBUTES=NONE
REVOKE DATE=NONE  RESUME DATE=NONE
LAST-ACCESS=00.154/08:36:36
CLASS AUTHORIZATIONS=NONE
NO-INSTALLATION-DATA
NO-MODEL-NAME
LOGON ALLOWED  (DAYS)          (TIME)
-----
ANYDAY
GROUP=SKRBGRP  AUTH=USE  ANYTIME
CONNECTS= 12  UACC=NONE  CONNECT-OWNER=SKRBGRP  CONNECT-DATE=00.152
CONNECT ATTRIBUTES=NONE  LAST-CONNECT=00.154/08:36:36
REVOKE DATE=NONE  RESUME DATE=NONE
SECURITY-LEVEL=NONE SPECIFIED
CATEGORY-AUTHORIZATION
NONE SPECIFIED
SECURITY-LABEL=NONE SPECIFIED

OMVS INFORMATION
-----
UID= 0000000000
HOME= /etc/skrb/home/kdc
PROGRAM= /bin/sh
CPUTIMEMAX= NONE
ASSIZEMAX= NONE
FILEPROCMA= NONE
PROCUSERMAX= NONE
THREADSMAX= NONE
MMAPAREAMAX= NONE
```

Figure 3-13 Output from LISTUSER SKRBKDC command

6. Define a profile for the SKRBKDC started task in the RACF STARTED class, as shown in Figure 3-14.

```
RDEFINE STARTED SKRBKDC.** OWNER(STCGROUP) STDATA(USER(SKRBKDC) GROUP(SKRB-
GRP))
```

Figure 3-14 SKRBKDC started task definitions example

Check the new defined profile by listing it, as shown in Figure 3-15.

```
RLIST STARTED SKRBKDC.** STDATA

CLASS      NAME
-----
STARTED    SKRBKDC.** (G)

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----
00     STCGROUP      NONE              NONE         NO

INSTALLATION DATA
-----
NONE

APPLICATION DATA
-----
NONE

AUDITING
-----
FAILURES(READ)

NOTIFY
-----
NO USER TO BE NOTIFIED

STDATA INFORMATION
-----
USER= SKRBKDC
GROUP= SKRBGRP
TRUSTED= NO
PRIVILEGED= NO
TRACE= NO
```

Figure 3-15 Output from RLIST command

3.2.2 Setting up the Kerberos environment variable files

The Kerberos environment variable files `/etc/skrb/krb5.conf` and `/etc/skrb/home/kdc/envvar` must be customized for your environment.

Samples of these configuration files are supplied in `/usr/lpp/skrb/examples`. Copy the samples to the locations indicated previously.

The `krb5.conf` file requires the following updates:

1. Update the `default_realm` parameter with your installation's Kerberos realm for the OS/390 system.

Our DNS name for our OS/390 system is `WTSC57.KRB390.IBM.COM` and our Kerberos realm is `KRB390.IBM.COM`

2. Update the `realms` parameter with your OS/390 realms and any other so-called peer realms.

We updated the `realms` parameter with our OS/390 realm `KRB390.IBM.COM` and added the DNS name for the OS/390 KDC and the OS/390 `kpasswd_server`.

3. Update the `domain_realm` parameter to reflect the OS/390 Realm in lowercase and uppercase.

Figure 3-16 displays the configuration file, `/etc/skrb/krb5.conf` and displays our changes to the configuration file.

```
Ÿlibdefaults"
default_realm = KRB390.IBM.COM
kdc_default_options = 0x40000010
use_dns_lookup = 0
Ÿrealms"
KRB390.IBM.COM = {
    kdc = wtsc57.krb390.ibm.com:88
    kpasswd_server = wtsc57.krb390.ibm.com:464
}
KRB2000.IBM.COM = {
    kdc = pauldeg.krb2000.ibm.com:88
    kpasswd_server = pauldeg.itso.ibm.com:464
}
Ÿdomain_realm"
.krb2000.ibm.com = KRB2000.IBM.COM
.krb390.ibm.com = KRB390.IBM.COM
```

Figure 3-16 Kerberos configuration file

The next step is to configure the environment variable file `/etc/skrb/home/kdc/envvar` with the required changes for your environment.

The defaults in this file are usually fine, except perhaps the timezone and the required logging you want to perform for the Kerberos server (`SKRBKDC`).

Figure 3-17 displays our configuration file, with our changes.

```
General server options
SKDC_DATABASE=SAF
SKDC_PORT=88
SKDC_KPASSWD_PORT=464
SKDC_NETWORK_THREADS=15
SKDC_LOCAL_THREADS=15
SKDC_LOGIN_AUDIT=FAILURE
System configuration options
LANG=En_US.IBM-1047
TZ=EST5EDT
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/En_US.IBM-1047/%N
Message/debug options
_EUV_SVC_MSG_LOGGING=STDOUT_LOGGING
_EUV_SVC_DBG_MSG_LOGGING=1
_EUV_SVC_DBG=KRB_KDC.8,KRB_KDB.8
_EUV_EXC_ABEND_DUMPS=0
```

Figure 3-17 Environment variable definitions for the Kerberos server

This completes the setup for the Kerberos server, but before you start the Kerberos server, some additional RACF definitions are required. These are discussed in 3.3, “Kerberos integrated with RACF” on page 98.

3.2.3 Setting up HFS for Kerberos cache files

The Kerberos runtime stores network credentials in so-called *cache files*. These are stored in an HFS directory called `/var/skrb/creds`. This directory structure does not exist by default, and requires setup. Also, these files should be erased periodically. There are several ways to do this:

- ▶ Use a temporary file system mounted at `/var/skrb/creds`. This results in all the credentials cache files being deleted each time the system is restarted.
- ▶ Erase all of the files in `/var/skrb/creds` when the `/etc/rc` initialization script is run. This results in all of the credentials cache files being deleted each time the system is restarted.
- ▶ Set up a **cron** job to run the **kdestroy** command with the **-e** option. This results in the deletion of only expired credentials cache files. This is the preferred method for managing the credentials cache files. The **cron** job should run with UID 0 so that it can delete the cache files.

The `/var/skrb/creds` directory permission bits should be set to 777 using the **chmod** command, as shown in Figure 3-18.

```
chmod 777 /var/skrb/creds
```

Figure 3-18 `chmod` command example to set permission bits for credentials cache(s)

3.3 Kerberos integrated with RACF

RACF provides the functions to customize and access data for use with Kerberos. The Kerberos database is maintained within RACF. The OS/390 SecureWay NAPS Server requires a registry of principal and global information, which is stored via RACF through User and General Resource Profiles.

The administration of the OS/390 SecureWay NAPS Server is done through the RACF panels and commands and obtains this information via SAF callable service. Kerberos application servers can use SAF callable services to parse Kerberos tickets to obtain principal names, and to map from principal to RACF user and vice versa.

Local Kerberos principals are defined as RACF users with a KERB segment. The information about the local and foreign realms are defined in the RACF class REALM in specific profiles. The profiles contain:

- ▶ Local realm information, the name, key, and ticket lifetime (MIN, MAX and DEFAULT in seconds).
- ▶ Foreign realm trust relationships. These are defined in pairs, which also include a key.

RACF maps foreign Kerberos principals using the KERBLINK class profiles.

The Kerberos principal's password and the RACF user password are integrated. The Kerberos password is subject to RACF **SETOPTS** rules and installation-defined rules.

Principals must keep their secret keys secret. If an intruder steals a principal's key, it can then masquerade as that principal or impersonate any server to the legitimate principal.

RACF Remote Sharing Facility (RRSF) has to be defined in local mode in order to generate the corresponding Kerberos secret key whenever the user changes their password. Kerberos uses RRSF services to make sure this happens.

Some RRSF RACF functions require a previously established User ID association. A User ID association is an association between two or more user IDs on the same or different RRSF nodes. There are two type of User ID associations, *peer* and *managed*:

- ▶ A *peer* association allows either of the associated user IDs to direct commands to the other and allows password synchronization.
- ▶ In a *managed* association, one of the user IDs is designated as the managing ID, and the other is designated as the managed ID; the managed ID cannot direct commands to the managing ID. There is no password synchronization in a managed association.

In order to be able to use the password synchronization and command direction functions, you need to activate and define profiles in to the RRSFDATA class. For more information refer to the redbook *RACF Version 2 Release 2 Installation and Implementation Guide*, SG24-4580.

3.3.1 Defining RRSF in local mode

The following steps are required to define RRSF in local mode:

1. Activate the RACF RRSFDATA class if not already activated. The RRSFDATA class needs to be RACLISTed and activated for generic command processing:

```
SETOPTS CLASSACT(RRSFDATA)
SETOPTS GENERIC(RRSFDATA)
SETOPTS GENCMD(RRSFDATA)
SETOPTS RACLIST(RRSFDATA)
```

Figure 3-19 SETOPTS command example to activate local RRSF mode

- Define a new member IRROPT01 in SYS1.PARMLIB and include the **TARGET** command as shown in Figure 3-20 to configure the RRSF in local mode.

```

TARGET -
NODE(SC57) -
DESCRIPTION('WS57TS SYSTEM') -
PREFIX(SYS1.RACF) -
OPERATIVE LOCAL -
WORKSPACE(VOLUME(PDGTS1))

```

Figure 3-20 IRROPT01 member in SYS1.PARMLIB

Table 3-1 displays the **TARGET** parameters and their values.

Table 3-1 TARGET parameters

TARGET parameter	Description
NODE(<i>nodename</i>)	Specifies the generic name of the TARGET node to be defined.
DESCRIPTION('description')	Specifies a comment used to describe this TARGET node.
PREFIX(<i>qualifier...</i>)	Specifies the high-level qualifiers that RRSF uses for the workspace data set.
OPERATIVE	Specifies that the connection to the TARGET node is to be made active.
LOCAL	Indicates the node being defined is the local node. If not specified, the default on a new definition is a remote mode.
WORKSPACE	Specifies allocation information concerning the workspace data sets.

- Modify the RACF procedure in SYS1.PROCLIB to process the updated RACF parameter library by adding PARM='OPT=01' to the EXEC statement. Add the RACFPARM ddname to point to SYS1.PARMLIB to identify the library that contains the RRSF parameters. Figure 3-21 shows the RACF procedure in SYS1.PROCLIB.

```

//RACF PROC
//RACF EXEC PGM=IRRSSM00,REGION=0M,PARM='OPT=01'
//RACFPARM DD DSN=SYS1.PARMLIB,DISP=SHR

```

Figure 3-21 RACF procedure in SYS1.PROCLIB

- For these changes to take effect, refresh the RACF subsystem by stopping and restarting the RACF started task using the locally defined RACF subsystem prefix '#'. Figure 3-22 displays the output of the **STOP** command.

```

#STOP
IRRB069I (#) RACF SUBSYSTEM STARTING SHUTDOWN PROCESSING. 208
IRRJ001I (#) RACF RACF LOCAL NODE TRANSACTION PROGRAM COMPLETED UNDER
209
          USER ID RACF GROUP TSO.
IRRB005I (#) RACF SUBSYSTEM TERMINATION IS COMPLETE. 214
IEF404I RACF - ENDED - TIME=14.26.01

```

Figure 3-22 Output of #STOP command

Issue the MVS **START** command, specifying RACF as the procedure name. Figure 3-23 displays the MVS **START RACF** command output.

```

S RACF,SUB=MSTR
IRR812I PROFILE RACF.** (G) IN THE STARTED CLASS WAS USED 78
      TO START RACF WITH JOBNAME RACF.
IRR001I (#) RACF SUBSYSTEM HRF7703 IS ACTIVE. 841
IRRG008I (#) RACF SUBSYSTEM IS PROCESSING PARAMETER LIBRARY MEMBER 842
      IRROPT01.
IRRM002I (#) RACF SUBSYSTEM TARGET COMMAND HAS COMPLETED SUCCESSFULLY.
847
IRRG010I (#) RACF SUBSYSTEM PROCESSING OF PARAMETER LIBRARY MEMBER 848
      IRROPT01 IS COMPLETE.
IRR002I (#) INITIALIZATION COMPLETE FOR RACF SUBSYSTEM. 849
IRRJ000I (#) RACF RACF LOCAL NODE TRANSACTION PROGRAM STARTING UNDER
850
      USER ID RACF GROUP TSO.

```

Figure 3-23 Output of **START RACF** subsystem command

5. You can check the status of the RRSF environment using the **TARGET LIST** command, using the locally defined RACF subsystem prefix '#', as shown in Figure 3-24 along with the output from the command.

```

#TARGET LIST
IRRM009I (#) LOCAL RRSF NODE SC57 IS IN THE OPERATIVE ACTIVE STATE.

```

Figure 3-24 Output of **TARGET LIST** command

6. You can also check the RACF subsystem using the **SET LIST** command. Figure 3-25 shows this command and the output.

```

#SET LIST
IRR005I (#) RACF SUBSYSTEM INFORMATION: 566
TRACE OPTIONS                - NOIMAGE
                             - NOAPPC
SUBSYSTEM USERID             - RACF
JESNODE (FOR TRANSMITS)      - WTSCICF
AUTOMATIC COMMAND DIRECTION IS *NOT* ALLOWED
AUTOMATIC PASSWORD DIRECTION IS *NOT* ALLOWED
PASSWORD SYNCHRONIZATION IS *NOT* ALLOWED
AUTOMATIC DIRECTION OF APPLICATION UPDATES IS *NOT* ALLOWED
RACF STATUS INFORMATION:
  TEMPLATE VERSION           - HRF7703
  DYNAMIC PARSE VERSION      - HRF7703

```

Figure 3-25 Output of **SET LIST** command

3.3.2 RACF setup for Kerberos realms

Use the following steps to set up RACF for Kerberos realms.

1. Before you define the local REALM, activate (CLASSACT) and RACLIST the REALM class, as shown in Figure 3-26.

```
SETOPTS CLASSACT(REALM)
SETOPTS RACLIST(REALM)
```

Figure 3-26 SETOPTS command example to activate the REALM class

2. The PTKTDATA class, if not already active, has to be activated and has to be RACLISTed, as shown in Figure 3-27.

```
SETOPTS CLASSACT(PTKTDATA)
SETOPTS GENERIC(PTKTDATA)
SETOPTS GENCMD(PTKTDATA)
SETOPTS RACLIST(PTKTDATA)
```

Figure 3-27 SETOPTS command example to activate the PTKTDATA class

Important: No profile is needed in the PTKTDATA class. The Kerberos server (SKRBKDC) generates a temporary passticket under the covers to change a principal's password when the **kpasswd** command is issued.

3. A user must have access to the SKRBKDC application in order to use the **kpasswd** command to change their password. By using the RACF **RDEFINE** command, you can define the SKRBKDC application to the RACF APPL class, as shown in Figure 3-28.

```
RDEFINE APPL SKRBKDC OWNER(SYS1) UACC(READ) DATA('KERBEROS APPLID')
```

Figure 3-28 RDEFINE command example to define the SKRBKDC application

Tip: Alternately, you can set the Universal Access to NONE and explicitly authorize individual groups or users to the SKRBKDC application.

Attention: Although the product manual indicates you to define this APPL profile, the **kpasswd** command is not delivered in OS/390 Version 2 Release 10. Therefore, we feel that you can skip this step.

In z/OS Release 2 the **kpasswd** command is supported.

4. Define your local realm to the REALM class, using the RACF **RDEFINE** command to define the KERBDFLT profile reflecting the default REALM and policy, as shown in Figure 3-29.

```
RDEFINE REALM KERBDFLT KERB(KERBNAME(KRB390.IBM.COM)
PASSWORD(password) MINTKTFLE(15) DEFTKTFLE(36000) MAXTKTLFE(86400)UACC=NONE
```

Figure 3-29 RDEFINE command example to define the local REALM

Attention: Our OS/390 environment has a domain name of WTSC57.KRB390.IBM.COM and a REALM name of KRB390.IBM.COM

Figure 3-29 shows that the local REALM is defined with a minimum ticket lifetime of 15 seconds, a maximum ticket lifetime of 24 hours and a default ticket lifetime of 10 hours.

Use the RACF **RLIST** command to display the KERBDFLT profile in the REALM class, as shown in Figure 3-30.

RLIST REALM KERBDFLT KERB				
CLASS	NAME			
-----	----			
REALM	KERBDFLT			
LEVEL	OWNER	UNIVERSAL ACCESS	YOUR ACCESS	WARNING
-----	-----	-----	-----	-----
00	LEMON	NONE	NONE	NO
..... removed for brevity				
KERB INFORMATION				

KERBNAME= KRB390.IBM.COM				
MINTKTLE= 0000000015				
MAXTKTLE= 0000086400				
DEFTKTLE= 0000036000				
KEY VERSION= 001				

Figure 3-30 Partial output from RLIST REALM KERBDFLT

5. Define any foreign REALMs to the RACF REALM class.

Your local Network Authentication and Privacy Service (Kerberos) server can trust authentications completed by other servers, and can be trusted by other servers, by participating in trust relationships.

To participate in trust relationships, you must define each server as a foreign realm. Then, you can allow users who are authenticated in foreign realms (foreign principals) to access protected resources on your local OS/390 system by mapping one or more RACF user IDs to foreign principal names. You do not need to provide foreign principals with the ability to log on to your local OS/390 system. You can simply provide mapping to one or more local user IDs so they can gain access privileges for local resources that are under the control of an OS/390 application server, such as DB2.

In our example, we defined a Windows 2000 Realm to the RACF REALM class, so that we can use it later for testing the Kerberos integration between Windows 2000 and the OS/390 Network Authentication and Privacy Services using DB2.

The Windows 2000 domain is called *pauldeg.krb2000.ibm.com* and the REALM is called *KRB2000.IBM.COM*. We defined the following profiles to set up the trust relationship between the OS/390 REALM and the Windows 2000 REALM:

```
RDEFINE REALM /.../KRB390.IBM.COM/krbtgt/KRB2000.IBM.COM KERB(PASSWORD(xx))
RDEFINE REALM /.../KRB2000.IBM.COM/krbtgt/KRB390.IBM.COM KERB(PASSWORD(xx))
```

Important: The password defined here is needed later when we define the same trust relationship on our Windows 2000 domain.

This password is not associated with any User ID and is not constraint to any **SETROPTS** rules for passwords.

6. Define Kerberos port 88 for the KDC and port 464 for the password server to your TCP/IP profile, to reflect the use of these ports, as shown in Figure 3-31 on page 104.

```
88 TCP OMVS SAF KERB88      ; Kerberos Server
464 TCP OMVS SAF KERB464    ; Kerberos Server
```

Figure 3-31 TCP/IP profile data set showing the Kerberos ports

7. Depending on your installation, you may or may not have started with the protection of TCP/IP ports using the RACF SERVAUTH class.
8. Accordingly, you should authorize the SKRBKDC Started Task Userid to port 88 and 464, using the commands shown in Figure 3-32.

```
PERMIT EZB.PORTACCESS.SC57.ITCPIP.KERB88 CLASS(SERVUATH) ID(SKRBKDC)
ACCESS(READ)
PERMIT EZB.PORTACCESS.SC57.ITCPIP.KERB464 CLASS(SERVUATH) ID(SKRBKDC)
ACCESS(READ)
```

Figure 3-32 PERMIT command example to authorize Kerberos TCP/IP ports

The SKRBKDC started task also requires access to the TCP/IP stack itself, using a new profile in the RACF SERVAUTH class, as shown in Figure 3-33.

```
PERMIT EZB.STACKACCESS.SC57.ITCPIP CLASS(SERVUATH) ID(SKRBKDC) ACCESS(READ)
```

Figure 3-33 SERVAUTH profile to protect TCP/IP stack access

9. You are now ready to start your Kerberos server SKRBKDC.
You will receive the informational messages shown in Figure 3-34.

```
S SKRBKDC
$HASP100 SKRBKDC ON STCINRDR
IEF695I START SKRBKDC WITH JOBNAME SKRBKDC IS ASSIGNED TO USER
SKRBKDC , GROUP SKRBGRP
$HASP373 SKRBKDC STARTED EUVF04001I Security server version 2.10, Service
level 0W45102.
EUVF04002I Security runtime version 2.10, Service level 0W45102.
EUVF04018I Security server initialization complete.
```

Figure 3-34 SKRBKDC startup messages

3.4 Kerberos principals

This section describes how to define Kerberos principals. We distinguish between two types of principals, *local* and *foreign*.

- ▶ A *local* principal is a Kerberos user defined to the local REALM.
- ▶ A *foreign* principal is a Kerberos user from another Kerberos REALM.

3.4.1 Local principals

You define local principals as RACF users using the **ADDUSER** and **ALTUSER** commands with the new KERB option. This creates a KERB segment for the user. Each local principal must have a RACF password. Therefore, do not use the NOPASSWORD option when defining local principals. You can specify the following information for your local principals:

KERBNAME Local principal name.

Important: Upper and lower case letters are accepted and maintained in the case in which they are entered.

MAXTKLFE Maximum ticket lifetime for the local principal.

An example of defining a Kerberos principal is shown in Figure 3-35.

```
ALTUSER GRAAFF KERB(KERBNAME('Paul de Graaff'))
```

Figure 3-35 ALTUSER command example to assign a Kerberos local principal

Restriction: The local principal name specified can only be defined once. If you try to define it to two RACF User IDs, you receive the following error message:

IRR52165I The value for the KERB segment KERBNAME operand must be unique.

Command processing ends.

Generating keys for local principals

Each local principal must have a key registered with the local Network Authentication and Privacy Service (Kerberos) server in order to be recognized as a local principal. The user's definition as a local principal is not complete until the key is generated. The key is generated from the principal's RACF user password at the time of the user's password change. If you want a key to be generated, be sure to use a password change facility that will not result in an expired password that the user must change at next logon. For example, you can use the **NOEXPIRED** keyword of the **ALTUSER** command.

A local principal's key will be revoked whenever the user's RACF user ID is revoked or the RACF password is considered expired. If the user's key is revoked, the server will reject ticket requests from this user.

You can change a user's password so that a key can be generated using the **ALTUSER** command with the **NOEXPIRED** option, for example:

```
ALTUSER GRAAFF PASSWORD(new1pw)NOEXPIRED
```

Restriction: Do not use the **NOPASSWORD** option on the **ALTUSER** command.

Attention: You must specify a password value so a key can be generated.

All characters of the password will be folded to uppercase.

Users can change their own passwords, completing their own definitions as local principals, by using any standard RACF password-change facility, such as one of the following:

- ▶ **TSO PASSWORD** command (without the ID option)
- ▶ **TSO logon**
- ▶ **CICS signon**

Important: The RACF address space must be started for the password change to complete and the key to be generated.

Password change requests from applications that encrypt the password prior to calling RACF will not result in usable keys.

Automatic local principal name mapping

For each local principal you define on your system using the KERB keyword of the **ADDUSER** and **ALTUSER** commands, RACF automatically creates a mapping profile in the KERBLINK class. When you issue the **ALTUSER** command with the NOKERB keyword or issue a **DELUSER** for a user with a KERB segment, RACF automatically deletes the KERBLINK profile.

The KERBLINK profile maps the local principal name to the user's RACF user ID. The name of the KERBLINK profile for a local principal is the principal name specified as the KERBNAME value with the **ADDUSER** or **ALTUSER** command. We show the KERBLINK profile for user ID graaff in Figure 3-36 as it was defined in Figure 3-35 on page 105.

```
sr mask(P) class(kerblink)
PaulødeøGraaff
```

Figure 3-36 Local principal definition in the KERBLINK class

You can see in the profile *PaulødeøGraaff* that blanks are indeed replaced by ø character. If you list the profile, you notice a little quirk in the RACF command processing not accepting mixed-case profile names, as shown in Figure 3-37. This is explained further as a restriction in “Considerations for local principal names” on page 107.

```
r1 KERBLINK PaulødeøGraaff
ICH13003I PAULøDEøGRAAFF NOT FOUND
```

Figure 3-37 RLIST command example to list KERBLINK profile

If we do an **RLIST ***, we do see the output, as shown in Figure 3-38 on page 107.


```

RLIST *

CLASS      NAME
-----
KERBLINK   Paul¢de¢Graaff

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----
00     GRAAFF      NONE              NONE         NO

INSTALLATION DATA
-----
NONE

APPLICATION DATA
-----
GRAAFF

```

Figure 3-38 RLIST command out to show the KERBLINK profile

The figure shows that the local principal *Paul de Graaff* maps back to RACF User ID *GRAAFF*.

Considerations for local principal names

The name of the KERBLINK profile contains the local principal name that is being mapped. Local principal names may contain imbedded blanks and lower case characters, as shown in Figure 3-35 on page 105.

Blanks are not permitted as a part of a RACF profile name. Therefore, when building the KERBLINK profile name, as a result of specifying KERBNAME with the **ADDUSER** or **ALTUSER** command, RACF command processing will replace each blank with the X'4A' character (which often resolves to the ¢ symbol). This can be seen in the output from the **RLIST KERBLINK *** command shown in Figure 3-36 on page 106, and in the output from the RACF data base unload utility (IRRDBU00).

Restriction: RACF command processing also prevents the X'4A' character () from being specified as part of the actual local principal name.

3.4.2 Foreign principals

You map foreign principal names to RACF user IDs on your local OS/390 system by defining general resource profiles in the KERBLINK class. You can map each principal in a foreign realm to its own user ID on your local OS/390 system, or you can map all principals in a foreign realm to the same user ID on your system.

RACF user IDs that map to foreign principals do not need KERB segments. These user IDs are intended to be used only to provide local OS/390 identities to associate with access privileges for local resources that are under the control of an OS/390 application server, such as DB2.

Each mapping profile in the KERBLINK class is defined and modified using the **RDEFINE** and **RALTER** commands. The name of the KERBLINK profile for a foreign principal contains the principal name, fully qualified with the name of the foreign realm. The profile name uses the following format:

```
.../foreign_realm /[foreign-principal_name ]
```

If you wish to map a unique RACF user ID to each foreign principal, you must specify the foreign realm name and the foreign principal name. If you wish to map the same RACF user ID to every foreign principal in the foreign realm, you need only specify the foreign realm name. In each case, you specify the local user ID using the APPLDATA keyword of the **RDEFINE** or **RALTER** command.

Example of mapping foreign principal names

In the following example, the users *PAUL* and *VAL* have their foreign principal names mapped with individual user IDs on the local OS/390 system. All other foreign principals presenting tickets from the KERB2000.IBM.COM REALM are mapped to the KRB2000 user ID on the local OS/390 system.

```
RDEFINE KERBLINK /.../KERB2000.IBM.COM/PAUL APPLDATA('GRAAFF')
RDEFINE KERBLINK /.../KERB2000.IBM.COM/VAL APPLDATA('VALERIA')
RDEFINE KERBLINK /.../KERB2000.IBM.COM/ APPLDATA('KERB2000')
```

Attention: All characters of the foreign realm name and the foreign principal name will be folded to uppercase.

3.5 Kerberos commands

This section describes the Kerberos commands supplied with the Network Authentication and Privacy Services.

3.5.1 Description of the Kerberos commands

The following commands are supplied:

kdestroy	Destroys a Kerberos credentials cache file. To delete a credentials cache, the user must be the owner of the file or must be a root (uid) user.
keytab	Used to add or delete a key from a key table or to display the entries in a key table.
kinit	Obtains or renews a Kerberos ticket-granting ticket. The KCD options specified in the Kerberos configuration file are used if no ticket options are specified on the kinit command.
klist	Displays the contents of a Kerberos credentials cache or key table.
ksetup	Manages Kerberos service entries in the LDAP directory.

In order to use these commands, you must update your `PATH` statement in your `.profile` with the full path name of the directory (`/usr/lpp/skrb/bin`) containing the Kerberos commands. It also requires updates to the `NLSPATH` statement to reflect the Kerberos message catalog. Figure 3-39 on page 109 displays the required changes to your `.profile`.

```
# =====
# Start of Kerberos section
# =====
echo "--> Start of Kerberos Additions"
export PATH=$PATH:/usr/lpp/skrb/bin
echo "PATH:" $PATH
#
export NLSPATH=$NLSPATH:/usr/lpp/skrb/lib/nls/msg/%L/%N
echo "NLSPATH:" $NLSPATH
#
```

Figure 3-39 .profile content example showing Kerberos updates

Important: If you do not make these changes to your UNIX System Services environment, you will receive an error message indicating you are executing the DCE versions of these commands:

EUVP08317A The DCE runtime library cannot communicate with the DCE kernel.

For a complete description of these commands, see *OS/390 SecureWay Security Server Network Authentication and Privacy Service Administration*, SC24-5896.

3.5.2 Kerberos command examples

This section describes the use of the Kerberos commands.

kinit

The **kinit** command obtains or renews the Kerberos ticket-granting ticket. The KDC options specified by *kdc_default_options* in the Kerberos configuration file are used if no ticket options are specified on the **kinit** command. The **kinit** command has a lot of keywords, but we only show the following examples:

kinit -s	Obtains a ticket-granting ticket using the current signed-on RACF User ID
kinit	Obtains a ticket-granting ticket and the principal name is obtained from the credentials cache (if present)
kinit -k	Obtains a ticket-granting ticket using a key table to obtain the principal information

kinit -s command example

To obtain a ticket-granting ticket for a Kerberos principal, you can either use RACF services to obtain the principal associated or use a so-called key table. The **kinit -s** command obtains a ticket-granting ticket for the current signed-on RACF User ID, as shown in Figure 3-40.

```
GRAAFF @ SC57:/u/graafl/kerberos>kinit -s
EUVP06014E Unable to obtain initial credentials.
          Status 0x96c73a2d - Service key is not available.
```

Figure 3-40 kinit -s command example (1)

When we issued the **kinit -s** command for the current signed-on RACF User ID GRAAFF, we receive an error that the service key is not available. When we defined the local principal for User ID GRAAFF (see Figure 3-35 on page 105), a key was *not* generated for the local principal. When we issue the **LU GRAAFF KERB** command, no key is generated, as shown in Figure 3-41 on page 110.

```

LU GRAAFF KERB NORACF

USER=GRAAFF
KERB INFORMATION
-----
KERBNAME= Paul de Graaff

```

Figure 3-41 *LISTUSER* command example to display the Kerberos principal

As we explained in “Generating keys for local principals” on page 105, keys only get generated when a RACF password change occurs. So after we change the password for the RACF User ID GRAAFF, we get a key generated, as shown in Figure 3-42.

```

LU GRAAFF KERB NORACF

USER=GRAAFF
KERB INFORMATION
-----
KERBNAME= Paul de Graaff
KEY VERSION= 001

```

Figure 3-42 *LISTUSER* command example to show the key generated for the principal

We can now try again to get a ticket-granting ticket issued, using the **kinit -s** command, as shown in Figure 3-43.

```

GRAAFF @ SC57:/u/graafl/kerberos>kinit -s
GRAAFF @ SC57:/u/graafl/kerberos>klist
Ticket cache: FILE:/var/skrb/creds/krbcred_a9b31900
Default principal: Paul de Graaff@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/02/26-23:48:16 to 2001/02/27-09:48:16

```

Figure 3-43 *kinit -s* command example (2)

To display the ticket-granting ticket, we used the **klist** command, as shown in Figure 3-43. We discuss the **klist** command in more detail in “klist” on page 111.

kinit with no keywords example

Next we tested the **kinit** command without specifying any keywords. The **kinit** command obtains the principal name from the credentials cache. If no credential cache exists, the command will fail, as shown in Figure 3-44.

```

GRAAFF @ SC57:/u/graafl>kinit
EUVF06010E Principal name must be specified.

```

Figure 3-44 *kinit* command example (2)

Figure 3-45 on page 111 shows the interaction with the user when issuing the **kinit** command and a credential cache does exist. You are prompted for a password associated with the local principal. If you are using RACF instead of a key table for storage of local principals, then this is your RACF password associated with your RACF User ID.

```
GRAAFF @ SC57:/u/graaff>kinit
EUVF06017R Enter password:

GRAAFF @ SC57:/u/graaff>
```

Figure 3-45 *kinit* command example without any keywords.

Important: The password entered here has to be done in *uppercase* text. RACF only accepts uppercase passwords.

***kinit -k* command example**

We then tested the use of a key table with the **kinit** command rather than using RACF. The use of the **keytab** command to set up a key table is explained in “keytab” on page 114. For this example we assume a principal is defined called paul@KRB390.IBM.COM. Figure 3-46 shows using a key table with the **kinit** command, by using the **-k** keyword.

```
GRAAFF @ SC57:/u/graaff>kinit -k paul@KRB390.IBM.COM
GRAAFF @ SC57:/u/graaff>klist
Ticket cache: FILE:/var/skrb/creds/krbcred_b210de60
Default principal: paul@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/06/08-13:39:50 to 2001/06/08-23:39:50
GRAAFF @ SC57:/u/graaff>
```

Figure 3-46 *kinit -k* command example

Note: The password must be entered in *uppercase* text.

When using the **-k** keyword, you do not need to specify the name and location of the key table if you want to use the default key table. The default key table name is obtained from the `default_keytab_name` configuration file (`krb5.conf`) entry. The default name is `/etc/skrb/krb5.keytab`.

Tip: The default key table name can also be changed using the environment variable `KRB5_KTNAME`.

klist

The **klist** command is used to display the contents of a Kerberos credentials cache or key table. We show the following examples of using the **klist** command:

klist	This is the default, and lists the tickets in the credentials cache.
klist -e	Displays the encryption type for the session key and the ticket.
klist -f	Displays the ticket flags.
klist -k	Displays the entries in the keytable.
klist -k -K	Displays the encryption key value for each key table entry.

klist command example

When you issue the command **klist** without any keywords, it actually is as if you had issued a **klist -c** command. Figure 3-47 shows the output of a **klist** command after a ticket-granting ticket is obtained.

```
GRAAFF @ SC57:/u/graafl/kerberos>kinit -s
GRAAFF @ SC57:/u/graafl/kerberos>klist
Ticket cache: FILE:/var/skrb/creds/krbcred_a9b31900
Default principal: Paul de Graaff@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/02/26-23:48:16 to 2001/02/27-09:48:16
```

Figure 3-47 *klist command example output*

klist -e command example

The **klist -e** command displays the encryption type for the session key and the ticket, as shown in Figure 3-48.

```
GRAAFF @ SC57:/u/graafl>klist -e
Ticket cache: FILE:/var/skrb/creds/krbcred_b210de60
Default principal: paul@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/06/08-13:39:50 to 2001/06/08-23:39:50
Encryption type: DES_CBC_CRC
```

Figure 3-48 *klist -e command example*

Important: The **-e** option is valid only when listing a credentials cache.

klist -f command example

The **klist -f** command displays the ticket flags, as shown in Figure 3-49.

```
GRAAFF @ SC57:/u/graafl>klist -f
Ticket cache: FILE:/var/skrb/creds/krbcred_b210de60
Default principal: paul@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/06/08-13:39:50 to 2001/06/08-23:39:50
Flags: FIA
```

Figure 3-49 *klist -f command example*

The flags indicate the following:

- F Forwardable ticket
- I Initial ticket
- A Preauthentication used

Important: The **-f** option is valid only when listing a credentials cache.

For a detailed description of all the flags, see *OS/390 SecureWay Security Server Network Authentication and Privacy Service Administration*, SC24-5896.

klist -k command example

The **klist -k** command lists the entries in a key table, as shown in Figure 3-50.

```
GRAAFF @ SC57:/u/graaft>klist -k
Key table: /etc/skrb/krb5.keytab

Principal: paul@KRB390.IBM.COM
Key version: 1
```

Figure 3-50 *klist -k command example*

klist -k -K command example

The **klist -k -K** command lists the entries in a key table and displays the encryption key value for each key table entry, as shown in Figure 3-51.

```
GRAAFF @ SC57:/u/graaft>klist -k -K
Key table: /etc/skrb/krb5.keytab

Principal: paul@KRB390.IBM.COM
Key version: 1
Key: f1bc4fa49e4975ad
```

Figure 3-51 *klist -k -K command example*

kdestroy

The **kdestroy** command deletes a Kerberos credentials cache file.

The **-e** option causes the **kdestroy** command to check all of the credentials cache files in the default cache directory (/etc/skrb/var/creds). Any file that contains only expired tickets that have expired for the time delta are deleted. The time delta is expressed as nwndnhnmns, where n represents a number, w indicates weeks, d is days, h is hours, m is minutes, and s indicates seconds. The components must be specified in this order, but any component may be omitted (for example, 4h5m represents four hours and 5 minutes and 1w2h represents 1 week and 2 hours). If only a number is specified, the default is hours.

Important: To delete a credentials cache, the user must be the owner of the file or must be a root (uid 0) user.

Figure 3-52 on page 114 shows an example of the **kdestroy** command used to delete the credentials cache of principal Paul de Graaff.

```

GRAAFF @ SC57:/u/graaft>kinit -s
GRAAFF @ SC57:/u/graaft>klist
Ticket cache: FILE:/var/skrb/creds/krbcred_b2118de0
Default principal: Paul de Graaff@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
Valid 2001/06/08-14:26:38 to 2001/06/09-00:26:38
GRAAFF @ SC57:/u/graaft>kdestroy
EUVF06034I Credentials cache FILE:/var/skrb/creds/krbcred_b2118de0
destroyed.

```

Figure 3-52 *kdestroy command example*

keytab

The **keytab** command manages a key table. A key table can be used to define principals, either local or foreign. Key tables are traditionally used in UNIX-based environment. Support for key tables here provides compatibility with these environments.

To define the local principal *graaft* to the default key table, issue the **keytab** command, as shown in Figure 3-53.

```

GRAAFF @ SC57:/u/graaft>keytab add paul -p paul
GRAAFF @ SC57:/u/graaft>keytab list paul
Key table: /etc/skrb/krb5.keytab

Principal: paul@KRB390.IBM.COM
Key version: 1
Entry timestamp: 2001/06/08-15:08:12

```

Figure 3-53 *keytab command example to add a principal*

You can now obtain a ticket-granting ticket using the key table instead of RACF. Next, issue the **kinit** command to obtain a ticket-granting ticket using the key table, as shown in Figure 3-54.

```

GRAAFF @ SC57:/u/graaft>kinit -k paul
EUVF06014E Unable to obtain initial credentials.
Status 0x96c73a06 - Client principal is not found in security
registry.

```

Figure 3-54 *kinit -k command example using a key table*

An error indicating that the client principal is not found in the security registry is returned. What really happened here? When you look at the trace of the **kinit** command, it becomes clear, as shown in Figure 3-55.

```

....
kdb_racf_get_principal(): No RACF profile for paul
kdc_as_process_request(): AS_REQ: kdb_get_principal() failed for
paul@KRB390.IBM
kdc_as_process_request(): AS_REQ: KDC error 6 processing request from
paul@KRB390.IBM for krbtgt/KRB390.IBM.COM@KRB390.IBM.COM

```

Figure 3-55 *kinit -k trace example output*

As you can see from the trace output, it states no RACF profile was found for paul. What does this mean? Local Kerberos principals are always defined in RACF, and foreign principals in their respective REALM (KDC). The messages indicate that the Kerberos server tried to map the local principal to a RACF User ID and could not find a local principal named paul.

The next step is to define a RACF User ID with a KERB segment and a KERBNAME of *paul*. We change the RACF User ID GRAAFF to reflect the local Kerberos principal paul, as shown in Figure 3-56.

```
alu graaff kerb(kerbname(graaff) password(xxx) noexpired
```

Figure 3-56 altuser command example to change local principal name

We try to execute the **kinit -k** command again, as shown in Figure 3-57.

```
GRAAFF @ SC57:/u/graaff>kinit -k paul
EUVF06016E Password is not correct for paul@KRB390.IBM.COM.
```

Figure 3-57 kinit -k command example with password failure

So what is wrong with the password now?

The password we use to add the principal has to match the RACF password for the RACF User ID it is mapped to. So, we have to redefine the local principal in the key table using the correct (RACF) password. To redefine the local principal, we have to delete and add the principal as shown in Example 3-1.

Example 3-1 keytab command - redefine the principal

```
GRAAFF @ SC57:/u/graaff>keytab delete paul
GRAAFF @ SC57:/u/graaff>keytab add paul -p racfpw
GRAAFF @ SC57:/u/graaff>klist -k
Key table: /etc/skrb/krb5.keytab

Principal: paul@KRB390.IBM.COM
Key version: 1
```

We can now issue the **kinit -k paul** command again. Example 3-2 shows that we still receive a password error. This is due to the fact that we added the principal with the correct password, but in lowercase.

Example 3-2 kinit -k command (1)

```
GRAAFF @ SC57:/u/graaff>kinit -k paul
EUVF06016E Password is not correct for paul@KRB390.IBM.COM.
```

Again we need to redefine the principal, as shown in Example 3-1, but now add the (RACF) password in *uppercase*. We are able to successfully obtain a ticket-granting ticket, as shown in Example 3-3.

Example 3-3 kinit -k command (2)

```
GRAAFF @ SC57:/u/graaff>kinit -k paul
GRAAFF @ SC57:/u/graaff>klist
Ticket cache: FILE:/var/skrb/creds/krbcred_b2a52720
Default principal: paul@KRB390.IBM.COM

Server: krbtgt/KRB390.IBM.COM@KRB390.IBM.COM
```

3.6 Auditing

SMF Type 80 records are created for login requests (Kerberos initial ticket requests). Both success and failure events can be logged as determined by the `SKDC_LOGIN_AUDIT` environment variable. The event code is 68 and the record includes relocate sections 333 (Kerberos principal name), 334 (request source), and 335 (KDC error code).

The Kerberos principal is stored as a global name (`/.../realm-name/principal-name`) and not as a Kerberos name (`principal-name@realm-name`). This is done to avoid code page problems caused by the at-sign variant character. If the request is received through TCP/IP, the request source is the network address (`nnn.nnn.nnn.nnn:ppppp`). If the request is received through Program Call, the request source is the system user ID of the requester. The KDC error code is a value between 0 and 127.

Figure 3-58 shows the smf records (truncated) generated for the `kinit` commands issued in Example 3-2 on page 115 and Example 3-3 on page 115.

KTICKET	FAILURE	14:19:08	2001-06-15	/.../KRB390.IBM.COM/pau1	GRAAFF	24
.....						
KTICKET	SUCCESS	14:22:42	2001-06-15	/.../KRB390.IBM.COM/pau1	GRAAFF	

Figure 3-58 SMF record examples for `kinit` command

The first record showed an error code of 24, which means that the *preauthentication* (password) failed.

3.7 Windows 2000 use of Kerberos

Kerberos V5 is the default authentication protocol in Windows 2000. This section describes the way Kerberos is used by Windows 2000.

1. The first action occurs when the client types in a password. The Kerberos client on the workstation sends an AS request to the Authentication Service on the KDC asking the Authentication Service to return a ticket to validate the user is who they say they are. The Authentication Service verifies the client's credentials and sends back a ticket-granting ticket.
2. The second action is when the client requests access to a service or a resource by sending a ticket-granting service request to the ticket-granting service on the KDC. The ticket-granting service returns an individual ticket for the requested service that the client can submit to whatever server holds the service or resource the clients wants.
3. The third action is when the Kerberos client actually submits the service ticket to the server and requests access to the service or resource. These are the AP messages.

In Microsoft's implementation, the access security identifiers (SIDs) are contained in the PAC that is part of the ticket sent to the server. This third action need not have a response by the server unless the client has specifically asked for mutual authentication. If the client has marked this exchange for mutual authentication, the server returns a message to the client that includes an authenticator timestamp. For a typical domain logon, all three of these actions occur before the user is allowed access to the workstation.

3.8 Windows 2000 interoperation with OS/390 NAPS

The following steps provide a high-level overview of how to set up a Windows 2000 server to interoperate with the OS/390 Network Authentication and Privacy Services (NAPS). Details of the steps are presented in subsequent sections.

1. Install Active Directory. (Consult the Microsoft website at www.microsoft.com for details about how to install this product.)
2. Install the Windows 2000 DNS server.
 - a. Define DNS service records for the 390 Kerberos services.
 - b. Define DNS text records to map the 390 DNS domain to the 390 Kerberos realm.
3. Set up peer trust between the Windows 2000 Realm and the OS/390 Realm.
4. Define the 390 Kerberos Realm to each Windows 2000 workstation.

3.8.1 Install Windows 2000 DNS server

The normal Windows 2000 server install sets up the active directory and creates SRV records for the `_kerberos` and `_kpasswd` services provided by the Windows 2000 domain controller. However, the server install does not create a TXT record to map host names in the Windows 2000 domain, so you need to create one yourself.

Define DNS Service record

Create SRV records for the `_kerberos` and `_kpasswd` services using the UDP and TCP protocols. This example uses the Windows 2000 DNS management console to create the SRV entries for the `krb390.ibm.com` domain. The SKRBKDC started task will be running on wtsc57 system, so there is a `_kerberos` and `_kpasswd` entry for that system,

The Kerberos runtime randomly selects entries with the same priority when attempting to contact a service provider, so this example uses the same priority for all entries to provide rudimentary load balancing.

Define DNS TXT record

Create a TXT record to map the host name `wtsc57` in the `krb390.ibm.com` DNS domain to the `KRB390.IBM.COM` Kerberos realm. Figure 3-59 shows the Windows 2000 DNS management console to create the TXT record.

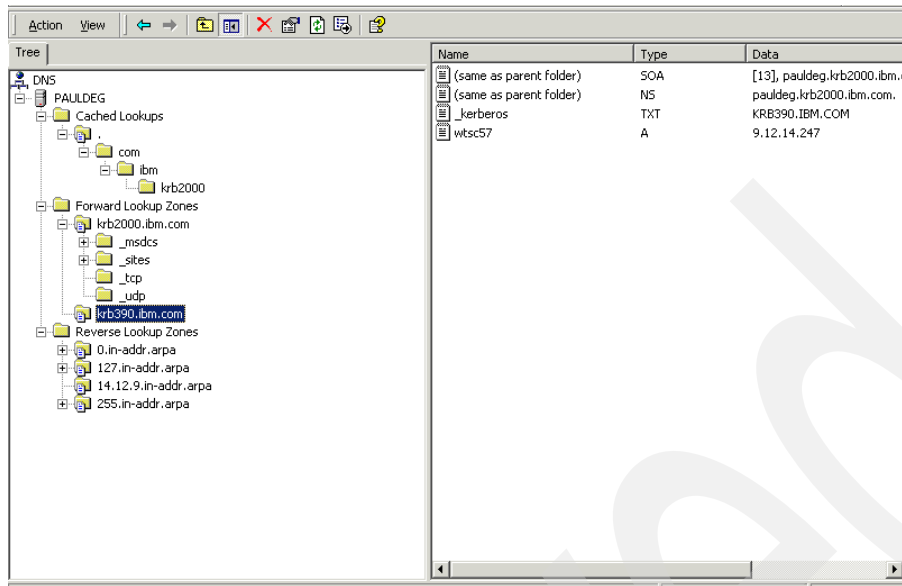


Figure 3-59 DNS window that shows the TXT record

3.8.2 Set up peer trust between the Kerberos realms

Set up a peer-to-peer trust relationship between the KRB390.IBM.COM realm and the KRB2000.IBM.COM realm. This allows clients in the KRB2000.IBM.COM realm to access services in the KRB390.IBM.COM realm and vice versa.

The RACF commands shown in Example 3-3 on page 115 set up the z/OS side of the peer-to-peer trust relationship.

Example 3-4 Peer trust definition for Kerberos realms

```
RDEFINE REALM /.../KRB390.IBM.COM/krbtgt/KRB2000.IBM.COM          KERB(PASSWORD(peerw2kp))
RDEFINE REALM /.../KRB2000.IBM.COM/krbtgt/KRB390.IBM.COM
KERB(PASSWORD(peer390p))
```

Important: The password is case-sensitive on the RACF **RDEFINE REALM** command, while the realm name will be folded to uppercase.

Use the Active Directory Domains and Trusts management console to set up the Windows 2000 side of the peer-to-peer trust relationships, as shown in Figure 3-60 on page 119. Open the **Properties** dialog for the krb2000.ibm.com domain. The password specified for the “Domains trusted by this domain” entry must be the same as the password (peerw2kp) specified on the /.../KRB390.IBM.COM/krbtgt/KRB2000.IBM.COM profile in the REALM class. The password specified for the “Domains that trust this domain” entry must be the same as the password (peer390p) specified on the /.../KRB2000.IBM.COM/krbtgt/KRB390.IBM.COM profile in the REALM class.

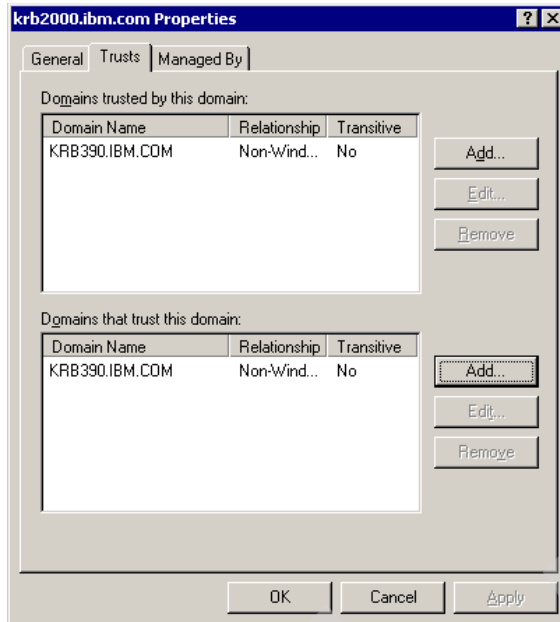


Figure 3-60 Windows 2000 Domains and Trust window

3.8.3 Define the 390 KDC to each Windows 2000 workstation

Define the location of the KRB390.IBM.COM KDC on each Windows 2000 client using the **ksetup** command, as shown in Example 3-5. The **ksetup** command is shipped as part of the Windows 2000 Support Tools on the Windows 2000 CD.

Example 3-5

```
ksetup /addkdc KRB390.IBM.COM wtsc57.krb390.ibm.com
```

This command is effective immediately; however, when you re-boot your system, you will notice that as part of the Windows 2000 login you now have an option to log in to the Kerberos realm KRB390.IBM.COM.

3.9 DB2 Version 7 usage of Kerberos

DB2 Version 7 for OS/390 is the first release to make use of Kerberos technology as a means of authentication. In our test environment we show how Kerberos technology provides “single signon” between a Windows 2000 environment and OS/390. In our example we use DB2 Connect Version 7.1 Fixpak 2 for Windows and DB2 Version 7 on OS/390.

3.9.1 DB2 Connect Version 7.1 setup

After you install DB2 Connect Version 7.1, you need to configure DB2 Connect to connect to an OS/390 DB2 subsystem.

Use the DB2 Client Configuration Assistant to configure DB2 Connect, as shown in Figure 3-61 on page 120.



Figure 3-61 DB2 Connect: Add database wizard (1)

Click the **Add Database** button to start configuring the database you want to connect to. You will then be presented with the Add Database wizard window, as shown in Figure 3-62.

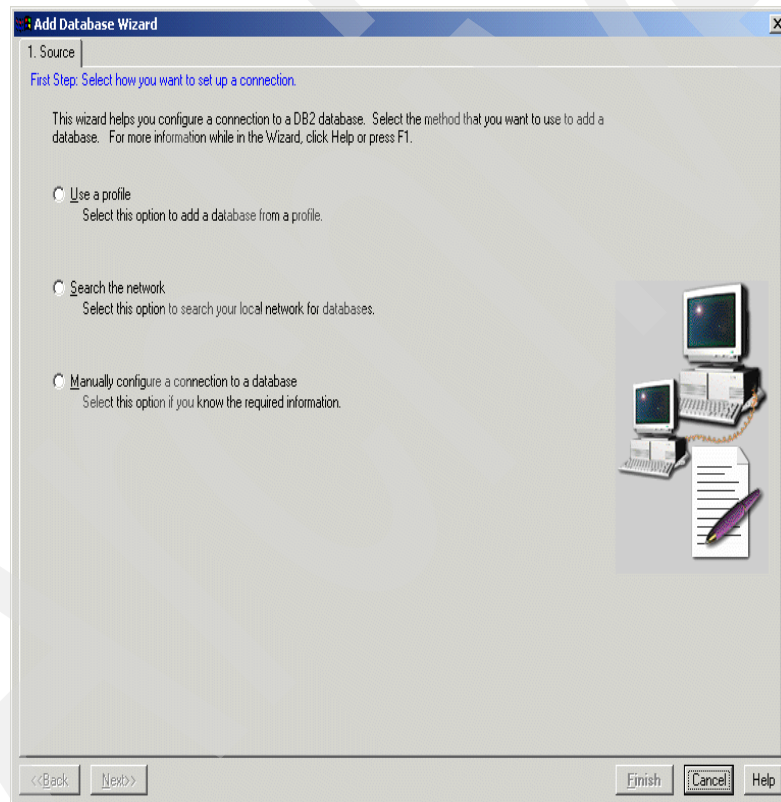


Figure 3-62 DB2 Connect: Add database wizard (1)

Select the option to manually configure a database connection. After selecting this option, two more tabs appear, LDAP and Protocol, as shown in Figure 3-63 on page 121.

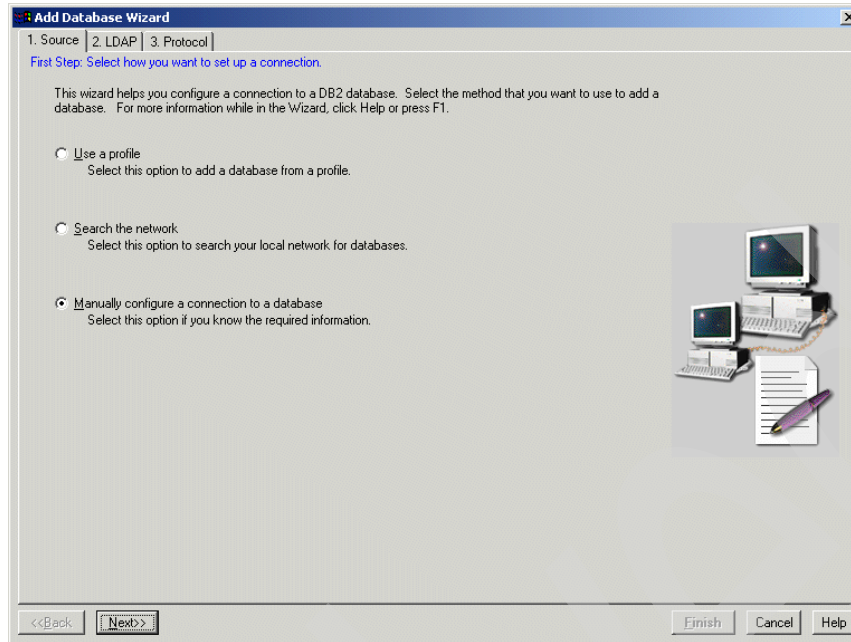


Figure 3-63 DB2 Connect: Add database wizard (2)

You can then select the **Protocol** tab, and skip the LDAP one. Select the protocol you want to use to communicate to DB2 for OS/390. In our case, we selected TCP/IP. You also need to specify that the database resides on a host(390) and that you directly connect to OS/390 and do not use a DB2 gateway, as shown in Figure 3-64.

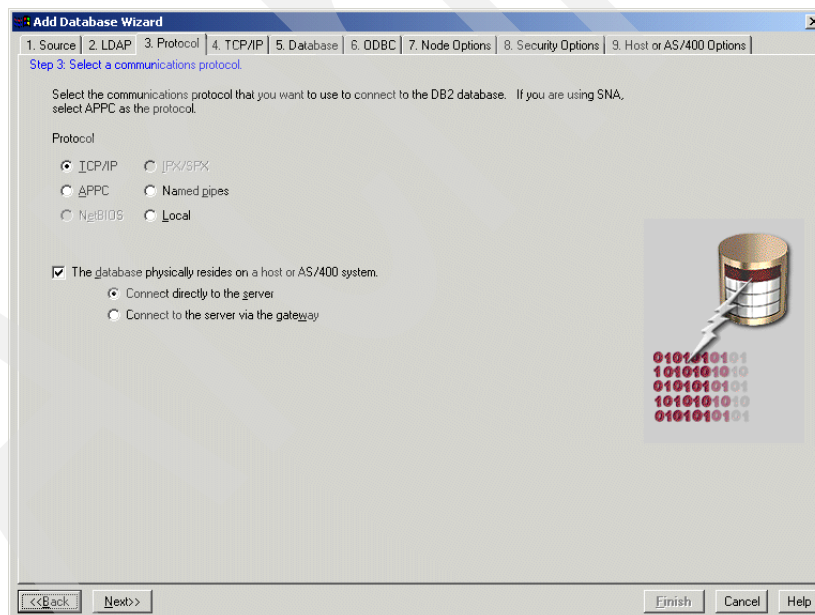


Figure 3-64 DB2 Connect: Add database wizard (3)

When you select these options more selection tabs appear. Tab 4, TCP/IP allows you to specify the 390 hostname, in our case WTSC57.KRB390.IBM.COM, and the port number the DB2 DDF started task uses. The port number is displayed at startup of the DB2 DDF started task, as shown in Figure 3-65 on page 122.

```

DSNL004I  =DB2A DDF START COMPLETE 952
          LOCATION  DB2A
          LU        USIBMSC.SCPDB2A
          GENERICLU -NONE
          DOMAIN    wtsc57.krb390.ibm.com
          TCPPORT    33366
          RESPORT    33367

```

Figure 3-65 DB2 DDF started task messages to indicate TCPPORT number

Figure 3-66 shows the information we used in our example for the TCP/IP tab.

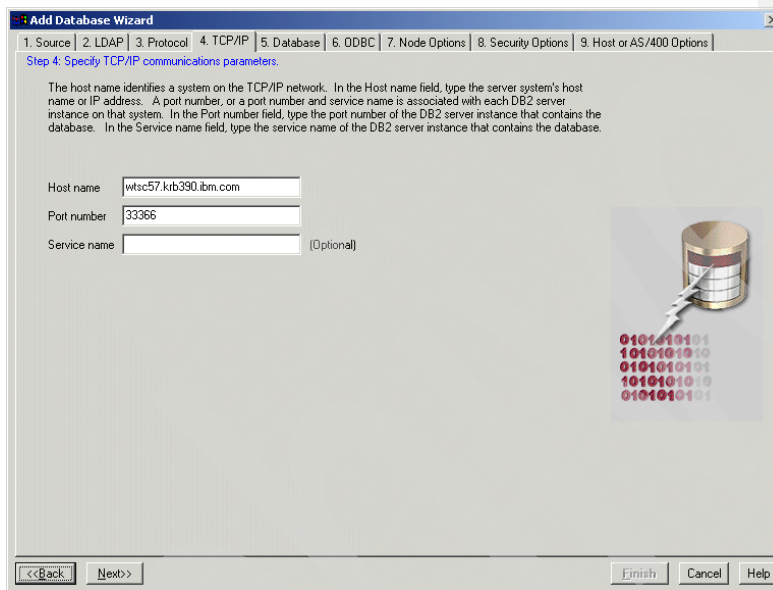


Figure 3-66 DB2 Connect: Add database wizard (4)

Next, select the **Database** tab, which allows you to specify the database name you want to use. For connecting to DB2 for OS/390, it is actually the DB2 location name, which can be displayed using the DSNJU004 utility. Our DB2 subsystem uses the same location name as the subsystem name, which is DB2A, as shown in Figure 3-67 on page 123. We used a database alias of DB2AK to indicate that we use Kerberos as the authentication mechanism.



Figure 3-67 DB2 Connect: Add database wizard (5)

Select the **ODBC** tab. You do not need to make any changes here, as shown in Figure 3-68.

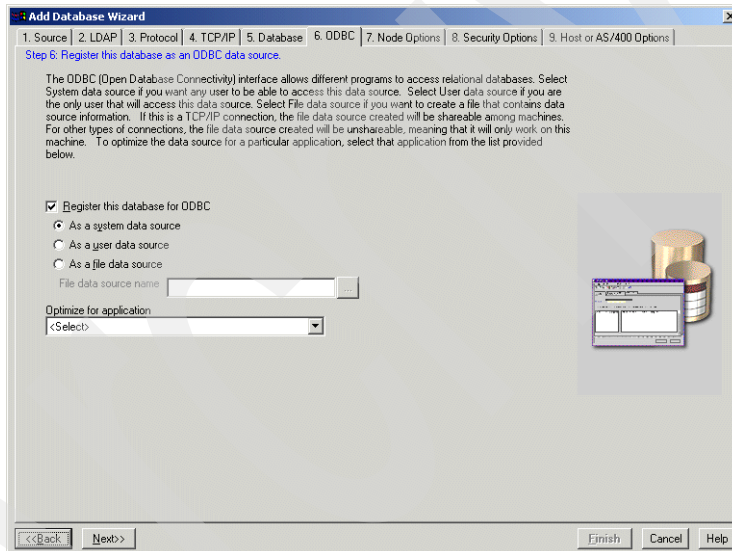


Figure 3-68 DB2 Connect: Add database wizard (6)

The **Node options** tab can be skipped, so can go straight to the **Security** tab to configure the security options you want to use. Select the **Configure Security** option, then select Kerberos as the authentication option. The Kerberos principal name is the name of the local principal name (KERB segment) of the DB2 DDF started task, in our example db2a@KRB390.IBM.COM, as shown in Figure 3-69 on page 124.

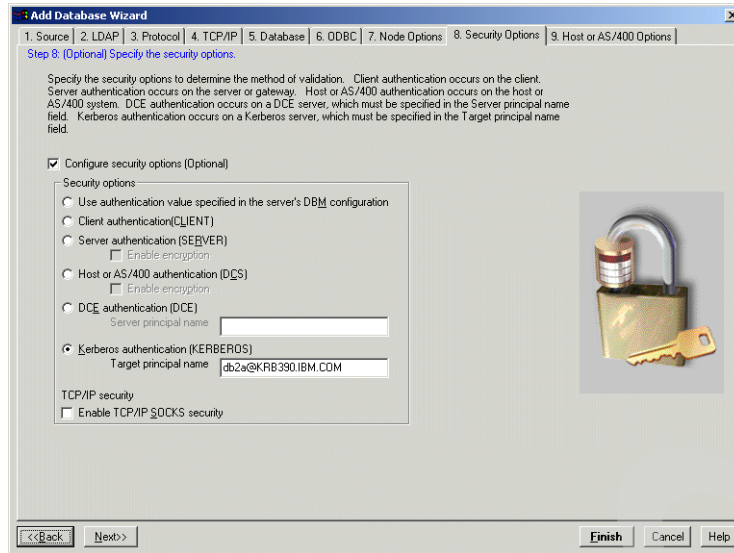


Figure 3-69 DB2 Connect: Add database wizard (7)

When completed, click the **Finish** button to end the configuration process. A conformation window is displayed that shows that the database is configured successfully, as shown in Figure 3-70.

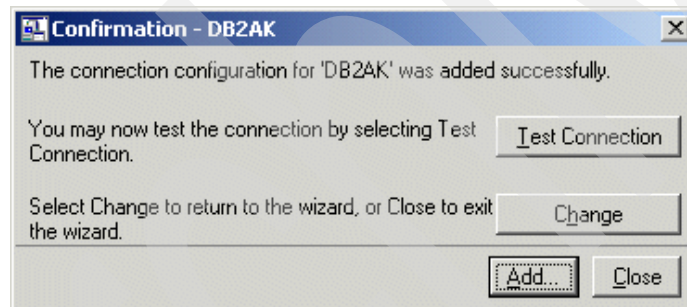


Figure 3-70 DB2 connect: Database configuration confirmation window

When you click **Close**, you see the main window of the DB2 Client Configuration Assistant, as show in Figure 3-71 on page 125, that shows the newly configured database connection.

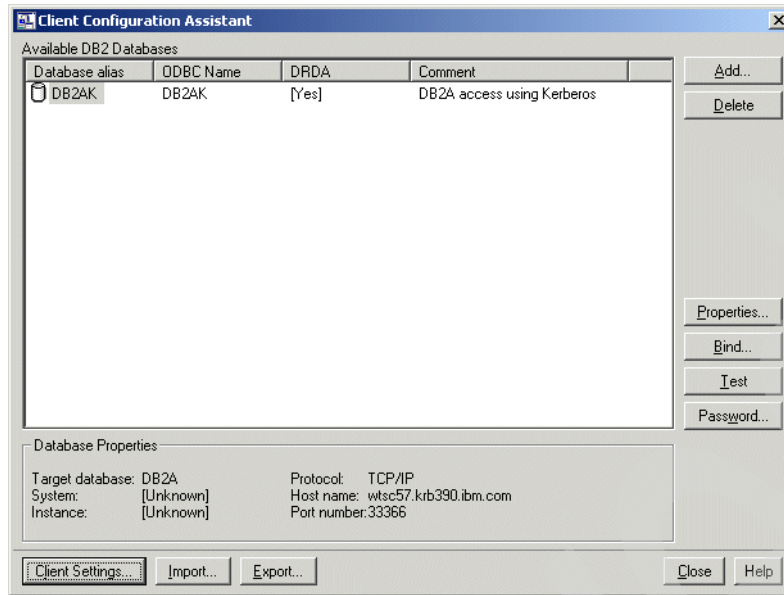


Figure 3-71 DB2 Connect: Configuration Client Assistant window

You can test the database connection using the DB2 Command Center. In our example, by issuing the DB2 connect command to connect to database alias DB2AK, we can see, as shown in Figure 3-72, that we are successfully connected to DB2 using authorization ID GRAAFF, which is how we set up the mapping profile in the RACF KERBLINK class.

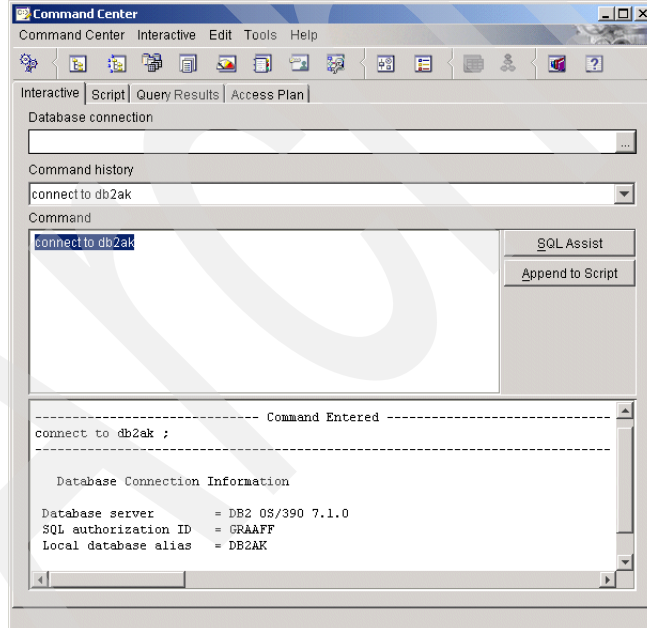


Figure 3-72 DB2 command center window

Archived



Part 3

LDAP and its use with other products

Faint background watermark text: "ARCHIVED"

Archived

Policy Director usage of the LDAP server on z/OS

Lightweight Directory Access Protocol (LDAP) is a technology that can provide directory services to a range of applications. A directory service, a critical part of distributed computing, is the central point where network services, security services, and applications can form an integrated distributed computing environment. The simplicity of LDAP enables users to store and retrieve data easily from the directory.

z/OS and OS/390 are complete software packages for the IBM zSeries and System/390 line of servers, respectively. They offer a range of services necessary for a variety of computing environments. Of particular value are their respective SecureWay security servers. The security server includes an LDAP client and server, and System Authorization Facility (SAF).

For scalability and performance reasons, it is advantageous to use LDAP on z/OS or OS/390 as a user registry for Tivoli Policy Director. Furthermore, by taking advantage of LDAP's native authentication object class, Policy Director can be configured to authenticate against an existing SAF user registry.

This chapter describes the configuration and use of the z/OS LDAP server to show how it can be used by Tivoli's Policy Director.

4.1 Test environment

In our test environment, we are using Policy Director Version 3.8 with FixPack 2 and e-Fix 4, on Windows 2000 with servicepack 4.

The LDAP server is the SecureWay LDAP Server for z/OS Version 1 Release 2 using the TDBM backend.

The Policy Director console is running on a Windows 2000 servicepack 4 workstation.

Note: e-Fix 4 is available for download from the following ftp site:
[ftp://ftp.tivoli.com/support/patches/LA/3.8-P0L-0004E.neobendium servicepack 4](ftp://ftp.tivoli.com/support/patches/LA/3.8-P0L-0004E.neobendium_servicepack_4)

The latest Policy Director FixPacks are available at:
<https://www.tivoli.com/secure/support/patches/>

4.1.1 z/OS LDAP server setup

We applied two PTFs to the system to ensure proper functionality:

► UW99408 (APAR OW50971)

Contains enhancements to the LDAP server for OS/390 V2R10, z/OS V1R1, and z/OS V1R2.

► UW99370 (APAR OW49959)

Contains RACF enhancements to support the SDBM updates from PTF UW99408. There is a prerequisite for this PTF that also needs to be applied to supply SAF support for Policy Director calls. The PTF providing SAF support is UW99372 (APAR OW49960).

We did not use the LDAPCNF utility provided with the z/OS SecureWay Security Server LDAP to set up the LDAP server. For more information on setting up the z/OS SecureWay Security Server LDAP with the LDAPCNF utility, see Chapter 7, “LDAP server on z/OS” on page 219.

Installing the LDAP server SDBM backend

The configuration for the LDAP server can be stored in the HFS file system or in a PDS. We stored the slapd.conf and slapd.envvars in the HFS.

Setting up the RACF environment for the LDAP server

We followed the standard definitions as described in *SecureWay Security Server LDAP Server Administration and Use*, SG24-5923.

There is additional setup needed for the user ID when using SDBM, as shown in Example 4-1.

Example 4-1

```
REDEFINE FACILITY IRR.RUSERMAP UACC(NONE)
PERMIT IRR.REUSERMAP CLASS(FACILITY) ID(GLDSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

We ran the LDAP server as a started task. Once the user ID is created, the GLDSRV started task must be defined. To define the started task for the user ID created, the RACF commands in Example 4-2 are used.

Example 4-2

```
RDEFINE STARTED GLDSRV.** STDATA(USER(GLDSRV))
SETROPTS REACLIST(STARTED) REFRESH
```

The JCL needed to run the LDAP server as a started task is provided with the product as a procedure. This JCL can be found at GLDHLQ.SGLDSAMP on the system where the LDAP server is installed. This procedure can be started in the System Display and Search Facility (SDSF) or from the operator's console, once the sample JCL has been placed into the installation-specific library for procedures (sysm.proclib). The JCL must be tailored before it can be run. A copy of the JCL we ran is included in Example 4-3.

Example 4-3

```
//GLDKARIN PROC REGSIZE=0M,
// PARS='',
// PGLDHLQ='SYS1',
// PDB2HLQ='DSN710',
// OUTCLASS=*
//*-----
//GO EXEC PGM=GLDSLAPD,REGION=&REGSIZE,TIME=1440,
// PARM=('&PARMS >DD:SLAPDOUT 2>&1')
//*-----
//STEPLIBDD DSN=&PGLDHLQ..SGLDLNK,DISP=SHR
// DD DSN=&PDB2HLQ..SDSNLOAD,DISP=SHR
//*-----
//CONFIGDD PATH='/usr/local/karin/ldap/slapd.conf'
//ENVVARDD PATH='/usr/local/karin/ldap/slapd.envvars'
//*-----
//SLAPDOUT DD SYSOUT=&OUTCLASS
//SYSOUT DD SYSOUT=&OUTCLASS
//SYSUDUMP DD SYSOUT=&OUTCLASS
//CEEDUMP DD SYSOUT=&OUTCLASS
```

Configuring the LDAP server for SDBM

The slapd.conf and slapd.envvars need to be copied from the directory in which they are installed to the directory in which they are used. We copied the files into the /etc/ldap directory, as shown in Example 4-4.

Example 4-4

```
# cp /usr/lpp/ldap/etc/slapd.conf /etc/ldap/ .
# cp /usr/lpp/ldap/etc/slapd.envvars /etc/ldap/ .
```

The SDBM schema files containing the RACF information need to be copied from the /usr/lpp/ldap/etc directory into the /etc/ldap directory, as in Example 4-5.

Example 4-5

```
# cp /usr/lpp/ldap/etc/schema.*.at /etc/ldap/ .
# cp /usr/lpp/ldap/etc/schema.*.oc /etc/ldap/ .
```

The slapd.at.racf configuration file contains the LDAP server RACF attribute type definitions, including the RACF and Kerberos attributes. The IBM schema configuration file schema.IBM.at also contains these RACF and Kerberos attribute type definitions.

We used the IBM schema files. In this case, the attribute types and object classes needed to support RACF access are contained in schema.IBM.at and schema.IBM.oc respectively. When using the IBM schema files, slapd.at.racf and slapd.oc.racf files should not be included by the LDAP server configuration file. Instead, the schema.IBM.at and schema.IBM.oc files should be included. The schema.system.at, schema.system.oc, schema.user.at and schema.user.oc schema files must also be included in the slapd.conf.

The slapd.conf include statements for SDBM should be similar to Example 4-6.

Example 4-6

```
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc
```

Note: The slapd.conf and slapd.envvars files can be edited using either **oedit** or **vi**. The schema files should not be modified. They document the interface to RACF through SDBM.

The slapd.conf file must be modified to include global parameters and SDBM-specific parameters. Our slapd.conf for the SDBM backend is shown in Example 4-7.

Example 4-7

```
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc
# ----- global parameters
listen ldap://4389
security none
adminDN "cn=Admin"
# ----- SDBM parameters
database sdbm GLDBSDBM
suffix "racfdb=local"
```

After configuring the slapd.conf, we started the LDAP server in SDSF with the command:

```
/s GLDSRV
```

When the LDAP server has been started and is ready, the following message is displayed:

```
GLD00122I SLAPD is ready for requests
```

Configuring the LDAP server for TDBM

When the LDAP server is run with this configuration, there are some DB2 requirements, as follows:

- ▶ There must be a DB2 subsystem up and running; this section does not cover the steps to get this subsystem up and running.
- ▶ Some DB2 tablespaces and tables must be set up for the LDAP Server; this section describes the allocation steps for these items.
- ▶ The Call Level Interface (CLI) must be set up; this section does not go through the setup of ODBC or CLI, but does show the JCL to connect the LDAP server to the CLI plan.

When using TDBM, the LDAP server DB2 database must be created by running two SPUFI scripts. Sample SPUFI scripts are provided. The SPUFI scripts for creating the database and table spaces can be found in GLDHLQ.SGLDSAMP(TDBMDB) and the script for creating the table indexes can be found in GLDHLQ.SGLDSAMP(TDBMINDX).

An example of the commands we ran are shown in Example 4-8.

Example 4-8

```
CREATE DATABASE PDLAP STOGROUP PDSTOGRP
...created tablespaces here...
...created table indices here...
GRANT EXECUTE ON PLAN DSNACLI TO CBLDAP
GRANT SELECT ON SYSIBM.SYSCOLUMNS TO CBLDAP
GRANT DBADM ON DATABASE PDLAP TO CBLDAP
```

Note: Be sure to run the scripts under a user ID with DB2 SYSADM authority in order to create the database and tablespace, and grant resource authorization.

The CLI is used by the LDAP server to request services from DB2. In order to use the CLI, the job DSNHLQ.SDSNSAMP(DSNTIJCL) must be run. This establishes the environment needed for the LDAP server to use the CLI. It is often referred to as “binding the CLI plan.”

The DB2 CLI initialization file must be created. A sample of the CLI initialization file can be found at DSNHLQ.SDSNSAMP(DSNAOINI). We created the file in the HFS. The format of the CLI file should be the same in either a PDS or an HFS file.

Example 4-9 shows a copy of the dsnaoini.ini file we used. In our example, RDBNDBOT is the location name and DB1T is the subsystem name.

Example 4-9

```
;Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=DB1T

;Example SUBSYSTEM stanza for your DB2 subsystem name
[DB1T]
MVSATTACHTYPE=CAF
PLANNAME=DSNACLI

;Example DATA SOURCE stanz for your data source
[RDBNDBOT]
AUTOCOMMIT=0
CONNECTTYPE=2
```

Note: The LDAP server can be set up to use either the Call Attachment Facility (CAF) or the Recoverable Resource Manager Services Attachment Facility (RRSAF) to access DB2. See *DB2 for OS/390 and z/OS: ODBC Guide and Reference*, SC26-9941 for more information about these choices.

To get the DB2 server location, use the DSNJU004 program supplied by DB2. This will produce a report that contains the DDF record information required by the LDAP server. The location is at the bottom of the output report.

The slapd.conf must be updated to support the TDBM backend. Detailed descriptions of the database parameters are in the *SecureWay Security Server LDAP Server Administration and Use*, SG24-5923. The DB2 parameters that must be added to slapd.conf for TDBM are briefly described here.

servername	DB2 server location name. This is the DB2 DDF name and must match a DATA SOURCE stanza in the DSNAINI file.
databasename	The name of the database this backend will use to store directory data.
dbuserid	A z/OS user ID that will be the owner of the tables.
suffix	This denotes the root of a subtree in the namespace managed by this server within this backend. The suffix is used to indicate the DNs that are within the LDAP directory for this server. In our example, the suffix is o=mopldap.
dsnaoini	This defines the location of the dsnaoini.ini file. In our example, the file is stored in the HFS.

Example 4-10 shows the slapd.conf file that we used. Both the SDBM and the TDBM backends are defined in our example.

Example 4-10

```
listen ldap://:4389
security none
# ----- SDBM backend
database sdbm GLDBSDBM
suffix "racfdb=local"
# ----- TDBM backend
database tdbm tdbm GLDBTDBM
suffix "o=mopldap"
servername RDBNDBOT
databasename PDLAP
dbuserid CBLDAP
dsnaoini /usr/local/karin/ldap/dsnaoini.ini
```

The schema files schema.IBM.ldif and schema.user.ldif contain the objects and attributes used to organize data for the TDBM backend. Each schema file must be modified to match the organization DN suffix in the LDAP configuration file. There is a single line describing the DN of the schema to be updated.

The line in schema.user.ldif and schema.IBM.ldif that we modified is:

```
dn: cn=schema, o=mopldap
```

The schema files must be loaded via an **ldapmodify** command, as follows:

```
ldapmodify -h <hostname> -p <port> -D <bindDN> -w <bind password> -f <schema file>
```

The file schema.IBM.ldif must be loaded before schema.user.ldif. It is not necessary to reload the schema for each suffix DN configured.

For more information regarding the installation of LDAP, see *SecureWay Security Server LDAP Server Administration and Use*, SG24-5923.

Enabling native authentication

LDAP has the ability to authenticate to the Security Server through TDBM by supplying a Security Server password on a simple bind to a TDBM backend. There are two LDAP operations affected by native authorization: bind and password modify.

We defined the ou=native subtree to the o=mopldap tree in order to test the modification. We chose to define useNativeAuth to selected. In this case, only selected users defined in ou=native, o=mopldap will be using native authentication.

Example 4-11 shows the modifications we made to the slapd.conf file to enable native authentication.

Example 4-11

```
# ----- TDBM native auth
useNativeAuth selected
nativeAuthSubtree "ou=native, o=mopldap"
nativeUpdateAllowed yes
```

Restart the LDAP server for the configuration modifications to take effect. The LDAP server job log will contain the following line:

The useNativeAuth configuration option SELECTED has been enabled.

The ou=native,o=mopldap subtree must be defined to the LDAP server before any entries can be added that will take advantage of native authentication.

Example 4-12 shows the process we used to define the ou=native,o=mopldap subtree.

Example 4-12

```
ldapmodify -h <hostname> -p <port> -D "cn=LDAP admin,o=mopldap" -w <password> -f
nativeou.ldif
where nativeou.ldif contains the following:
dn: ou=native, o=mopldap
objectclass: top
objectclass: organizationalUnit
ou: native
```

Now users can be added to the LDAP directory. We used the **ldapadd** command:

```
ldapadd -h <hostname> -p <port> -D "cn=LDAP admin,o=mopldap" -w <password> -f native.ldif
```

to add users to the TDBM backend. Example 4-13 shows an excerpt of the ldif file we used to add the users.

Example 4-13

```
dn: cn=Karin Good, ou=native, o=mopldap
objectclass: person
objectclass: inetOrgPerson
objectclass: ibm-nativeAuthentication
objectclass: organizationalPerson
cn: Karin Good
sn: Good
ibm-nativeId: KARIN

dn: cn=Jack NotYet, ou=native, o=mopldap
objectclass: person
objectclass: inetOrgPerson
objectclass: ibm-nativeAuthentication
objectclass: organizationalPerson
```

```
cn: Jack NotYet
sn: NotYet
ibm-nativeId: JACNOX

dn: cn=Jack Bad, ou=native, o=mopldap
objectclass: person
objectclass: inetOrgPerson
objectclass: ibm-nativeAuthentication
objectclass: organizationalPerson
cn: Jack Bad
sn: Bad
ibm-nativeId: JACBAD
```

The userids must also be defined in RACF, where the RACF ID is the value of the `ibm-nativeId` attribute defined in LDAP.

Note: The native authentication users can be defined in LDAP before they are created in RACF. However, the users must be defined in RACF before using native authentication.

For more information regarding native authentication, see *SecureWay Security Server LDAP Server Administration and Use*, SG24-5923.

LDAP customization for Policy Director

There are a couple of configuration changes needed to customize the zSeries LDAP server for Policy Director. They are:

- ▶ Configure a base suffix for Policy Director.
- ▶ Configure the LDAP server with `secAuthority=Default`.

Note: We used a dedicated LDAP server for Policy Director. The base suffix that we defined for Policy Director was used for the global sign-on root in the Policy Director configuration.

The `secAuthority=Default` suffix must be configured in the `slapd.conf` file. The LDAP server must be restarted for the modification to take effect.

Example 4-14 shows the `slapd.conf` we used to customize the z/OS LDAP server for Policy Director.

Example 4-14

```
security none
listen ldap://5389
adminDN "cn=root"
#----- SDBM config
database sdbm GLDSDBM
suffix "racfdb=local"
#----- TDBM config
database tdbm GLDTDBM
suffix "o=ibm,c=us"
suffix "secAuthority=Default"
servername RDBNBOT
databasesname PDLAP
dbuserid CBLDAP
sdnaoini /usr/local/ldap/dsnaoini.ini
```

We ran the **ldapsearch** command, shown in Example 4-15, to ensure the change was successful.

Example 4-15

```
ldapsearch -h 9.100.203.110 -p 3389 -V 3 -b "" -s base "objectclass=*"
```

The **ldapsearch** command returns the base information about the LDAP server, as shown in Example 4-16.

Example 4-16

```
supportedcontrol=2.16.840.1.113730.3.4.2
supportedcontrol=1.3.18.0.2.10.2
namingcontexts=o=IBMMOP
namingcontexts=secAuthority=Default
namingcontexts=sysplex=portinfo
subschemasubentry=CN=SCHEMA, o=IBMMOP
supportedsaslmecanism=EXTERNAL
supportedldapversion=2
supportedldapversion=3
ibmdirectoryversion=z/OS V1R2
```

The definition of `secAuthority=Default` will be added later.

4.1.2 Policy Director Setup

Policy Director version 3.8 supports the z/OS LDAP server as the user registry.

Note: In our environment, we did a complete native installation of Policy Director on the Windows 2000 workstation. In setting up this environment, we relied heavily upon the Policy Director version 3.8 Cookbook. Once we had a working environment on Windows 2000, we configured Policy Director to point to the z/OS LDAP server as the user registry.

Note: For more details, see the Policy Directory version 3.8 Cookbook. This cookbook can be obtained by registered Policy Director users at the following Web site:

http://www.tivoli.com/secure/support/downloads/secureway/policy_dir/pd3.8/tpd3_8_install_guide.html

Policy Director prerequisites include the IBM Global Security Toolkit (GSKit), an HTTP server, DB2, and an LDAP server. All of the prerequisite software is included on the CDs shipped with Policy Director version 3.8.

We installed the GSKit version 4.0.3.168 and the IBM HTTP server version 1.3.12 on the Windows 2000 workstation.

We chose to install DB2 independent of the IBM SecureWay Directory code. We did this in an attempt to control each piece of software. In case of an installation or configuration failure, we were able to pinpoint the issue more quickly.

Note: Read the LDAP server installation section of the Policy Director version 3.8 cookbook before installing DB2. There are a couple of caveats to look out for:

- ▶ If DB2 exists on the system, uninstall it.
- ▶ If the folder `\ldapdb2` exists on the system, delete it.
- ▶ If the userids `db2admin` and/or `ldapdb2` exist on the system, delete them.

To check if the userids exist, right-click **My Computer -> Manage**. This will display the Computer Management Window. From the left frame, select **Local Users and Groups -> Users**. This will display the userids that exist on the system.

Following a successful installation of DB2, restart the workstation.

The IBM SecureWay LDAP server and client can now be installed. When the installation is started, a window will display the installed components. The GSKit, IBM HTTP server, and DB2 should all be listed. Since we had all the prerequisites installed, we selected to install only the LDAP server and LDAP client.

Following a successful installation of the Directory server and client, restart the workstation.

Before configuring the Directory server, check to be sure the following services are running. (Right-click **My Computer -> Manage**. From the left frame of the Computer Management window, click **Services** to display the active services.)

- ▶ DB2 - DB2
- ▶ DB2 - DB2DAS00
- ▶ IBM HTTP Server

To begin the Directory server configuration utility, click **Start -> Programs -> IBM SecureWay Directory -> Directory Configuration**. The directory server configuration utility requires you to supply an administrator dn and password; in our case the values were cn=root and secret.

The directory server configuration utility will display a couple more panels for you to provide input, or take the defaults. Then a configuration summary screen will be displayed. If you are satisfied with your input, continue.

At this point, the DB2 tables are configured for the environment. A command prompt window will display the progress and will indicate success or failure.

Note: If you are installing WEBSeal, it is recommended that you modify the port that the HTTP server listens on to avoid conflicts. The HTTP server port value will be listed in the httpd.conf file.

In order to complete the configuration, suffixes need to be added to the directory. We used the Web Administration tool to do this. To start the Web Admin Tool, point a browser at `http://hostname:portnumber/ldap/index.html`. In our case, this was `http://127.0.0.1:389/ldap/index.html`.

We supplied our administrator dn and password (cn=root and secret) to get access to the server. The IBM SecureWay Directory Administration panel is displayed. At the top of the screen, it will indicate that you must add suffixes.

Click **Add Suffixes** and enter secAuthority=Default in the suffixDN field. To add the suffix, click **Update**. The server must be restarted for the suffix to be recognized.

We also added a suffix for the Policy Director users and Global Sign On. We added the suffix o=ibm,c=us and restarted the server again for the suffix to be recognized.

The appropriate suffixes have been added and now need to be defined. We used the Directory Management Tool to define the suffixes.

Click **Start -> Programs -> IBM SecureWay Directory -> Directory Management Tool** to begin. In order to perform modifications and additions, you must rebind to the LDAP server with the admin dn and password. To rebind, click **Rebind** (listed under Servers in the left panel). In the Rebind to Server dialogue, select **Authenticated**. Then supply the admin dn and password to bind to the server.

Once the bind is successful, the Browse directory tree panel will be displayed. A couple of messages may appear on the screen indicating that certain entries do not contain data.

Click **Add** in the upper right hand frame to add an entry. Define/Add the suffix previously configured for the Policy Director users and Global Sign On. In our case, this was o=ibm,c=us.

Note: When this suffix is defined, a warning may be displayed indicating that the suffix secAuthority=Default does not contain any data. Click **OK** to dismiss this message.

Once the suffix is added, the LDAP configuration is complete. Restart the workstation.

We now began the installation of Policy Director. We chose to install Policy Director RunTime Environment (PDrte), Policy Director Management Server (PDMgr), and the Policy Director Authorization Server (PDAdl). From the Policy Director CD, you can select to install all three of these packages at one time.

Note: Do *not* restart the system if you intend to install WEBSeal. Once the installations of PDrte, PDMgr and PDAdl are complete, insert the WEBSeal CD and install WEBSeal before configuring the other components.

When all the components have been installed, restart the workstation.

To begin configuring the Policy Director components, click **Start -> Programs -> Policy Director -> Configuration**. The Configuration Window is displayed. The configuration tool will force you to configure the components in the correct order.

We followed the configuration instructions in the Policy Directory version 3.8 Cookbook when configuring the Policy Director components.

We used the following values to configure the Policy Director RunTime Environment:

LDAP Host Name	127.0.0.1
LDAP Port Number	389
LDAP DN for GSO Database	o=ibm,c=us

We chose not to enable SSL communication with the LDAP server.

We used the following values to configure the Policy Director Management Server:

LDAP Administrator Name	cn=root
LDAP Administrator Password	secret
Policy Director Administrator Password	secret
SSL listening port number	7135
SSL Certificate lifetime	365
SSL connection timeout	7200
Enable root CA Certificate download	no

A message will be displayed when the configuration of PDMgr is successful.

Using z/OS LDAP as the Policy Director user registry

We still need to point Policy Director at the z/OS LDAP server as the user registry. There are a few tasks we need to complete in order to do this. Specifically, we have to:

- ▶ Export LDAP definitions from the Windows 2000 LDAP server to the z/OS LDAP server.
- ▶ Update pd.conf to point to the z/OS LDAP server.
- ▶ Update ldap.conf to point to the z/OS LDAP server.
- ▶ Update user password information.
- ▶ Update ACLs.
- ▶ Turn off the LDAP server on Windows 2000.

We opened the LDAP Server WebAdministration Tool and selected the **db2ldif** utility. We exported the ldif file (**export.ldif**) containing the LDAP definitions for Policy Director to a directory on the workstation. Then we ftp'ed the file to the z/OS system.

We used the **ldapadd** command to update the z/OS LDAP server with the Policy Director definitions.

```
ldapadd -h 9.100.203.110 -p 5389 -D "cn=root" -w secret -c -v -f export.ldif
```

See the *SecureWay Security Server LDAP Server Administration and Use Guide* SC24-5923-02 for more information on the **ldapadd** command and other LDAP utility commands.

The **ldapadd** command should write messages to the screen indicating that entries are being created or modified in the LDAP directory.

The pd.conf and ldap.conf files on the Windows 2000 workstation need to be updated to point to the z/OS LDAP server.

In the pd.conf file, the following lines should be updated with your z/OS LDAP information. The definitions we used are shown in Example 4-17.

Example 4-17

```
user-reg-server=9.100.203.110
user-reg-host=9.100.203.110
user-reg-hostport=5389
```

The following lines in the ldap.conf files need to be updated with your z/OS LDAP information. The definitions we used are shown in Example 4-18.

Example 4-18

```
host=9.100.203.110
port=5389
```

The user passwords for

```
cn=ivmgrd/master,cn=SecurityDaemons,secAuthority=Default
```

and

```
cn=SecurityMaster,secAuthority=Default
```

must be updated in the z/OS LDAP server. These values will not have been set properly in the export.ldif. The randomly generated ivmgrd password can be found on the Windows 2000 workstation in the ivmgrd.conf file.

```
bind-dn=cn=ivmgrd/master,cn=SecurityDaemons,secAuthority=Default
bind-pwd=<randomly generated password>
```

We modified the password using an ldif file. Example 4-19 shows the ldif file we used.

Example 4-19

```
dn: cn=ivmgrd/master,cn=SecurityDaemons,secAuthority=Default
changetype: modify
replace: x
userpassword: <randomly generated password form the ivmgrd.conf file>
-
```

Note: In this ldif file, the (-) is required on the final line.

We were able to choose any password for the cn=SecurityMaster,secAuthority=Default entry. Example 4-20 shows the ldif file we used for the modification.

Example 4-20

```
dn: cn=SecurityMaster,secAuthority=Default
changetype: modify
replace: x
userpassword: secret
-
```

To apply the above changes, we used the **ldapmodify** command.

```
ldapmodify -h 9.100.203.110 -p 5389 -D "cn=root" -w secret -f <ldif file>
```

The **ldapmodify** command must be performed for each ldif file.

We next modified the LDAP ACLs for secAuthority=Default and the user registry suffix, in our environment o=ibm,c=us. We used ldif files to perform the modifications. Example 4-23 shows the ldif files we used.

Example 4-21

```
dn: secAuthority=Default
changetype: modify
replace: x
aclentry:group:cn=remote-acl-users,cn=SecurityGroups,secAuthority=Default:object:ad:normal:
rsc
aclentry:group:cn=ivacld-server,cn=SecurityGroups,secAuthority=Default:object:ad:normal:rsc
aclentry:goup:cn=SecurityGroups,secAuthority=Default:object:ad:normal:rwsc:sensitive:rwsc:c
ritical:rwsc:restricted:rwsc:system:rwsc

and

dn: o=ibm,c=us
changetype: modify
replace: x
aclentry:group:cn=remote-acl-users,cn=SecurityGroups,secAuthority=Default:object:ad:normal:
rsc
aclentry:group:cn=ivacld-server,cn=SecurityGroups,secAuthority=Default:object:ad:normal:rsc
aclentry:goup:cn=SecurityGroups,secAuthority=Default:object:ad:normal:rwsc:sensitive:rwsc:c
ritical:rwsc:restricted:rwsc:system:rwsc
```

We turned off the LDAP server running on the Windows 2000 workstation from the Computer Management Window (right-click **My Computer** -> **Manage** -> **Services** to display the services that are running).

We then restarted the PDMgr from the Computer Management Window, by selecting Policy Director Management Server, stopping it and restarting it.

IBM Host On-Demand Version 6

The browser-based access of IBM Host On-Demand Version 6 gives you a simple way to reach critical host data, without requiring you to install any software on your workstation. Host On-Demand uses the power of Java technology to open the doors to your host system whenever you need it, anywhere you need it, directly from your browser. Just click on a hyperlink to launch the Host On-Demand Java applet. This Web-to-host connectivity solution provides secure Web browser access to host applications and system data through Java-based emulation, so you can take existing host applications to the Web without programming.

Using Secure Sockets Layer (SSL) Version 3.0, Host On-Demand extends secure host data access across intranets, extranets, and the Internet. Mobile workers access a secure Web site, receive authentication and establish communication with a secure enterprise host. With client and server certificate support, Host On-Demand can present a digital certificate (X.509, Version 3) to the Telnet server—such as IBM Communications Server for NT v6 or later, or IBM Communication Server for OS/390 V2R8 or later—for authentication. Host On-Demand can also integrate the SSL client authentication. This allows you to benefit from industry standard public key infrastructure (PKI) methods.

In this chapter we describe the security features provided by the IBM Host on-Demand (HoD) Version 6 product in detail. The security features covered include:

- ▶ LDAP native authentication
- ▶ Telnet negotiated security
- ▶ Express logon

For more general information on Host On-Demand V6, refer to the documentation on the Host On-Demand product website as well as the redbook *Host On-Demand V6*, SG24-6182.

5.1 LDAP native authentication

User ID and password management has become an increasingly important issue for users as the number of systems and applications that require authentication continues to grow. In order to save a user's preferences at the Host On-Demand server, a user ID is required to uniquely identify the user. This ID is used as the index under which the user's preferences are stored in the repository. Host On-Demand does not require passwords to be implemented with the user ID; however, most customers implement a password for an additional level of identification. Because of the platform-independent nature of Host On-Demand, this user ID and password management as implemented by Host On-Demand is independent of any other user ID and password management system.

Host On-Demand Version 6 still requires the administrator to define and manage user IDs when a registered user model is implemented, but with the introduction of the native authentication component we allow the administrator to associate the Host On-Demand user ID with a user ID and password on the native operating system. The native platform authentication service allows users to log on to Host On-Demand using the same password as they would to log on to the operating system (Windows NT, AIX or OS/390) where Host On-Demand is active.

When a user logs on to Host On-Demand, their password is validated against the system password, rather than a separate Host On-Demand password, thus providing the customer with the following benefits:

- ▶ Reducing the number of passwords that the end user must remember.
In many cases this means that the user will have only one password to remember.
- ▶ Better security, and a reduction in the administrative workload for the Host On-Demand administrator by delegating password management to an administrative system that can implement a password management policy that typically includes:
 - Enforcement of password rules
 - Enforcement of password expiration times
 - Ability to revoke access by invalidating a password

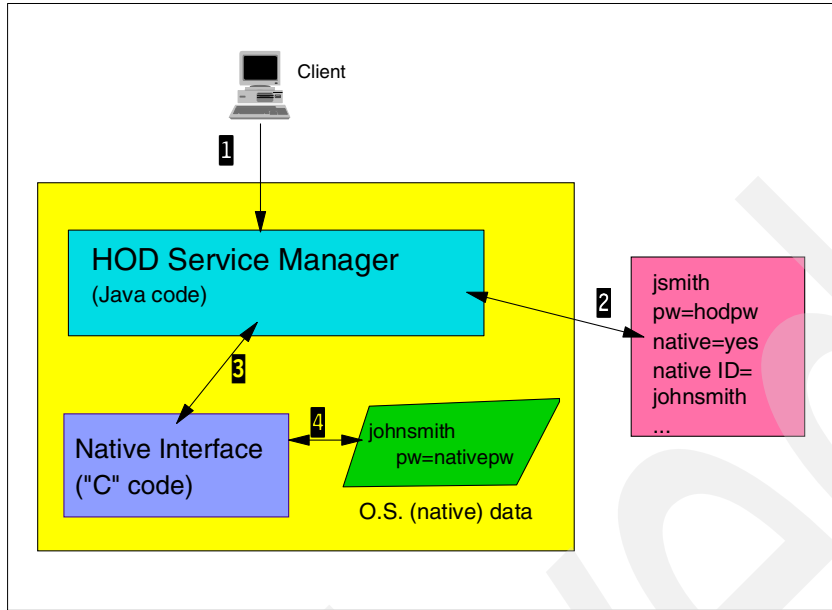


Figure 5-1 Native authentication log-in flow

When a user logs on (1) the user ID and password are sent to the Host On-Demand Service Manager. The Service Manager sends a request for logon information about the user to the LDAP server (2). The LDAP server returns a message indicating if the user is configured for native authentication. If the user is not configured for native authentication, the password stored in the LDAP directory server is returned to the Service Manager. If the user is configured for native authentication, the native ID stored in the LDAP directory is returned, along with an indicator that native authentication should be invoked.

The Service Manager checks the returning information from the LDAP directory server. If the user is configured to use native authentication, the Service Manager sends the user ID and the password to be authenticated to a Host On-Demand module written in C and compiled for the specific operating system (3). That module will invoke the appropriate native operating system security call to validate the user ID and password combination (4). If the user is not configured for native authentication, the Service Manager compares the password that was entered by the user with the password returned by the LDAP server. If the user ID and password are successfully validated by the operating system, processing continues.

All other returns will result in an invalid password error message. Other than a legitimately invalid password, one of the most common reasons for this return message is an expired password. There is no mechanism within Host On-Demand to intercept an expired password and prompt for a new one. The user will be required to correct this condition via some other interface and then log on again to Host On-Demand.

5.1.1 Native authentication requirements

Native platform authentication service must be installed on a Windows NT, AIX, or OS/390 Host On-Demand server. On the Host On-Demand server, LDAP directory services must be enabled and configured for native authentication individually for each user that is to use native authentication.

The LDAP directory server may reside anywhere in the network and may run on any platform. Native authentication requires TDBM or RDBM, which means that DB2 is the backend where the information is retrieved from. The LDAP entries you need to get from the LDAP administrator are shown in Figure 5-2 on page 148. Even though most setups still require you to install the schema files which are shipped with Host On-Demand, we didn't need to do this since the schemas were shipped in the schema.IBM.Idif file which we used in our LDAP setup. For more information on the LDAP server setup, refer to Chapter 7, "LDAP server on z/OS" on page 219.

Follow these steps to use native platform authentication with Host On-Demand:

1. Enable OS/390 users for native authentication.
2. Start the native platform authentication service.
3. Configure current users for native authentication.

5.1.2 Installation of native authentication

The files to support native platform authentication are installed with the Host On-Demand server during the installation process.

The native authentication code runs as a separate executable module called HODRAPD, which is invoked using the hodrapd.sh script. The HODRAPD module is installed during SMP/E CALLLIBS processing and it is automatically link-edited during the JCLIN CALLLIBS processing in the APPLY process.

When the Native Platform Authentication Service is started from UNIX System Services, the HODRAPD module is executed from SYS1.LINKLIB or your alternate LINKLIB data set. If you choose to move the HODRAPD module to an alternate LINKLIB data set, that data set must be accessed by the system LNKLIST or LPALIB.

During installation of Host On-Demand Version 6, the hod60mvs.sh shell script not only untars the Host On-Demand Version 6 product, it also creates the necessary link so that when the user starts native authentication with hodrapd.sh, the HODRAPD load module is executed. If the link to HODRAPD gets lost, the statements in Example 5-1 can be used to restore the link.

Example 5-1

```
export HOD_DIR=/usr/lpp/HOD
touch $HOD_DIR/hostondemand/private/HODRAPD
ln -s $HOD_DIR/hostondemand/private/HODRAPD (continued on next line)
$HOD_DIR/hostondemand/private/hodrapd
chmod 744 $HOD_DIR/hostondemand/private/HODRAPD
chmod +t $HOD_DIR/hostondemand/private/HODRAPD
```

The native authentication code logs its messages to the syslog, which may need to be configured to log the desired level of messages. The hodrapd module writes its messages to the user.* entry in the syslog file.

5.1.3 Starting and stopping native authentication services

To start the native authentication code, run hodrapd.sh (located in the /usr/lpp/HOD directory). The shell script must be started by a user with root authority. It is recommended that you start HODRAPD as a started task from the operator console rather than from an OMVS session. Example 5-2 on page 147 shows the JCL procedure we used to start HODRAPD.

Example 5-2 HODRAPD sample JCL procedure

```
//HODRAPD PROC
/* Function: IBM WebSphere Host On-Demand V6 Native Authentication
//HODSRVG EXEC PGM=BPXBATCH,REGION=OM,TIME=NOLIMIT,
// PARM='sh /usr/lpp/HOD/hodrapd.sh'
//SYSPRINT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD DUMMY
//STDOUT DD PATH='/tmp/HODRAPD.stdout',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
// PATHMODE=SIRWXU
//STDERR DD PATH='/tmp/HODRAPD.stderr',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
// PATHMODE=SIRWXU
```

After starting the service, verify that it started correctly by checking the syslog for a starting message from odsrapd if you have logging enabled. Otherwise the message will be displayed on the z/OS console. Alternatively, you may verify that the service started. You can check the MVS console for a starting message, or you can use the TSO **netstat conn** command or **onetstat -a** from UNIX System Services. Example 5-3 shows the result from an MVS **netstat** command:

Example 5-3 Netstat output of HODRAPD

GLDSRV	00000049	0.0.0.0..3389	0.0.0.0..0	LISTEN
GLDSRV	0000004B	9.100.203.110..3389	9.100.203.110..1042	ESTBLSH
GLDSRV	00000050	9.100.203.110..3389	9.100.203.110..1044	ESTBLSH
HODRAPD2	0000001D	0.0.0.0..2569	0.0.0.0..0	LISTEN
HODSRVR	0000004F	9.100.203.110..1044	9.100.203.110..3389	ESTBLSH
HODSRVR	00000052	0.0.0.0..3470	0.0.0.0..0	LISTEN
HODSRVR	0000001B	0.0.0.0..8999	0.0.0.0..0	LISTEN
HODSRVR	0000004E	0.0.0.0..20070	0.0.0.0..0	LISTEN
HODSRVR	0000004A	9.100.203.110..1042	9.100.203.110..3389	ESTBLSH

Native authentication uses the well-known port 2569. As you can see, the userid with a local socket = 0.0.0.0..2569 value is the user that started the process and is in listen status.

If the port is not in listen status, verify that the permission bits for HODRAPD in the private directory (/usr/lpp/HOD/hostondemand/private) are set correctly and the sticky bit (as denoted by the T in the display) is turned on.

```
-rwxr--r-T 1 OEKERN OMVSGRP 0 Nov 12 19:30 HODRAPD
```

Permissions should be set to 744. If that is not done, use the **chmod** command to change it:

```
chmod 744 /usr/lpp/HOD/hostondemand/private/HODRAPD
```

You can set the sticky bit via the **chmod** command:

```
chmod +t /usr/lpp/HOD/hostondemand/private/HODRAPD
```

This is assuming that you have used the default install, otherwise replace /usr/lpp/HOD/hostondemand with whatever your install path is.

When native authentication is started, you can see 2 processes with their associated process IDs (PID). In the OMVS shell issue the command **ps -ef**, or from TSO issue the **d omvs a=all** command to determine the processes. Example 5-4 on page 148 shows the output from the OMVS shell command.

Example 5-4 Display of HODRAPD PIDs

OEKERN	50335614	3968	-	Nov 27 ?	0:00 hodrapd -x
OEKERN	3968	1	-	Nov 27 ?	0:00 hodrapd -x

There are two ways to stop native authentication services. Either way, you need to have the PID shown in Example 5-4. In our case the PID is 3968. These are the commands to stop the service:

- ▶ From the OMVS shell: **kill 3968**
- ▶ From the TSO console: **F BPX0INIT,TERM=3968**

5.1.4 Working with native authentication

Now you should be ready to work with native authentication. Remember, that native authentication requires the LDAP server, RACF, and DB2 to be set up correctly.

1. Log on to the Host On-Demand administrator and select to Use Directory Service (LDAP), as shown in Figure 5-2.

The screenshot shows a configuration window titled "Use Directory Service (LDAP)". It contains several input fields and checkboxes. The "Use Directory Service (LDAP)" checkbox is checked. The "Destination Address" field contains "9.100.203.110". The "Destination Port" field contains "3389". The "Administrator Distinguished Name" field contains "cn=LDAP administrator". The "Administrator Password" field is masked with "xxxxxx". The "Distinguished Name Suffix" field contains "o=ibmmop". There is an "Advanced" checkbox which is unchecked. Below it, the "Users Location" field contains "cn=users,o=ibmmop", the "Groups Location" field contains "cn=user groups,o=ibmmop", and the "Domain Location" field contains "sys=HOD,o=ibmmop". At the bottom, there is a "Migrate Configuration to Directory Service" checkbox which is unchecked. "Apply" and "Cancel" buttons are at the bottom right.

Figure 5-2 Enabling connection to LDAP server

2. Enable native authentication for each user by selecting **Use Native Authentication** and entering the native userid, as shown in Figure 5-3 on page 149.

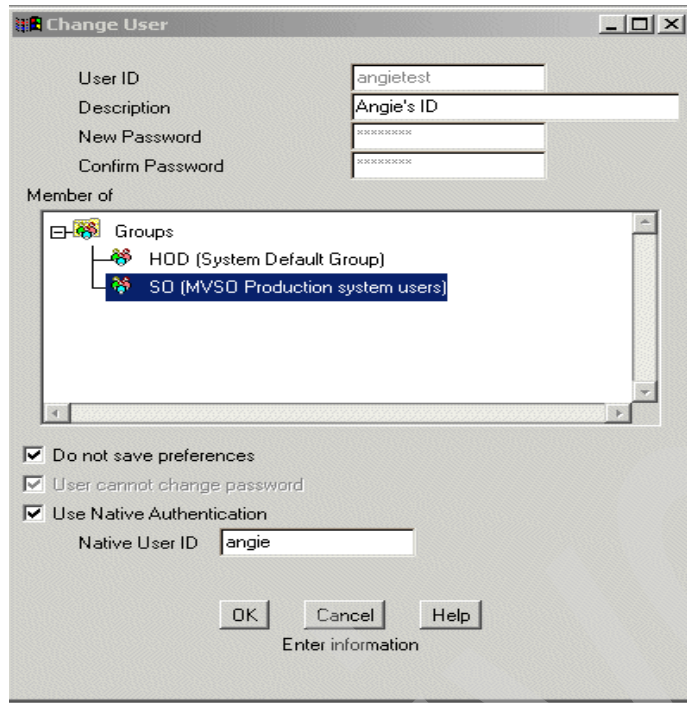


Figure 5-3 Enable user for Native Authentication

3. Click **OK**. This must be done for each user ID using native authentication services.
4. Log off the Host On-Demand administrator.
5. Download a Host On-Demand client, and log on to the user ID you defined using the password of the native user ID you specified. In our example, the user ID is Angietest and the password is that of the TSO user Angie as specified in RACF.

5.1.5 Problem determination

In this section we describe how to diagnose and fix a few common problems.

Cannot log on to the Administrator (LOG0001 error)

If changes were made to the LDAP configuration and you are unable to log in as the Administrator, you need to:

1. Delete the dirInfo.active file in the private directory.
2. Stop the Service Manager and the Native Authentication service.
3. Restart the Service Manager and Native Authentication service.
4. Load the admin page again and you should be able to log on to the Administrator and set up the connection to the LDAP server with the modified information you get from your LDAP administrator.

Enable Native Authentication is grayed out

Make sure that the Native Authentication service is started and in listen status, as shown in Example 5-3 on page 147. If that seems OK, make sure that the connection to the LDAP server was established as indicated in the **netstat** output in Example 5-3 (there is a connection to the LDAP server port 3389 from the HOD server on port 1044) or that the dirInfo.active is in the private directory.

Incorrect password

If you receive an incorrect password message and you know without a doubt that the user ID and password are correct, verify you have defined localhost in the /etc/hosts file. You must be able to resolve localhost. Add an entry for localhost as follows in the /etc/hosts file:

```
127.0.0.1    localhost
```

If you make a modification to the /etc/hosts file, you must recycle the Host On-Demand server. The HODRAPD task does not need to be recycled.

Another reason for this message could be that the user ID is actually revoked for whatever reason. If the changes in the /etc/hosts didn't correct the problem, contact your RACF (or other SAF product) administrator to correct this.

5.2 Telnet-negotiated security

Telnet-negotiated security allows a client to initiate a secure Telnet session. The session can begin as a non-secure session, but then be negotiated to a secure session as defined in the IETF internet-draft "TLS-based Telnet Security." The current draft can be found at:

<http://www.ietf.org/internet-drafts/draft-ietf-tn3270e-extensions-03.txt>

This draft defines extensions to Telnet that allow TLS to be negotiated over a Telnet connection.

The TLS protocol as defined in IETF Standards Track RFC 2246 "The TLS Protocol 1.0" is found at:

<http://www.ietf.org/rfc/rfc2246.txt>

The TLS Protocol 1.0 allows security negotiation down from TLS 1.0 to SSL. Host On-Demand clients will always negotiate down to SSL Version 3, since Host On-Demand supports internet-draft TLS-based Telnet security, meaning Telnet-negotiated security, but not TLS Protocol 1.0. The Telnet server must support TLS-based Telnet security for the Host On-Demand clients to use Telnet-negotiated security. The Communications Server for OS/390 Version 2 Release 10 and later supports negotiable Telnet security. Communications Server for OS/390 documentation refers to Telnet-negotiated security as "negotiable SSL."

For more information regarding Telnet-negotiated security, see the Telnet-negotiated security overview in the Host On-Demand online help.

In OS/390 V2R10 or above, with the CONNTYPE ANY keyword in the TELNETPARMS block, the telnet server can support both SSL clients and non-SSL clients over a single port. The CONNTYPE ANY statement indicates that the client can connect in either a secure or basic connection. The telnet server first establishes a telnet session, then negotiates security. If the client wishes to enter into a secure connection, SSL protocols will be used for all subsequent communication. If the client is not willing to enter a secure connection, a non-SSL or basic connection is used. For a complete discussion of Telnet-negotiated sessions refer to *IBM Host Access Client Package*, SG24-6182.

Note: It is *not* recommended to use CONNTYPE ANY if going through a firewall. This will allow a non-SSL connection through the firewall. For details about the CONNTYPE keyword refer to the *IBM Communications Server IP Configuration Reference* manual for your operating system release.

In order to implement Telnet-negotiated security, you must first enable SSL to activate the Telnet-negotiated radio button, then select **Yes** at the Telnet-negotiated radio button (see Figure 5-4).

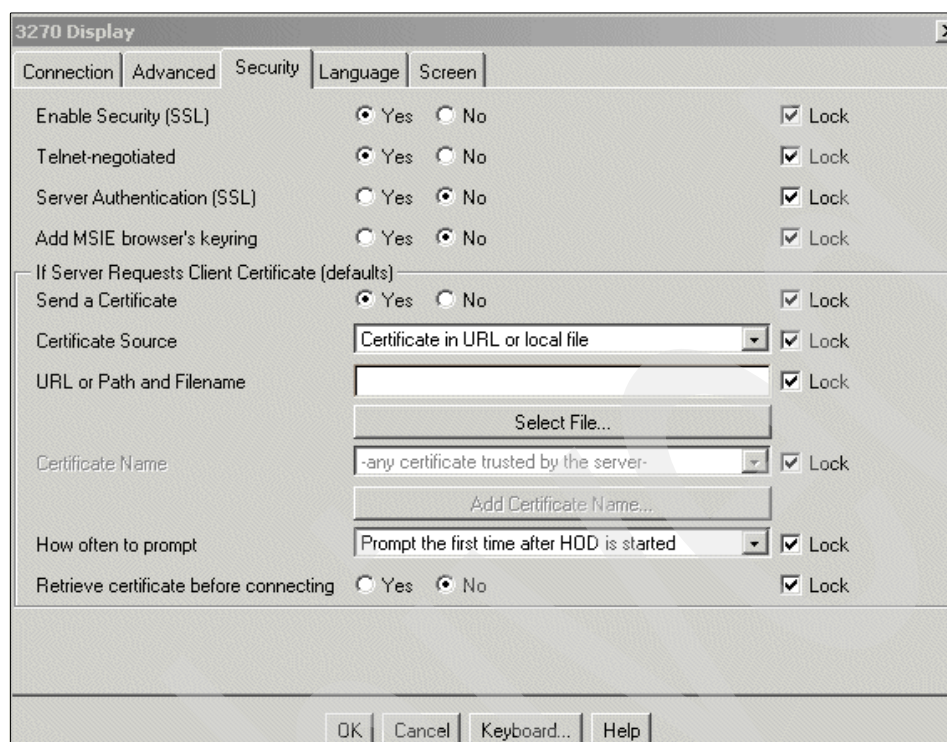


Figure 5-4 Enable Telnet-negotiated security

Selecting Telnet-negotiated determines if the SSL negotiation between the client and the server is done on the Telnet connection or on an SSL connection prior to the Telnet negotiations. The other SSL options are valid regardless of whether the Telnet-negotiated radio button is Yes or No.

If **Yes** is selected, then the Telnet protocol defined in IETF internet-draft "TLS-based Telnet Security" will be used to negotiate the SSL security after the Telnet connection is established. This support is only applicable with a Telnet server which supports TLS-based Telnet Security. Communication Server for OS/390 V2R10 and higher is the only IBM Telnet Server at this time which supports this function.

If **No** is selected, the traditional SSL negotiations will be done on an SSL connection with the server, and subsequently the Telnet negotiations with the server will be done. The default is **No** because few Telnet servers have this support since this is not yet an RFC.

There will be no migration considerations since this is not supported by Personal Communications Manager, and the default is No.

The Communication Server for OS/390 documentation refers to this feature as "negotiable SSL."

5.2.1 Session negotiation

A typical Telnet-negotiated SSL flow is shown in Figure 5-5 on page 152.

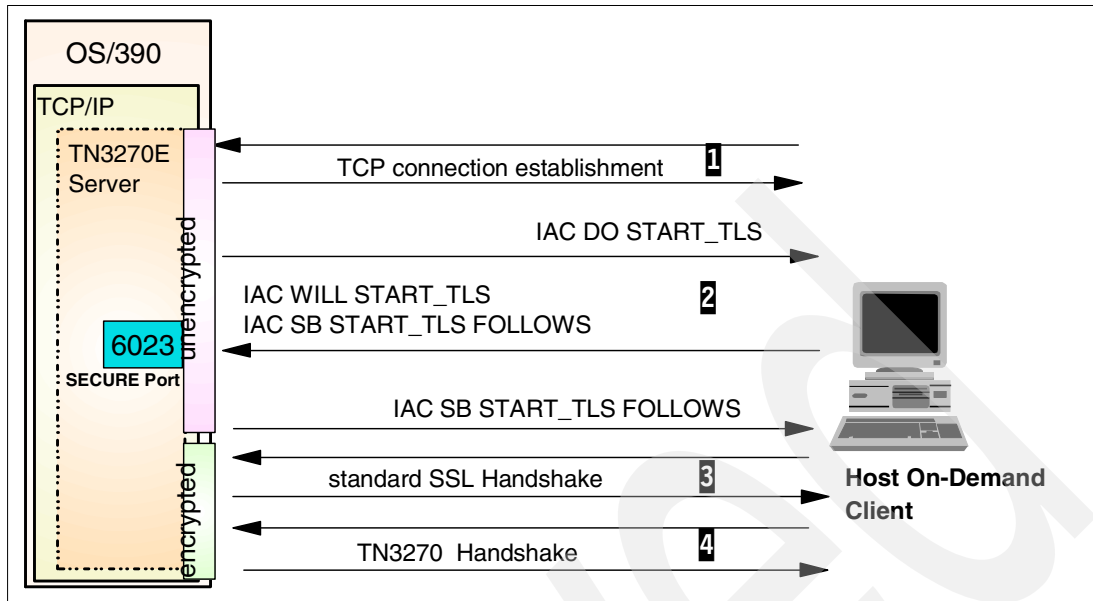


Figure 5-5 TLS-based Telnet SSL flow

1. IP connection is established.
2. The Telnet server sends the **IAC DO START_TLS** command to the client to verify whether it wants to perform the SSL negotiation.
3. If a positive response is received, then Telnet begins a normal SSL handshake.
4. If no positive response is received, the connection will be dropped.

The **IAC DO START_TLS** Telnet command, sent from the server, activates TLS at the beginning of a Telnet connection. The client can respond to this command by sending the **IAC WILL START_TLS** command, if the negotiation of a TLS connection is required. With the **IAC DONT START_TLS** command, the client can refuse the TLS connection negotiation. Sending the **IAC SB START_TLS FOLLOWS IAC SE** command initiates a TLS negotiation. When this subcommand has been sent and received, the TLS negotiation will begin.

If Enable Security (SSL) is Yes and Telnet-negotiated is Yes, then the Telnet connection will be started normally without SSL. However, the 3270 session will not start until the SSL negotiation completes successfully. If the server message is **WONT STARTTLS**, then the session will not start, and an error message will be issued stating "Security was requested, but the server does not support security."

If Enable Security (SSL) is No and the server requests a Telnet-negotiated session, Host On-Demand does not start a secure session and an error message is displayed on the status bar stating "The server requested security, but Security is not enabled."

5.3 Express logon feature

Many of the commonly used interfaces to automatically log users on to host applications expose RACF passwords and introduce the risk of severe security breaches. Such techniques include:

- ▶ Storing passwords in server databases
- ▶ Storing user IDs and passwords in cookies

- Including user IDs and password in URLs

The express logon feature(ELF), which was introduced in the IP Services of IBM Communications Server for OS/390 V2R10, provides an extremely secure method for automating the logon process. The express logon feature allows a user, with a TN3270 client and an X.509 certificate, to log on to System/390 host applications without having to type in a user ID and a password. The users need not memorize user IDs for the different host applications they are authorized to use. What they will need instead is a client certificate that is used to identify them and to check if they have been authorized for the specific application they are trying to access. This effectively eliminates the possible security risk of users selecting trivial passwords or writing down passwords and even leaving them on their desks.

Integrating 3270 host access into Web portals, for example, will be easier to implement because express logon makes the process of logging on to host applications more consistent with the user authentication used for accessing other Web applications.

To use the express logon feature you will need to:

- Define a RACF passticket profile for the application, described in 5.3.2, “The RACF-secured sign-on PassTicket” on page 157.
- Create the server certificate for the TN3270 server, discussed in 5.3.3, “Configuring the TN3270 server” on page 158.
- Create a client certificate, explained in 5.3.4, “Configuring the client” on page 161.
- Define the port for client authentication and add the EXPRESSLOGON parameter in the TELNETPARMS statement, described in 5.3.3, “Configuring the TN3270 server” on page 158

5.3.1 Overview

In order to use the express logon feature, the user or the administrator must record a Host On-Demand macro containing the express logon support. This macro then can be automatically started when the host connection is established; or the user can select the application he wants to log on to by starting the macro that was recorded for logging on to that application. Instead of a user ID and a password, the macro contains placeholder variables for user ID and password, which are replaced by the TN3270 server with the user’s actual user ID and a PassTicket before the logon request is forwarded to the application. This means that the user will not even be able to see the user ID used for his logon to the host application. The user ID and PassTicket are only seen by RACF and the Telnet server, and are passed between them over a secure connection.

The technique of using variables for user ID and password in the logon macro then makes it possible to automate the host application logon, and use the same macro for all users of a given application. Some possible scenarios are:

- The administrator configures individual user accounts and sets up sessions with logon macros for all major applications that can be started by the user.
- The administrator configures a general-use user account with a session for every major application, set up to automatically start the express logon macro for that application.
- The administrator uses the Deployment Wizard to customize an HTML page that automatically launches a host connection and starts the logon macro in order to navigate the user directly to the start point in his application.

A host session supporting express logon can be started for any type of client; it may be a cached or a download client, it may use the configuration server (that is, the session definitions are retrieved from the Host On-Demand server using a user ID and password), or a customized HTML page with all session definitions having been created using the Deployment Wizard.

The express logon feature is supported on two-tier and three-tier network designs. The two-tier design utilizes the z/OS TN3270 Telnet server. The three-tier design utilizes a middle-tier TN3270 server and a Digital Certificate Access Server (DCAS).

In order for an application to be accessed using the express logon feature, a PassTicket data class profile (PTKTDATA) must be defined on each target RACF system (that is, the host where DCAS is running, and any host where RACF and a target application are located).

Both network designs require a TN3270 client workstation that supports Secure Sockets Layer (SSL) connections with client authentication and an X.509 certificate. Using RACF services in z/OS, the client certificate must be associated with a valid user ID. The only client-side product that supports the express logon feature is IBM WebSphere Host On-Demand, V5 and V6.

Two-tier network design

In the two-tier design, the user starts an SSL connection with level 2 client authentication, which passes the client certificate to the MVS host TN3270 server. The MVS host TN3270 server uses RACF Secured Signon services to obtain a user ID and PassTicket.

The two-tier design is supported in z/OS V1R2 and OS/390 V2R10 + PQ47742 (UQ55691).

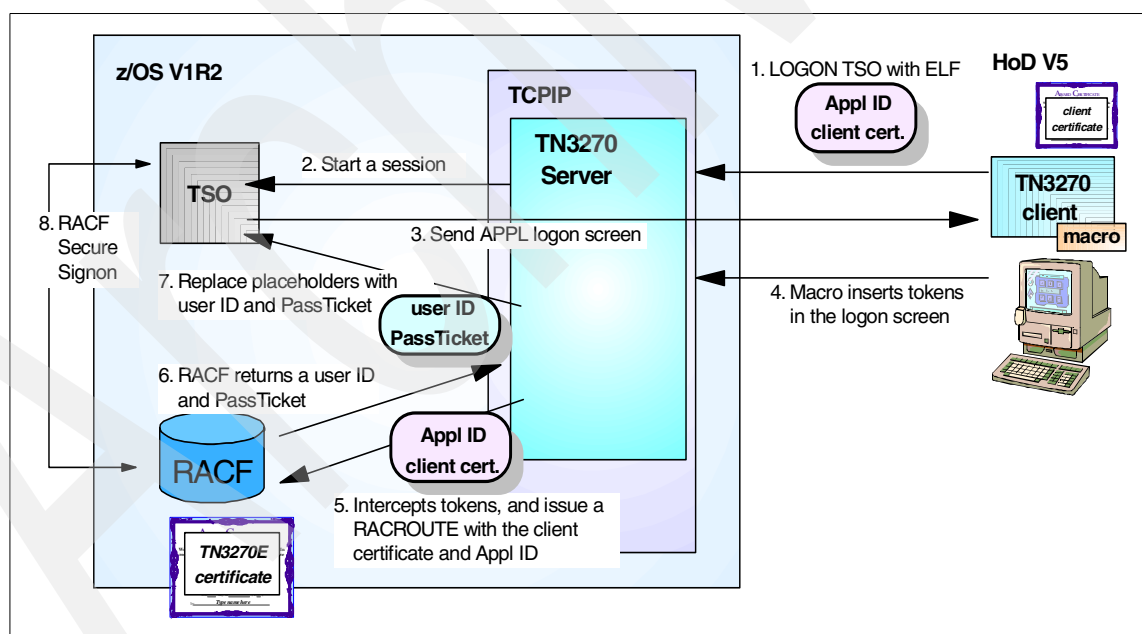


Figure 5-6 ELF two-tier network design

Here are the steps followed when a client wants to access a TSO session on z/OS:

1. The user has a Host On-Demand icon that starts an emulator session configured to use SSL client authentication. The session has a macro associated with it. A client certificate has to be available from the terminal and presented to the TN3270 server for the SSL handshake. During the SSL handshake, the client certificate is passed to the TN3270

server and validated. During Telnet function negotiation, the ELF capability is negotiated using RFC 1572.

2. The application ID is sent from the client to the TN3270 server and the server starts the session with TSO.
3. The logon screens come to the emulator.
4. The macro plays and inserts placeholder strings in the user ID and password fields.
5. The TN3270 Server intercepts the placeholder strings and sends the certificate and the target application ID to RACF.
6. RACF converts the Host On-Demand client's certificate to a TSO user ID and PassTicket and sends them back to the TN3270 server.
7. The TN3270 server inserts the user ID and PassTicket into the 3270 data stream at the macro-inserted placeholder locations and sends it to the application.
8. The application presents the user ID and PassTicket to RACF or another compatible host access control facility, which approves them, and the logon completes as usual.

Three-tier network design

In the three-tier design, the user starts the TN3270 connection to the middle-tier server.

There must be a Digital Certificate Access Requester (DCAR) and a Digital Certificate Access Server (DCAS).

The DCAS's client is the middle-tier TN3270 server or DCAR, which attempts to log on to an SNA application for the workstation client. The DCAS receives a digital certificate from the DCAR and returns a user ID and PassTicket. SSL communication is used between the DCAS and the DCAR. The server recognizes that the client wants the express logon function and invokes the DCAR, which opens an SSL connection with client authentication and passes the workstation's certificate and application name to the DCAS on the host. The DCAS uses RACF Secured Signon services to obtain a user ID and PassTicket, which the DCAS returns to the DCAR. The DCAR passes this information back to the TN3270 server.

The middle-tier IBM TN3270 servers supporting Express Logon are:

- ▶ Communications Server for OS/2 Warp V6.1
- ▶ Communications Server for Windows NT and Windows 2000 V6.1.1 PTF
- ▶ Communications Server for AIX 6.0.0.1 PTF

The term DCAR is used to describe the part of the TN3270 middle-tier server that supports the express logon feature and communicates as a client with the DCAS. It is not separate from the TN3270 middle-tier server.

A Digital Certificate Access Server resides on the host. DCAS uses RACF services to obtain a user ID.

The host also provides RACF Secured Signon services, which the DCAS or the MVS host Telnet server use to generate a PassTicket. A PassTicket is a RACF token similar to a password except that it is valid only for ten minutes.

The three-tier components are:

- ▶ A client workstation that supports SSL connections with client authentication and an X.509 certificate.
- ▶ A middle-tier TN3270 server, so called because it does not reside on the host, but rather between the client and the host. This server communicates with a DCAS using an SSL

connection with client authentication. It sends the user's certificate from the workstation and an application ID to the DCAS and expects to receive a user ID and PassTicket (a one-time password) in response. This is the user ID and password that will be used to log on to the SNA application.

- The Digital Certificate Access Server resides on the host. The DCAS uses RACF services to obtain a user ID that is associated with the certificate sent by the client. RACF also provides secured sign-on services, which the DCAS uses to generate a PassTicket. A PassTicket is a RACF token similar to a password except that it is valid only for 10 minutes.

The SNA connection between the TN3270 server and SNA application can be SNA LU2, DLUR, HPR/IP (EE), or AnyNet connection.

SSL communication with client authentication is required in the configuration of the express logon feature on the HOD client, TN3270 server, and OS/390 DCAS server.

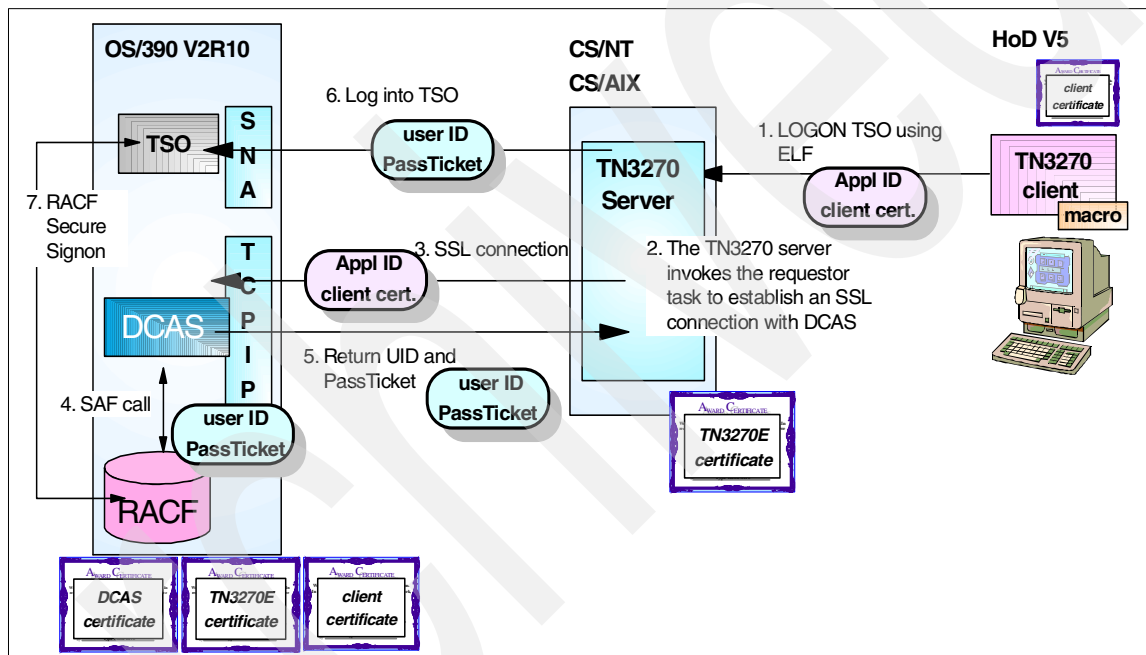


Figure 5-7 ELF three-tier network design

In the example illustrated in Figure 5-7, the client wants to access a TSO session on z/OS.

1. The user has a Host On-Demand icon that starts an emulator session configured to use SSL client authentication. The session has a macro associated with it. A client certificate has to be available from the terminal and presented to the TN3270 server for the SSL handshake.

During the SSL handshake, the client certificate is passed to the TN3270 server and validated. During Telnet function negotiation, the ELF capability is negotiated using RFC 1572.

The application ID is sent from the client to the TN3270 server. The logon screens come to the emulator as usual, but the macro plays and inserts placeholder strings in the user ID and password fields. The TN3270 Server intercepts the placeholder strings.

2. Once the application ID and client certificate are received by the TN3270 server, it invokes the DCAR function to establish the SSL communication to the DCAS server. The TN3270 server's certificate has to be sent to DCAS and authenticated.

3. The TN3270 server sends the certificate and the target application ID to DCAS on the z/OS host over a secure, trusted connection.
4. The DCAS server makes SAF calls to convert the HOD client's certificate to a TSO user ID and issue a PassTicket.
5. The DCAS server sends the user ID and PassTicket back to the TN3270 server.
6. The TN3270 server inserts the user ID and PassTicket into the 3270 data stream at the macro-inserted placeholder locations and sends it to the application.
7. The application presents the user ID and PassTicket to RACF or another compatible host access control facility, which approves them, and the logon completes as usual.

Once the express logon macro successfully ends processing, users will see exactly the same application screen as if they had manually logged on or used a traditional macro that prompts for user ID and/or password. Once finished, the user is free to start a macro (with or without express logon support) to initialize a session with the same or another host application.

We implemented the two-tier design on our system. For more information on how to set up the three-tier design, see *Communications Server for z/OS V1R2 Implementation Guide*, SG24-5227.

5.3.2 The RACF-secured sign-on PassTicket

RACF provides an alternative to the normal RACF password that remains the same for a specified time period (usually several weeks) until the user is prompted to enter a new password. The RACF PassTicket is a password generated dynamically, on request from a product or function, and *used only once*.

Of course, the 8-byte alphanumeric string of a PassTicket will not be encrypted on a normal 3270 session data stream. But anybody trying to use a PassTicket that he has recorded (by whatever means) will have only limited success. A PassTicket expires within 10 minutes. Express logon enforces encryption only on the Telnet part of the data flows. The LU to LU session may use normal SNA encryption, but this is not required.

The algorithm used to generate a PassTicket requires as input:

- ▶ The RACF user ID that identifies the user on the system on which the target application runs.
- ▶ The application name as defined for the target application.
- ▶ The RACF secured sign-on application key used as the encryption key by the DES algorithm, which is called several times during the process of generating the PassTicket.
- ▶ Time and date information in the form of a 4-byte number representing the number of seconds that have elapsed since January 1, 1970, at 0000 Greenwich Mean Time (GMT).

Define the PassTicket profile to RACF

For each application to which users are to gain access with a PassTicket, you must define a PTKTDATA class profile. We used TSO as the application; the RACF commands to create this profile are shown in Example 5-5.

Example 5-5 PassTicket definitions in RACF

```

SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA TSOSO SSIGNON(KEYMASKED(E6E6D30195D4D5C1)) UACC(NONE)
SETROPTS RACLIST(PTKTDATA) REFRESH

```

The RACF PassTicket generator algorithm uses cryptographic techniques to generate an 8-byte alphanumeric string from the input information and ensures that each PassTicket is unpredictable and unique. This algorithm is described in detail in *OS/390 Security Server (RACF) Macros and Interfaces*, SC28-1914.

5.3.3 Configuring the TN3270 server

The first step is to create the digital certificate for the TN3270 server. Either the GSKKMAN utility or RACF can be used to create the digital certificate.

For a complete description of the RACF enhancements, see Chapter 1, “z/OS 1.2 LDAP Directory enhancements” on page 3.

The steps involved to configure the TN3270 server for express logon are:

1. Create the server certificate.
2. Create the keyring file.
3. Add the certificate to the keyring.
4. Define the ports to the Telnet server.
5. Make certificate available to the client.

Create the server certificate

We used RACF to create a self-signed certificate. You can also use a Certificate Authority, such as Verisign or Thawte, to get a CA-signed certificate, or you can use RACF as the CA. The steps are similar to the ones described here. For more detailed information on how to create digital certificates using RACF or GSKKMAN, refer to *Communications Server for z/OS V1R2 Implementation Guide*, SG24-5227.

To create the certificate use the **RACDCERT GENCERT** command:

```
RACDCERT ID(OEKERN) GENCERT SUBJECTSDN (CN('MVS0.MOP.FR.IBM.COM') OU('MOPPPSC') C('FR'))  
WITHLABEL('TN3270 SERVER') KEYUSAGE(HANDSHAKE)
```

Create the keyring file

Use the **RACDCERT ADDRING** command to create the keyring:

```
RACDCERT ID(OEKERN) ADDRING(tn3270)
```

Note: Do not use any special characters, such as _ (underscore) or spaces, in the name of the keyring. Even though RACF will not complain about the underscore, for example, the setup for the secureport in the TELNETPARMS will fail with the message that this is an invalid keyring name (EZZ6035I).

Add the certificate to the keyring

To add the certificate to the keyring file use the **RACDCERT CONNECT** command:

```
RACDCERT ID(OEKERN) CONNECT (ID(OEKERN)) LABEL('TN3270 SERVER') RING(tn3270) DEFAULT)
```

You can check and make sure that this was successful by issuing the following command:

```
RACDCERT ID(OEKERN) listring *
```

Digital ring information for user OEKERN:			
Ring:			
>tn3270<			
Certificate Label Name	Cert Owner	USAGE	DEFAULT
-----	-----	-----	-----
TN3270 server	ID(OEKERN)	PERSONAL	YES

Figure 5-8 Listing the keyring information

Define the ports for SSL and client authentication

The next step is to create the port definitions in your TELNETPARMS. On our system we used 2 ports for this: port 1223 for server authentication only, and port 2223 for client authentication, as shown in Example 5-6.

Example 5-6 Telnet Server definitions for ELF support

```

;
TELNETPARMS
SECUREPORT 1223
    KEYRING SAF tn3270
    CONNTYPE ANY
ENDTELNETPARMS
;SSL, Client authentication and Expresslogon
TELNETPARMS
    DEBUG DETAIL
    SECUREPORT 2223
    KEYRING SAF tn3270
    CLIENTAUTH SAFCERT
    CONNTYPE SECURE
    EXPRESSLOGON
ENDTELNETPARMS
BeginVTAM
Port 1223 2223
    ; Define logon mode tables to be the defaults shipped with the
    ; latest level of VTAM
    TELNETDEVICE 3278-2-E NSX32702,SNX32702
    TELNETDEVICE LINEMODE INTERACT
    ; Define the LUs to be used for general users.
    DEFAULTLUS
        SOTC0001 SOTC0002 SOTC0003 SOTC0004 SOTC0005
    ENDDEFAULTLUS
;
DEFAULTAPPL TSO ; Set the default application for all TN3270(E)
ALLOWAPPL *

```

There are two essential parameters that are needed for SSL: SECUREPORT and KEYRING.

SECUREPORT

All SSL-enabled TN3270 ports must be defined by specifying a TELNETPARMS block for each port. The SECUREPORT port designation statement in the TELNETPARMS block indicates the port is capable of handling SSL connections.

KEYRING

The keyring type and location is specified in the KEYRING statement. Only one keyring can be used by the TN3270 server. The keyring can be defined in the TELNETGLOBALS or TELNETPARMS block. TELNETGLOBALS is the preferred definition method since it ensures that the same keyring has been defined for all SECUREPORTs. If specified in TELNETPARMS, the same keyring type and file must be specified for each SECUREPORT. The first keyring file name read is considered the correct keyring file name. The TELNETGLOBALS keyring is read first and then the TELNETPARMS keyrings are read in reverse order. Any keyring that does not match the first is rejected and the port update fails.

The express logon feature also requires the keywords CLIENTAUTH and EXPRESSLOGON.

CLIENTAUTH

The CLIENTAUTH parameter indicates that the client must send a client certificate to the server. If this parameter is not specified, a client certificate is not requested during the SSL handshake and no certificate-based client authentication is done. The level of validation done depends on the option specified. Valid CLIENTAUTH options are: ? SSLCERT - client authentication Level 1. ? SAFCERT - client authentication Level 2 and 3. ? NONE - No client authentication is requested. This is the default

Client authentication can be implemented in three levels:

► Level 1

To pass authentication, the CA that signed the client certificate must be considered trusted by the server (that is, a certificate for the CA that issued the client certificate is listed as trusted in the server's keyring). You can have it coding CLIENTAUTH SSLCERT in the TCP/IP profile. If the certificate issuer (CA or self-signed) is not part of the list of well-known CAs, the keyring must be primed with the signer of the CA or the self-signed client certificate.

► Level 2

The Level 1 checking provided by SSLCERT is done and Level 2 checking is done to verify that the certificate has been registered with RACF (or another SAF-compliant security product that supports certificate registration). To implement Level 2, the client certificate has to be defined in RACF associated to a userid and CLIENTAUTH SAFCERT has to be defined in the TCP/IP profile.

► Level 3

This authentication level provides, in addition to level 1 and level 2 support, the capability to restrict access to the server based on the user ID returned from RACF. The connection is allowed only if the user ID associated with the client certificate has READ access to the RACF resource. This level of control uses the SERVAUTH RACF class to restrict access to the server. If the SERVAUTH class is not active or the SERVAUTH profile for the server is not defined, it is assumed level 3 authentication is not requested and level 2 is assumed.

EXPRESSLOGON

The EXPRESSLOGON parameter statement allows a user at a workstation, with a TN3270 client and a X.509 certificate, to log on to an SNA application without entering a user ID or password. If this statement is not coded or NOEXPRESSLOGON is specified, EXPRESSLOGON function is not available to the client.

Make certificate available to the client

If the certificate was created using one of the well-known CAs, such as Verisign or Thawte, nothing further needs to be done since the certificate is already included in the WellKnownTrustedCAs.class file in the publish directory. Download or cached clients will have access to the certificate from the Host On-Demand server when this file is downloaded to their workstation.

However, if the certificate was either self-signed or signed by an unknown CA, it must be made available to the client. This is done via the CustomizedCAs.class file, which will also be downloaded to the client along with the client code. There are different ways of making the certificate available to the client:

- ▶ Have each client add the certificate into their browser. This is not necessarily the recommended method and can be very cumbersome, since it has to be done for each and every client.
- ▶ Create the CustomizedCAs.class file on the server, which will be automatically downloaded to the client.

Create the CustomizedCAs.class file and copy it to the publish directory

Run the Java connect command to create the CustomizedCAs.class file which will be downloaded to the client. The command is rather lengthy and you may need to use it again. Therefore, you may want to create a shell script to run this.

This is what our javakr.sh file looked like when we created the CustomizedCAs.class file to make the server certificate available to the client:

```
java -classpath ./usr/lpp/HOD/hostondemand/lib/sm.zip \  
com.ibm.hodsslght.tools.keyrng CustomizedCAs connect 9.100.203.110:1223
```

The '\ ' is a continuation, since the complete command doesn't fit on one line.

In a later example you will see that we issued the command again, but that time for port 2223 when we created the client certificate and had to add the signer of the certificate, in this case the RACF CA called "porting CA," to the keyring.

You will be prompted for the password for CustomizedCAs.class file. Do *not* specify a password. If you do, it will not work.

You can verify that the command was successful by issuing the Java verify command. Again, we put it in a file called javaver.sh:

Example 5-7 Verifying the CustomizedCAs.class file generation

```
java -classpath ./usr/lpp/HOD/hostondemand/lib/sm.zip \  
com.ibm.hodsslght.tools.keyrng CustomizedCAs verify
```

The last step is to move the CustomizedCAs.class file to the publish directory:

```
mv CustomizedCAs.class /usr/lpp/HOD/hostondemand/HOD
```

5.3.4 Configuring the client

Before you can start recording a macro using the express logon facility, you have to define a session that is able to provide express logon support. When recording the macro, the session definitions are not checked; that is, you can record a macro for express logon support that might not work correctly when played.

The session must be configured for SSL and client authentication. A client certificate must have been installed on the client or must be accessible from a server. The destination IP address must specify a server that has been set up to support the express logon facility.

Setting up the client certificate

The client certificate can be created either using the GSKKMAN utility or RACF. The following example is based on RACF. In this example we used a CA that was already defined on our system, called “porting CA.”

Create the certificate and sign it with the RACF CA called “porting CA” using the **RACDCERT GENCERT** command:

```
RACDCERT ID(ANGIE) GENCERT SUBJECTSDN(CN('Angie Boone') OU('PSSC') O('IBM')
L('Montpellier') C('FR')) SIZE(1024) WITHLABEL('ANGIEB') SIGNWITH(CERTAUTH
LABEL('porting CA')) KEYUSAGE(HANDSHAKE)
```

When we did this, we got an error message indicating a date problem:

The certificate that you are creating has an incorrect date range. The certificate is added with NOTRUST status.

To check the expiration date on the porting CA, display the CA certificate via the **RACDCERT CERTAUTH LIST** command:

```
RACDCERT CERTAUTH LIST(LABEL('porting CA'))
```

The dates are shown in Example 5-8.

Example 5-8 Display of the CA certificate

```
Label: porting CA
Certificate ID: 2QiJmZmDhZmjgZewma0J1YdAw8FA
Status: TRUST
Start Date: 2001/10/11 00:00:00
End Date: 2002/10/11 23:59:59
Serial Number:
>00<
Issuer's Name:
>CN=porting CA.O=psscmap.C=fr<
Subject's Name:
>CN=porting CA.O=psscmap.C=fr<
Key Usage: CERTSIGN
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
Ring Owner: OEKERN
Ring:
>tn3270e<
Ring Owner: OEKERN
Ring:
>tn3270<
```

Next list the certificate we just created for the client using the **RACDCERT LIST** command:

```
RACDCERT ID(ANGIE) LIST(LABEL('ANGIEB'))
```

As you can see in the output from this command, shown in Example 5-9 on page 163, the date it picked up as the issue date is today's date, and the expiration date is past the one of the CA. It also shows a status of NOTRUST.

Example 5-9 Display of the client certificate showing NOTRUST status

Digital certificate information for user ANGIE:

```
Label: ANGIEB
Certificate ID: 2QXB1cfJxcHVx8nFwkBA
Status: NOTRUST
Start Date: 2001/12/07 00:00:00
End Date: 2002/12/07 23:59:59
Serial Number:
    >03<
Issuer's Name:
    >CN=porting CA.O=psscmap.C=fr<
Subject's Name:
    >CN=Angie Boone.OU=PSSC.O=IBM.L=Montpellier.C=FR<
Key Usage: HANDSHAKE
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
*** No rings associated ***
```

To change the status to TRUSTED you can use the following command:

```
RACDCERT ID(ANGIE) ALTER(LABEL('ANGIEB')) TRUST
```

Now the output from the RACDCERT LIST command, shown in Example 5-10, shows that the certificate is trusted.

Example 5-10 Display of client certificate in TRUST status

Digital certificate information for user ANGIE:

```
Label: ANGIEB
Certificate ID: 2QXB1cfJxcHVx8nFwkBA
Status: TRUST
Start Date: 2001/12/07 00:00:00
End Date: 2002/12/07 23:59:59
Serial Number:
    >03<
Issuer's Name:
    >CN=porting CA.O=psscmap.C=fr<
Subject's Name:
    >CN=Angie Boone.OU=PSSC.O=IBM.L=Montpellier.C=FR<
Key Usage: HANDSHAKE
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
*** No rings associated ***
```

Since the certificate was signed by the RACF CA “porting CA,” it must be added to the server’s keyring file, so that it is known to the client. To do this, issue the **RACDCERT CONNECT** command:

```
RACDCERT ID(OEKERN) CONNECT (CERTAUTH LABEL('porting CA') RING(tn3270))
```

To make sure that the keyring contains all the necessary information, issue another **RACDCERT LISTRING** command:

```
RACDCERT ID(OEKERN) LISTRING (tn3270)
```

Example 5-11 Display of keyring file

Digital ring information for user OEKERN:

Ring:

>tn3270<

Certificate Label Name	Cert Owner	USAGE	DEFAULT
-----	-----	-----	-----
TN3270 server	ID(OEKERN)	PERSONAL	YES
porting CA	CERTAUTH	CERTAUTH	NO

Next run the Java connect command to connect the certificate to port 2223. Again, we used the javakr.sh script with the port number that will be used for ELF:

```
java -classpath ./usr/lpp/HOD/hostondemand/lib/sm.zip \
com.ibm.hodsslght.tools.keyrng CustomizedCAs connect 9.100.203.110:2223
```

You will be prompted for the password again for CustomizedCAs.class file. Remember that you must not enter a password here or else it will fail.

Now that the client certificate has been created, you need to export the digital certificate to a z/OS data set, using the **RACDCERT EXPORT** command:

```
RACDCERT ID(ANGIE) EXPORT(LABEL('ANGIEB')) DSN(MONT.P12) FORMAT(PKCS12DER)
PASSWORD('password')
```

Now the data set that contains the digital certificate in a DER-encoded X.509 format is ready to be imported into the client's browser. The procedure is described in the next section.

5.3.5 Installing the client certificate into the browser

On the workstation, create a directory in which you want to store the certificate. This requires the following steps:

1. Open a command prompt in Windows.
2. Create a directory and change into the new directory. In this example we used temp.
3. Open an ftp session to the z/OS host. You will be prompted for your RACF userid and password. Once you are authenticated, you should already be in the directory where the exported certificate is.
4. At the command prompt enter bin to download the certificate in binary format.
5. Issue the **get** command to retrieve the certificate into the temp directory.

Example 5-12 illustrates this sequence of events.

Example 5-12 FTP the certificate to the client

```
C:\temp>ftp 9.100.203.110
Connected to 9.100.203.110.
220-FTPD1 IBM FTP CS V1R2 at MVS0, 10:31:07 on 2001-12-10.
220 Connection will close if idle for more than 5 minutes.
User (9.100.203.110:(none)): angie
331 Send password please.
Password:
230 ANGIE is logged on. Working directory is "ANGIE.".
ftp> bin
200 Representation type is Image
ftp> get mont.p12
200 Port request OK.
125 Sending data set ANGIE.MONT.P12
```

```
250 Transfer completed successfully.  
ftp: 2396 bytes received in 0.00Seconds 2396000.00Kbytes/sec.  
ftp> quit  
221 Quit command received. Goodbye.
```

The next step is to install the certificate into your Internet Explorer or Netscape browser. The following is an example of how to install it into Internet Explorer. For instructions on how to install it on Netscape, refer to Chapter 2, "RACF enhancements" on page 15.

Bring up a browser window and select **Tools -> Internet options -> Content -> Certificates -> Import**. Then a window with the Certificate Import Wizard will pop up, as shown in Figure 5-9.

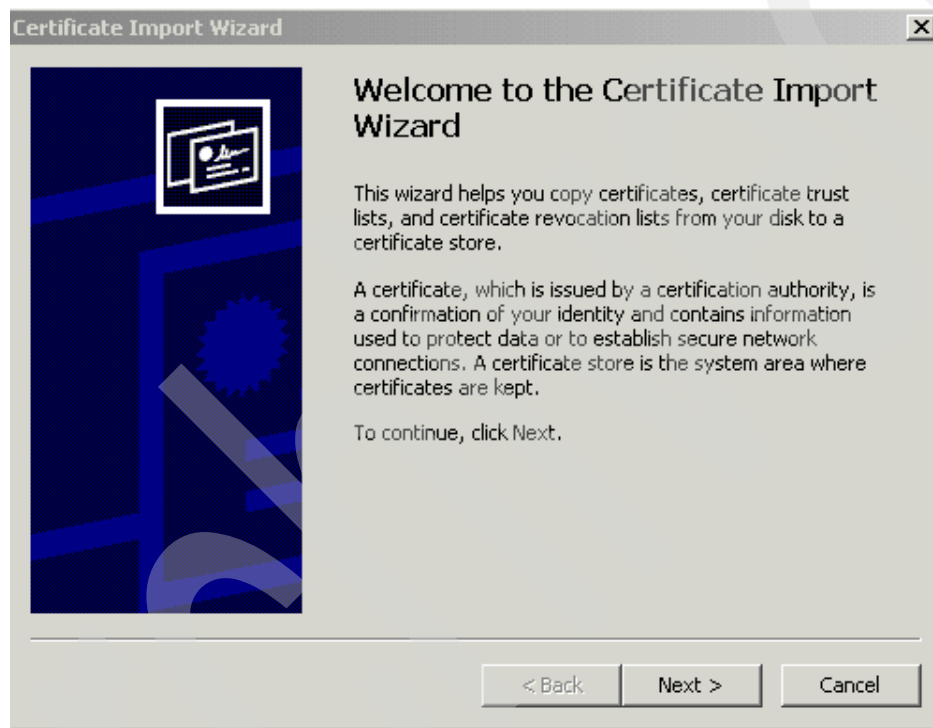


Figure 5-9 Certificate Import Wizard initial page

Next you will be prompted for the certificate file you wish to import. You can either enter the file name with the complete path directly in the field, or select **Browse** to search the directories for the right file.

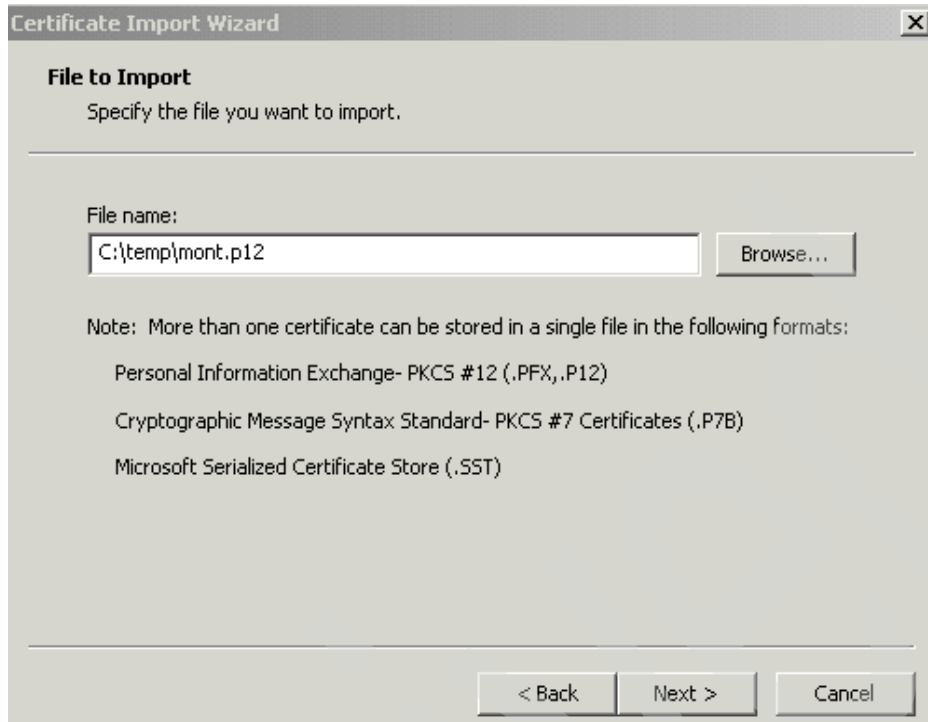


Figure 5-10 Specify the name of the certificate file

Once you have selected the file, you will be prompted for the password you created to protect the private key, as depicted in Figure 5-11.

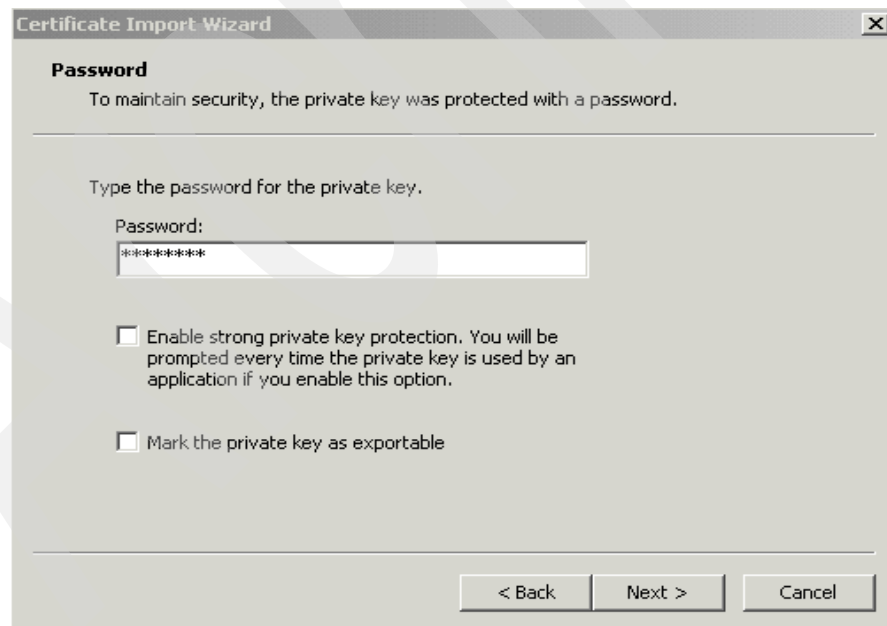


Figure 5-11 Specify the password

Now you are prompted to select where the certificate is to be stored. You can either define a directory yourself, or have the wizard select it for you. We chose to have the wizard automatically select it, as shown in Figure 5-12 on page 167.

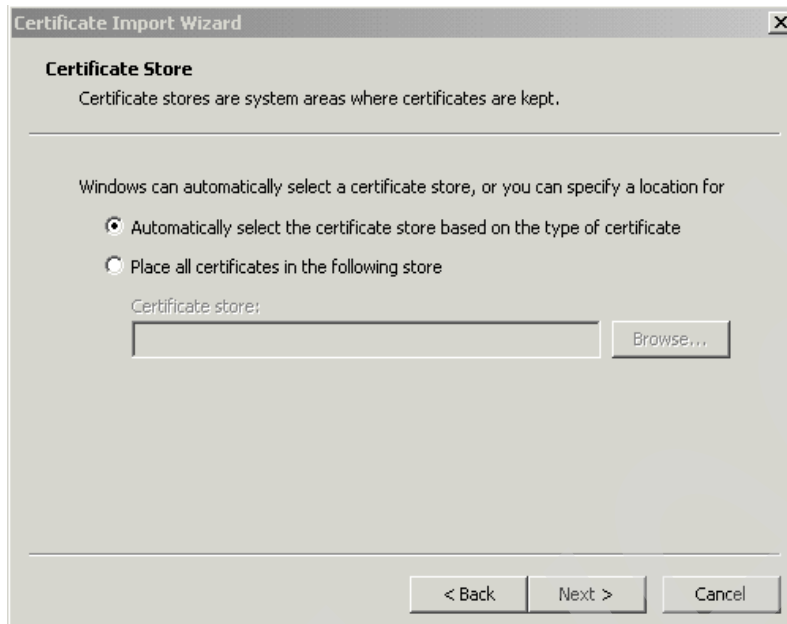


Figure 5-12 Select the certificate store

Once this is done, you are presented with the panel indicating that the certificate import is completed, as shown in Figure 5-13.



Figure 5-13 Completing the Certificate Import

The next panel asks if you want to add the certificate to the Root Store.

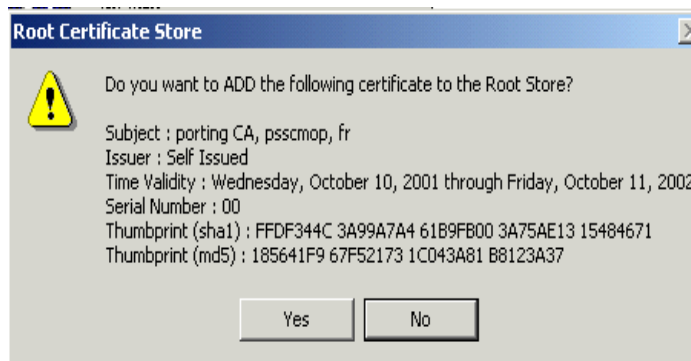


Figure 5-14 Adding the certificate to the Root Store

Finally, you will get the panel that indicates the successful import of the certificate.

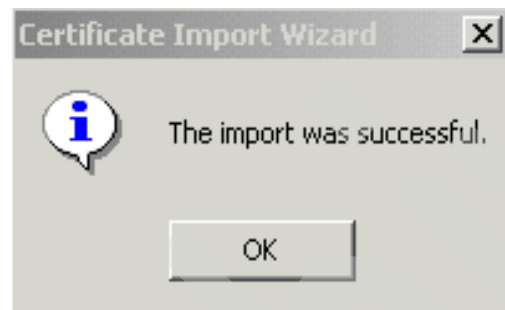


Figure 5-15 Successful certificate import

If the connection to the TN3270 server is through a Host On-Demand Redirector, the security option for the Redirector must be set to Pass-through. The TN3270 server (not the Redirector) has to do the SSL handshake protocol with the client and has to get the client certificate for later use during express logon and, consequently, has to do the client authentication.

Recording the macro: Basic definitions

Recording the macro is started the normal way by clicking the **Record** macro button on the session window's tool bar or by clicking **Actions -> Record Macro**. The session itself may have been started from a client by logging in as a user and then opening the intended session window. You may also record an express logon macro as an administrator customizing an HTML page that, when referenced, automatically opens the session window, starts the macro, logs the user on to his host application, and navigates the user to the application's start screen.

Figure 5-16 shows the first window that appears when you start recording an express logon macro. On this window you have to specify the name of the new macro (of course, you may also append to or overwrite an existing macro). Select the **Express Logon Feature** check box to indicate that you want to use express logon.

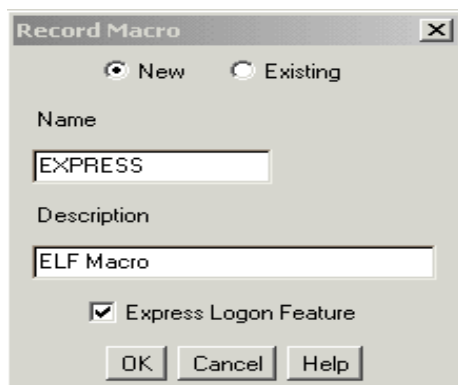


Figure 5-16 Recording the macro: Begin recording

Click **OK**; the window shown in Figure 5-17 appears, prompting you to enter the application ID of the application you want to log onto using this macro. This is the name of the application under which it has been defined to RACF in the PTKTDATA profile.

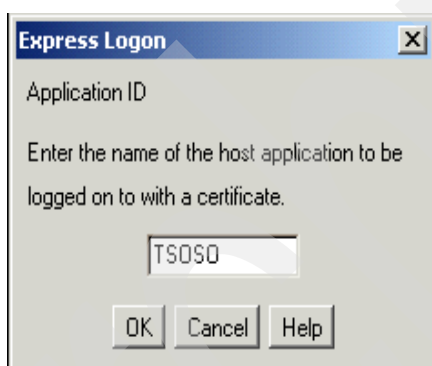


Figure 5-17 Recording the macro: Application ID

After you enter the application ID and click the **OK** button, the window in Figure 5-18 appears. It prompts you to actually start recording your actions on the session window. Once you have reached the screen prompting you for the user ID, click the **OK** button.



Figure 5-18 Recording the macro: Screen selection criteria

Figure 5-19 shows the next window, which asks if this is an alternate start screen. You can define alternate start screens, of which can be more than one, in the first or a follow-on editing pass through the macro. This will allow the user to start the macro (or have it started automatically) when the host session is initialized. After logging off from the application, a different logon screen might be presented to the user (for example, the application's logon screen and not VTAM's USSMSG10). This allows the user to use the same macro for one application, independent of where he starts. On this panel select **No** and then **Next>**.

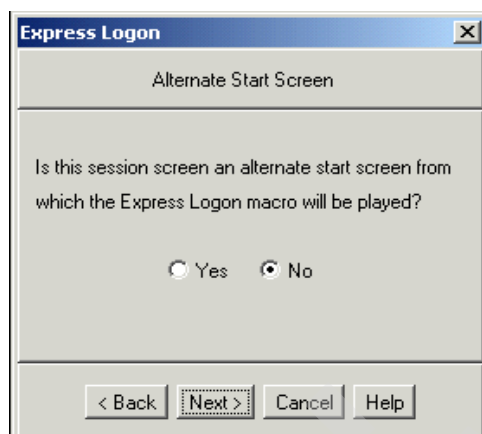


Figure 5-19 Recording the macro: Alternate start screen

The next window, when not defining an alternate start screen, asks if there is a user ID field on the current host screen. Selecting **Yes** and **Next** leads you to a window that lets you define the position of the user ID field on the host screen, as shown in Figure 5-20.

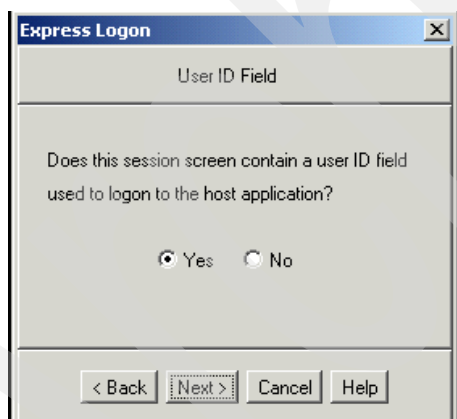
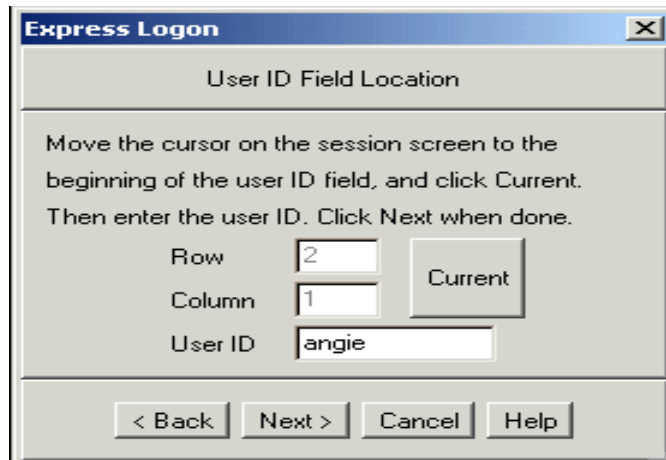


Figure 5-20 Recording the macro: User ID field

The simplest way of getting the correct row and column is to position the cursor on the user ID input field (normally it will already be correctly positioned) and click the **Current** button. This will update the input fields in the window with the current cursor position. In the user ID field, fill in a valid user ID. This user ID will only be used to log on to the host application when recording the macro; it will not be recorded in the macro. A placeholder variable, \$USR.ID\$, will be placed in the macro and actually filled in at the host screen's user ID field when the macro is played. The TN3270 server will replace this variable with the user's correct user ID.



Express Logon

User ID Field Location

Move the cursor on the session screen to the beginning of the user ID field, and click Current. Then enter the user ID. Click Next when done.

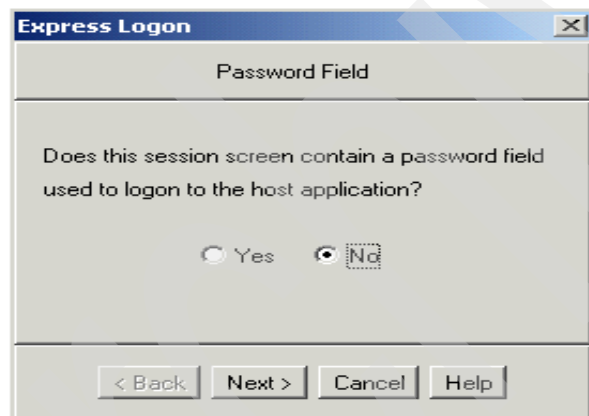
Row: 2
Column: 1
User ID: angie

Current

< Back Next > Cancel Help

Figure 5-21 Recording the macro: User ID field location

The next window asks if there is also a password field on the host screen that prompts for the user ID. If you answer **Yes**, the password field must be on the same screen as the user ID field. Since the session screen doesn't contain a password field, select **No** and **Next>**, as illustrated in Figure 5-22.



Express Logon

Password Field

Does this session screen contain a password field used to logon to the host application?

☐ Yes ☒ No

< Back Next > Cancel Help

Figure 5-22 Recording the macro: Password field

After navigating to the screen prompting for the password (by going to the TSO screen and clicking **ENTER**), click **OK** on the Screen Criteria selection panel shown in Figure 5-23 on page 172.

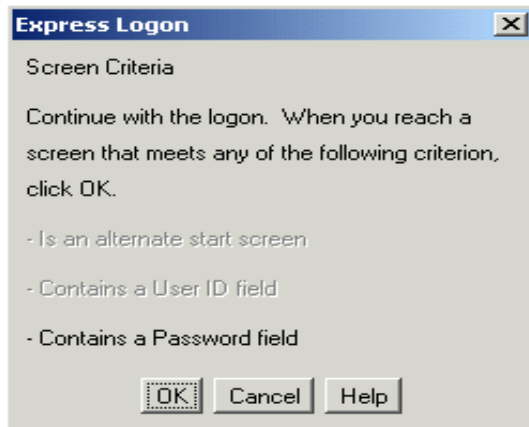


Figure 5-23 Recording the macro: Screen selection criteria

In the next panel, you have to define the position of the password field on the screen via a window similar to the one used for positioning the user ID field, as shown in Figure 5-24. Also, the password you are entering here is not recorded in the macro. It is only used to actually log on when recording the macro. The macro will again contain the placeholder variable, \$PSS.WD\$, that will be replaced with the PassTicket by the TN3270 server when playing the macro.

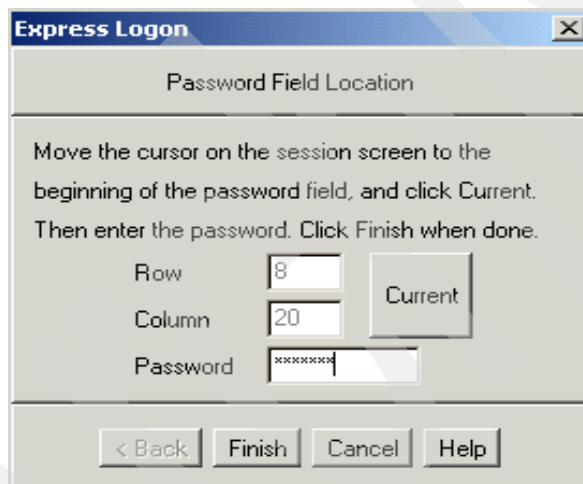


Figure 5-24 Recording the macro: Password field location

When you click **Finish** on the panel in Figure 5-24, the panel shown in Figure 5-25 on page 173 is displayed, giving instructions on how to continue. Only when you really want the user to press the Enter key, or whichever PF key is used for the logon, do you stop the macro. Otherwise, click the **OK** button to continue recording the macro until you have reached the application's start window where you want the user to be.

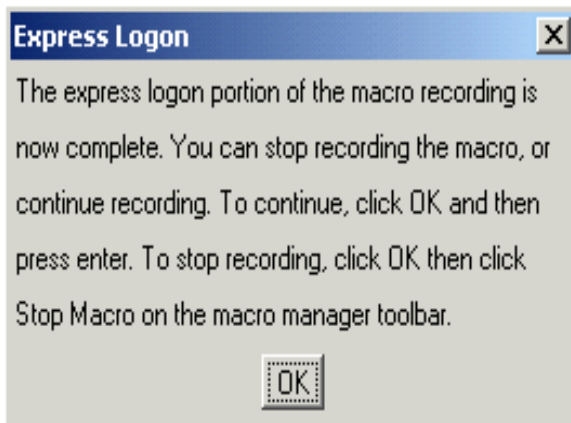


Figure 5-25 Recording the macro: Select to stop the macro

Once you click **OK** you will be presented with another window, which lets you select to have the macro autostart when you run this session. Select **Yes** on this panel. If you select **Yes**, the corresponding session definitions will be updated.

Finally, stop recording the macro and play it back to see if this worked OK.

The macro: An example

The following shows the express logon macro generated to log on to an application called ra03t. You can see that the user ID and password entered when recording the macros have been replaced with the placeholder variables, \$USR.ID\$ and \$PSS.WD\$.

Example 5-13

```
<HAScript name="tso03" description="Express Logon to TS003" timeout="60000" pausetime="300"
promptall="true" author="" creationdate="" supressclearevents="false" >
```

```
  <screen name="Screen1" entryscreen="true" exitsscreen="false" transient="false">
    <description>
      <oia status="DONTCARE" optional="false" invertmatch="false" />
      <numfields number="1" optional="false" invertmatch="false" />
      <numinputfields number="1" optional="false" invertmatch="false" />
    </description>
    <actions>
      <custom id="Application_ID" args="ra03t" />
      <input value="log ra03t[enter]" row="0" col="0" movecursor="true"
xlatehostkeys="true" encrypted="false" />
    </actions>
    <nextscreens timeout="0" >
      <nextscreen name="Screen2" />
    </nextscreens>
  </screen>
```

```
  <screen name="Screen2" entryscreen="false" exitsscreen="false" transient="false">
    <description>
      <oia status="NOTINHIBITED" optional="false" invertmatch="false" />
      <numfields number="3" optional="false" invertmatch="false" />
      <numinputfields number="3" optional="false" invertmatch="false" />
    </description>
    <actions>
      <input value="$USR.ID$" row="2" col="1" movecursor="true" xlatehostkeys="true"
encrypted="false" />
    </actions>
  </screen>
```

```

        <input value="[enter]" row="0" col="0" movecursor="true" xlatehostkeys="true"
encrypted="false" />
    </actions>
    <nextscreens timeout="0" >
        <nextscreen name="Screen3" />
    </nextscreens>
</screen>

<screen name="Screen3" entryscreen="false" exitsscreen="false" transient="false">
    <description>
        <oia status="NOTINHIBITED" optional="false" invertmatch="false" />
        <numfields number="55" optional="false" invertmatch="false" />
        <numinputfields number="11" optional="false" invertmatch="false" />
    </description>
    <actions>
        <input value="$PSS.WD$" row="8" col="20" movecursor="true" xlatehostkeys="true"
encrypted="false" />
        <input value="[enter]" row="0" col="0" movecursor="true" xlatehostkeys="true"
encrypted="false" />
    </actions>
    <nextscreens timeout="0" >
        <nextscreen name="Screen4" />
    </nextscreens>
</screen>
</HAScript>
<HAScript name="tso03" description="Express Logon to TS003" timeout="60000" pausetime="300"
promptall="true" author="" creationdate="" supressclearevents="false" >

    <screen name="Screen1" entryscreen="true" exitsscreen="false" transient="false">
        <description>
            <oia status="DONTCARE" optional="false" invertmatch="false" />
            <numfields number="1" optional="false" invertmatch="false" />
            <numinputfields number="1" optional="false" invertmatch="false" />
        </description>
        <actions>
            <custom id="Application_ID" args="ra03t" />
            <input value="log ra03t[enter]" row="0" col="0" movecursor="true"
xlatehostkeys="true" encrypted="false" />
        </actions>
        <nextscreens timeout="0" >
            <nextscreen name="Screen2" />
        </nextscreens>
    </screen>

    <screen name="Screen2" entryscreen="false" exitsscreen="false" transient="false">
        <description>
            <oia status="NOTINHIBITED" optional="false" invertmatch="false" />
            <numfields number="3" optional="false" invertmatch="false" />
            <numinputfields number="3" optional="false" invertmatch="false" />
        </description>
        <actions>
            <input value="$USR.ID$" row="2" col="1" movecursor="true" xlatehostkeys="true"
encrypted="false" />
            <input value="[enter]" row="0" col="0" movecursor="true" xlatehostkeys="true"
encrypted="false" />
        </actions>
        <nextscreens timeout="0" >
            <nextscreen name="Screen3" />
        </nextscreens>
    </screen>

```

```

<screen name="Screen3" entryscreen="false" exitsscreen="false" transient="false">
  <description>
    <oia status="NOTINHIBITED" optional="false" invertmatch="false" />
    <numfields number="55" optional="false" invertmatch="false" />
    <numinputfields number="11" optional="false" invertmatch="false" />
  </description>
  <actions>
    <input value="$PSS.WD$" row="8" col="20" movecursor="true" xlatehostkeys="true"
encrypted="false" />
    <input value="[enter]" row="0" col="0" movecursor="true" xlatehostkeys="true"
encrypted="false" />
  </actions>
  <nextscreens timeout="0" >
    <nextscreen name="Screen4" />
  </nextscreens>
</screen>

</HAScript>

```

Archived



z/OS 1.2 LDAP/WebSphere Application Server

In this chapter we describe how we installed LDAP with WebSphere for z/OS 4.0.1.

6.1 LDAP and WebSphere 4.0.1

The platform we used for this installation was based on the following product releases and versions:

- ▶ z/OS V1R2
- ▶ DB2 UDB V7
- ▶ WebSphere Application Server 4.0.1
- ▶ Java JDK 1.3.1

The objective was to migrate the LDAP backend used by WebSphere Application Server 4.0.1 from RDBM to TDBM.

The LDAP server for OS/390 Release 10 introduces a new DB2 backend database (TDBM). The TDBM database implementation provides greater scalability than the RDBM database implementation shipped in prior releases. RDBM is still available for compatibility; however, IBM recommends using TDBM.

TDBM allows schema to be changed dynamically through LDAP protocol, schema discovery, and publication.

TDBM, as introduced in OS/390 R10 and subsequent releases, provides many new functions and features, among which are:

- ▶ Schema discovery and schema publication instructions for use with TDBM.
- ▶ Dynamic schema modification information for use with TDBM.
- ▶ A new load facility, called **ldif2tdbm**, used for loading a large number of directory entries into a TDBM database.
- ▶ A new unload facility, called **tdbm2ldif**, used for unloading entries from a TDBM database.
- ▶ Greater scalability is achieved when TDBM is used in the recommended way.
- ▶ A new configuration option, `sslCertificate`.
- ▶ Native authentication support provided by APAR OW47596.

All publications and sample jobs provided by WebSphere Application Server only document the setup and customization of an RDBM backend for WebSphere Application Server.

This chapter describes the installation and implementation of WebSphere Application Server 4.0.1 WebSphere for z/OS, but the focus is primarily on the LDAP part of the installation. Refer to *WebSphere Application Server for OS/390: Application Server Planning, Installing, and Using*, GC34-4757 for step-by-step procedures to install and customize WebSphere Application Server 4.0.1.

6.2 WebSphere Application Server 4.0.1 overview

WebSphere Application Server V4.0.1 for z/OS and OS/390 is an e-business application deployment environment built on open standards-based technology. It is the cornerstone of WebSphere application offerings and services, providing support for servlets, JavaServer Pages (JSPs), and Enterprise Java Beans (EJB s), in compliance with Java 2 Enterprise Edition (J2EE) specifications. Version 4 brought together the standard and enterprise editions into a single product, providing a single offering.

WebSphere Application Server V4.0.1 for z/OS and OS/390 continues to support connector access to CICS, IMS, and DB2, as well as support to connect among applications running on Web servers, CORBA application servers, and back-end systems such as DB2, CICS, and IMS. Included in this support is Java Developer's Kit (JDK) 1.3.0, which provides the base support for applications at the Java 2 API level.

WebSphere Application Server V4.0.1 for z/OS and OS/390 now includes support for the following:

- ▶ A new configuration option offering more flexible implementation
- ▶ An EJB production environment with J2EE compliance
- ▶ Java Messaging Service (JMS), JavaMail, and Client
- ▶ Container Web services equivalent to those in the Version 4 distributed family of servers
- ▶ Web services with SOAP and UDDI open standards support servlets, JSPs, and EJBs in compliance with J2EE specifications

The features of WebSphere Application Server V4.0. and 4.0.1 for z/OS and OS/390 include:

- Support for deployment of J2EE, including:
 - EJB Version 1.1 specification level
 - Deployment of Java servlets written to the JavaSoft Version 2.2 specification level
 - Deployment of JSPs written to the JavaSoft Version 1.1 specification level
- Remote method invocation (RMI) using Internet inter-ORB protocol (IIOP)
- Object by value
- Access to J2EE services from Java Business Objects
- Java Transaction API (JTA)
- Integrated run time
- A highly secure Web deployment environment with Kerberos as the backbone and SSL as the endpoints
- Java RAS services integrated with existing RAS services and quality of services for z/OS and OS/390

It supports interoperability with:

- WebSphere Application Server for distributed platforms
- CICS TS V1.3, or later, for inbound IIOP support
- Products from other vendors that comply with the supported levels of the J2EE specification
 - It provides workstation-based support for administration and operations applications used to install, deploy, and manage WebSphere for z/OS applications.

- Software Development Kit (SDK) 1.3.0 provides the base support for applications at the Java 2 API level. You can use this level or one already installed at a higher service level.
- It is compliant with National Language Support (NLS) for z/OS-based products.

Through the incorporation of open industry standards (such as HTML, HTTP, IIOP, and CORBA) and J2EE-compliant Java technology standards for servlets, JSP technology, and EJBs, WebSphere Application Server v4.0.1 for z/OS and OS/390 enables new e-business applications and transactions to be scaled up for global deployment on the server of choice, regardless of the development platform.

New applications can be integrated with the extensive inventory of existing OS/390 applications. The result is the rapid creation of production-ready e-business applications, along with the ability to evolve to the EJB or CORBA-based deployment model of choice.

WebSphere Application Server V4.0.1 for z/OS includes a new configuration option designed to provide even faster installation and time to productivity for customers who are not yet planning to utilize capabilities of the full Java programming model. This option features minimum prerequisites because it does not require the functions of LDAP, workload management goal mode, or DB2 V7. (Note that DB2 V5 is required.)

Customers can easily grow their applications to utilize the full transactional and administrative capabilities of the WebSphere for z/OS product when their business demands.

WebSphere for z/OS provides superior price/performance for Java applications of all sizes and workload types. For example, a z/OS license provides the highest levels of reliability and scalability for the same leading WebSphere application server functionality as you can get on a distributed system, all for an equivalent or lower price.

6.2.1 Installing WebSphere Application Server 4.0.1

It is beyond the scope of this redbook to explain how to install WebSphere 4.0.1 on z/OS. Detailed instructions for this process are in *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization*, GA22-7834. It contains step-by-step procedures to activate the functions of the WebSphere Application Server.

Installing the WebSphere V4.0.1 runtime

To migrate from either WebSphere V3.02SE or V3.5SE to WebSphere V4.0.1 involves multiple changes in support of the J2EE servers introduced by WebSphere V4.0.1. You must set up the supporting infrastructures (DB2 7.1, RRS, WLM, LDAP, etc.), set up the system server structure (daemon, naming, system management, etc.), and utilize the new administration application to define and execute the installation verification programs (IVPs).

These changes require you to do a cold start and initialize the WebSphere 4.0.1 runtime as if it was your first installation.

Note: Release changes after WebSphere for z/OS V4.0 do not require the cold start method.

Running the customization dialog

The customization dialog is used by the system programmer or administrator responsible for installing and customizing WebSphere for z/OS. For more information see “Running the customization dialog” in *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization*, GA22-7834. The dialog covers a portion of WebSphere for z/OS customization. Specifically, it creates tailored jobs to:

- Copy the generated jobs into your system libraries

- ▶ Create the system management HFS structure and the initial environment file
- ▶ Create and customize the LDAP server
- ▶ Set up WebSphere for z/OS security controls (RACF)
- ▶ Define the WebSphere for z/OS runtime configuration (systems management server, naming server, interface repository server, daemon server)
- ▶ Run the installation verification programs (IVPs)

6.2.2 WebSphere Application Server 4.0.1 and LDAP

This section describes the relationships between WebSphere for z/OS, DB2, and LDAP. For WebSphere for z/OS, the LDAP component of the z/OS or OS/390 Security Server provides the directory services for the Java Naming and Directory Interface (JNDI) and CORBA naming and interface repository services. The contents of the directory are stored in DB2 tables.

At run time, your J2EE components require an LDAP server to be running for the JNDI name services. We recommend that you use the LDAP server you create during installation and customization for this purpose. CORBA (MOFW) components do not require an LDAP server to be running because they rely on the Naming Server, which runs the LDAP DLLs in its own address space. In both cases, you need an LDAP server for administrative purposes, such as adding users to the LDAP access control list.

Even if you already have an LDAP server on your system, we recommend that you create a new LDAP server and database for WebSphere for z/OS. The reasons are:

- ▶ The data you put in the database is of interest only to WebSphere for z/OS and accessible through WebSphere for z/OS services.
- ▶ An exclusive LDAP server and database helps you keep the WebSphere for z/OS databases synchronized.

Important: If you have an existing WebSphere Application Server Enterprise Edition for OS/390 V3.02 LDAP database, schema changes require that you migrate that database using an unload/reload operation.

For in-depth instructions for setting up LDAP, refer to *z/OS Security Server LDAP Server Administration and Use*, SC24-5923.

During installation and customization, you must create an LDAP server (or use an existing LDAP server), create the LDAP database, run bind jobs, set DB2 grants, and initialize the LDAP directories. Details are in the customized instructions produced by the customization dialog.

Structure of the LDAP configuration files

The main LDAP configuration file, `system.bboslapd.conf` uses include statements to include the other configuration files, containing all the static schema definitions that will be supported by the RDBM LDAP server. Typical LDAP configuration files also include a `dsnaoini` statement, which points to the DSNAOINI data set, the DB2 initialization file. However, in order to place our version of DSNAOINI into the HFS, the start procedures for LDAP, the Naming server region, and the Interface Repository server region must point to DSNAOINI through a DD statement (our samples do that for you). When you use such a DD statement in the start procedures, you do not need to use the `dsnaoini` statement in the LDAP configuration file. Thus, we comment out the `dsnaoini` statement in `bboslapd.conf`.

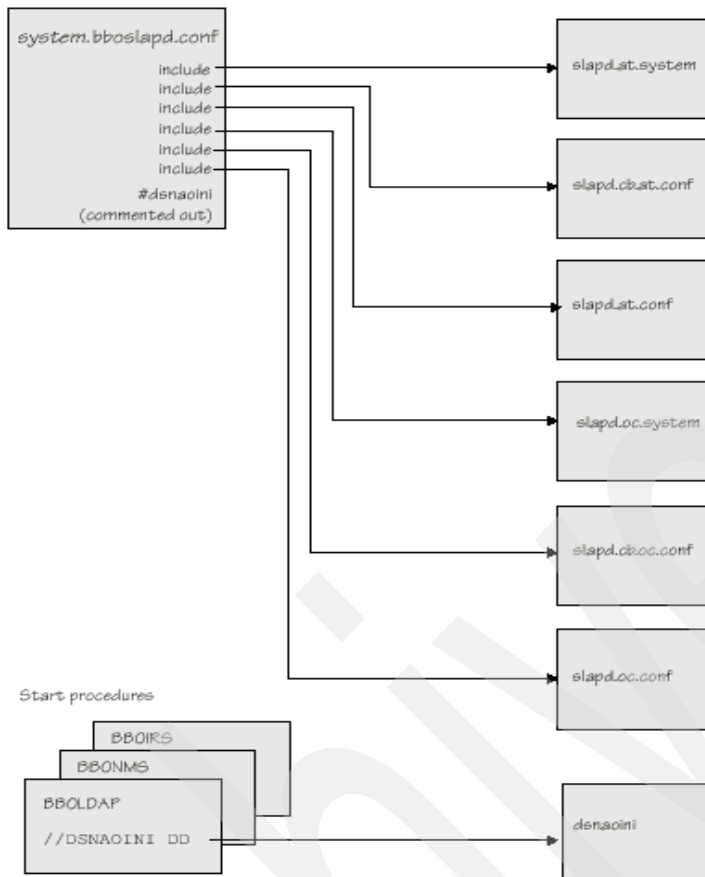


Figure 6-1 LDAP configuration file structure used by the RDBM LDAP sever

Guidelines, rules, and recommendations for DB2 and LDAP

1	DATE FORMAT	====>	ISO	ISO, JIS, USA, EUR, LOCAL
2	TIME FORMAT	====>	ISO	ISO, JIS, USA, EUR, LOCAL
3	LOCAL DATE LENGTH	====>	0	10-254 or 0 for no exit
4	LOCAL TIME LENGTH	====>	0	8-254 or 0 for no exit
5	STD SQL LANGUAGE	====>	NO	NO or YES
6	CURRENT DEGREE	====>	1	1 or ANY
7	CACHE DYNAMIC SQL	====>	NO	NO or YES
8	OPTIMIZATION HINTS	====>	NO	Enable optimization hints. NO or YES
9	VARCHAR FROM INDEX	====>	NO	Get VARCHAR data from index. NO or YES
10	RELEASE LOCKS	====>	YES	Release cursor with hold locks. YES,NO

Figure 6-2 Guidelines for DB2 Params

Follow these guidelines and recommendations to set up DB2 and LDAP:

- ▶ The CORBA support in WebSphere for z/OS does not support DBCS. You must configure DB2 with the DSNHDECP parameter MIXED=NO.
- ▶ LDAP RDBM support in WebSphere for z/OS requires you to configure DB2 with DSNHDECP parameter MIXED=NO.
- ▶ If you are configuring a J2EE server and you would like to support DBCS data, configure DB2 with the DSNHDECP parameter MIXED=YES and use the LDAP TDBM backend. The RDBM backend does not support MIXED=YES.

- ▶ Check the size of your DB2 logs. They might need to be larger because of the number of transactions WebSphere for z/OS generates.
- ▶ Increase the BP32K buffer pools to at least 100.
- ▶ Check the size of your DSNDB07 database.
- ▶ Check the 32K temporary work space for DB2. Your installation may not have had use for this work space before, but WebSphere for z/OS uses it.
- ▶ You must run a DB2 job called DSNTIJTM (in hlq.SDSNSAMP) during DB2 installation to allocate the work space. If this allocation is not large enough, you may get an SQL -904 return code when bringing up the LDAP server, the System Management Server, or the Naming Server.
- ▶ Take note of the fact that WebSphere for z/OS uses row-level locking and Type 2 indexes.
- ▶ If possible, keep the WebSphere for z/OS LDAP tables separate from other LDAP tables. The reason for keeping the sets of LDAP tables separate is that you need to back up the WebSphere for z/OS LDAP tables with the WebSphere for z/OS system management database as a unit. Performing such a coordinated backup is easier if the WebSphere for z/OS LDAP tables are separate from other LDAP tables. Additionally, if you need to restore the WebSphere for z/OS environment, restoring the WebSphere for z/OS LDAP tables will not interfere with LDAP tables used by other applications.

Access to naming services is controlled and managed by LDAP access control lists. The sample LDIF file we provide (bboldif.cb) provides two LDAP access IDs with write access to the name space: CBAAdmin and WASAdmin. Because they have write access, you may want to change the administrative password in the LDIF file.

If you change the password for CBAAdmin (you can do this through the customization dialog at installation and customization time), you must update the LDAPBINDPW environment variable for the Naming server and the LDAPIRBINDPW environment variable for the Interface Repository server. Update the environment variable in the current.env file for each server.

Note: General run-time name lookup requires read access to the name space. The sample LDIF file provides an access ID with read access called ANYBODY, which allows any user to access name services.

Rules for LDAP security

You can control access to LDAP directories, subdirectories, or entries by means of access control lists (ACLs). ACLs specify which users are allowed access to each LDAP entry and which types of operations those users may perform. For details, see *z/OS Security Server LDAP Server Administration and Use*, SC24-5923. Follow these rules regarding LDAP security:

- ▶ For CORBA components, IBM has configured the LDAP DLLs to run within the Naming and Interface Repository server instances, thus eliminating the need to have a separate LDAP server running with the WebSphere for z/OS run time.

If you do not follow the standard configuration of running the LDAP DLLs in the Naming and Interface Repository server instances, and rely on an LDAP server running with WebSphere for z/OS run time, do not implement ACL-based access control to WebSphere for z/OS data. If you do implement ACL-based access control with such a configuration, WebSphere for z/OS will not be able to access its data.

- ▶ You can use RACF user IDs in LDAP ACLs.

Example: If USER1 is a RACF user ID, use the following ACL statement. It gives USER1 the maximum access rights to the specified LDAP entry.

```
aclSource: cn=DEPT_A, o=IBM, c=US
aclEntry: access-id:USER1:object:ad:normal:rws
```

You cannot, however, use RACF group names in this way. For more information about this and how LDAP can access the RACF database, see *z/OS Security Server LDAP Server Administration and Use*, SC24-5923. If you use group names, your installation must place WebSphere for z/OS libraries, DB2 libraries, and SYS1.LINKLIB under program control.

For your initial LDAP configuration, we recommend that you do not set up LDAP with RACF group names.

6.2.3 Installation dialog

The customization dialog is an ISPF dialog that eliminates the need to hand-tailor sample jobs supplied with the product. You define the customization options once in the dialog panels, then the dialog generates the jobs with your options, eliminating the need to define them in several places. The benefit to you is reduced typos and inconsistencies, and a quicker customization.

Perform the following steps to run the customization dialog:

1. From the ISPF command line, enter the following command:

```
hlq.sbboclib(bbowstrt) ' options'
```

where:

- hlq is the high-level qualifier for the SBBCLIB data set.
- options are command options. Enclose any and all options in a single set of quotes
- hlq(value) specifies the data set qualifier(s) for the WebSphere for z/OS product data sets. The default value is BB0.
- appl(value) specifies the ISPF application name. The default value is BB0.
- lang(value) specifies the national language. Values can be either ENUS (English) or JAPN (Japanese). The default is ENUS.

Example:

```
'bbo.sbboclib(bbowstrt)' 'hlq(bbo) appl(bbo) lang(enus)'
```

You should then be presented with the screen shown in Figure 6-3 on page 185.

```
Option ==>                               Appl: BBO

Use this dialog to customize WebSphere for z/OS for the first time
or to migrate releases. Specify an option and press ENTER.

1 New customization. If you are customizing WebSphere for
  z/OS for the first time, use this option.

2 Migration with saved variables. If you have previously saved the
  customization variables using the dialog, use this option to
  migrate from WebSphere for z/OS V4.0 to V4.0.1.

3 Migration without saved variables. If you have never run the
  customization dialog, or have not previously saved the
  customization variables, use this option to migrate from
  WebSphere for z/OS V4.0 to V4.0.1.
```

Figure 6-3 WebSphere for z/OS customization: First screen

Select option **1** for a new installation.

```
----- WebSphere for z/OS Customization -----
Option ==>                               Appl: BBO
New Customization
Use this dialog to define WebSphere for z/OS variables and generate
customization jobs for your installation. Specify the HLQ for
WebSphere product data sets, an option, and press ENTER.

HLQ for WebSphere product data sets: WAS.V401.PID

1 Allocate target data sets. The data sets will contain the
  WebSphere customization jobs and data generated by the dialog.

2 Define variables. Define your installation-specific information
  for WebSphere customization.

3 Generate customization jobs. Validate your customization
  variables and generate jobs and instructions.

4 View the generated customization instructions.

Options for WebSphere Customization Variables

S Save customization variables. Save your WebSphere
  customization variables in a data set for later use.

L Load customization variables. Load your WebSphere
  customization variables from a data set.
```

Figure 6-4 WebSphere for z/OS customization: New customization

Then go through all the options in sequence. This dialog is very similar in nature to the serverPac or ProductPac installation dialog.

Basically, you first define all your options, which will in turn generate variables used in the generated JCL members.

The specific part for LDAP is accessed from option **2**, then **5**.

```
----- WebSphere for z/OS Customization -----
Option  ===>

Define Variables

Specify a number and press ENTER to define the WebSphere variables.
You should review all of the variables in each of the sections, even
if you are using all of the IBM-supplied defaults.
Once you complete all sections, press PF3 to return to the main menu.

1 - System Locations (directories, HLQs, etc)
2 - WebSphere Customization
3 - Server Customization
4 - IVP Customization
5 - LDAP Customization
6 - Security Customization

Changed?
Y
Y
```

Figure 6-5 WebSphere for z/OS customization: Define variables screen

Lightweight Directory Access Protocol (LDAP) provides the directory services for the JNDI and CORBA naming and interface repository services. For the WebSphere for z/OS implementation, LDAP stores its naming data in a DB2 database.

The dialog provides most of the LDAP configuration data, but you must specify the LDAP server definitions and some configuration information on this panel.


```

----- WebSphere for z/OS Customization -----
Option ==>

LDAP Customization (1 of 1)

LDAP Server Definitions

  Procedure name.: BBOLDAP
  Userid.....: CBLDAP
  UID.....: 2119
  Group.....: CBLDAPGP
  GID.....: 2219

LDAP Configuration Information

  IP Name.....: wasv4.secsres.pssc
  IP Port.....: 1389
  Administrator user DN..: cn=CBAdmin
  Administrator user pw..: secret

  DB2 STOGROUP value.....: BBOLDSTO
  DB2 database name.....: BBOLDAP
  Authid for DB2 tables...: BBOLDAP

LDAP Tablespace Information

  Tablespace for LDAP entry table...: BBOENT
  Tablespace for 32K tables.....: BB032K
  Tablespace for 4K tables.....: BB04K
  Tablespace for 4K mutex table.....: BB0MUTX

```

Figure 6-6 Websphere for z/OS customization: LDAP customization screen

6.2.4 Problems encountered and lessons learned

One of the problems we had during the installation was with the JDK 1.3.1 level we were using.

The **java -version** UNIX System Services command displays:

Example 6-1 Java version command display

```

java version "1.3.1"
    Java(TM) 2 Runtime Environment,
        Standard Edition (build 1.3.1)
    Classic VM (build 1.3.1, J2RE 1.3.1)
    IBM OS/390 Persistent Reusable VM
    build hm131s-20011015
    (JIT enabled: jitc)

```

When we tried to start the sample server BBOASR2, we got the following in the BBOASR2S joblog:

```

"CEE3501S The module Xm.dll was not found."
From compile unit /u/sovbld/am131s/am131s-20011004/src/awt/pfm/ /VDrawingArea.c at
entry point __dllstaticinit at compile unit offset +000001E8 at entry offset +000001E8
at address 3071C618.

```

We found APAR II13000, which explains the JVM's need for these X libraries. There is apparently some change to the JVM internals that causes this to be the case. (It is probably the result of the loss of the statically bound libawt.so, resulting in the use of the TCPIP supplied dlls.)

As per the generic APAR for SR11 (UQ99325), PQ52841, AWT applications require LIBPATH to include Xm.dll. The APAR refers to II13000, which states that Java 1.3 PTF SR11 (APAR PQ52841/PTF UQ99325) makes a change to AWT operation. At this service refresh, and for later service levels, the X libraries are no longer statically bound with the JVM libawt.so dll.

This means that for correct JVM operation, the X libraries must be available to the JVM via the LIBPATH envvar. The launcher adds the normal location (/usr/lib) to the LIBPATH, but if your installation moves the X libraries from this default location, you need to modify LIBPATH accordingly.

This information was also included on the prereq section of the download page:

www-1.ibm.com/servers/eserver/zseries/software/java/wizard/help_prereqs.html

Xm.dll and X11.dll are shipped with other parts in /usr/lpp/tcpip/X11R6/lib

With JAVA upgrade to 1.3.1, the X11.dll and Xm.dll objects must be marked as program-authorized. APAR PQ54336 refers to II13000 and advises that the dlls are also marked program-controlled.

Refer also to APAR PQ54914, which describes what needs to be done to make these objects program-authorized.

We include the content of the three mentioned APARs, PQ54914, II13000, and PQ52841 in the following three examples.

Example 6-2 APAR PQ54914

SYSRES=	SYSIN=	SYSOUT=	CPU=	RE-IPL=	ACTIVE
OPTYPE=		SPECIAL	ACTIVITY=	REGRESSION=	OPTIONS ARE:
PRE-SCREEN NO.=		RSCP=	RS038		
ERROR DESCRIPTION:					
If customer has the Java 2 SDK 1.3.1 upgrade (PQ52841) or higher, the following modules must be program authorized.					
Xm.dll and X11.dll					
.					
These modules reside in the /usr/lpp/tcpip/X11R6/lib subdirectory of TCPIP.					
.					
For existing releases of TCPIP, refer to the Local Fix section on how to make these objects program authorized.				FULL PAGE= ON	
LOCAL FIX:				SUBMITTER	
User issuing the command to turn on the program authorized bit must have the following authority:				PG 1, 2 OF 5	
.					
Super User (UID 0)				APAR= PQ54914	
Read access to BPX.FILEATTR.APF in your security product (eg. RACF).				OWNED BY: SF2	
Read/Write access to the directory the X11.dll and Xm.dll objects are located in.				ACTIVE	
.				OPTIONS ARE:	
With this authority, change to the directory the objects are in and issue the following command to set the appropriate setting.					
.					

```
extattr +p X11.dll Xm.dll
.
```

Once set, the following command can be issued for each object

```
to verify the setting.
.
USER1:/usr/lpp/tcpip/X11R6/lib: >extattr X11.dll
X11.dll
APF authorized = NO
Program controlled = YES      <--- This must be YES
Shared address space = YES
Shared library = NO
.
USER1:/usr/lpp/tcpip/X11R6/lib: >extattr Xm.dll
Xm.dll
APF authorized = NO
Program controlled = YES
Shared address space = YES
Shared library = NO
```

FULL PAGE= ON
SUBMITTER
PG 3, 4 OF

APAR= PQ54914
OWNED BY: SF2

ACTIVE
OPTIONS ARE:

Example 6-3 APAR II13000

APAR= II13000	SER=	IN INCORROUT	ACTIVE
JAVA FOR OS/390 AND Z/OS:AWT APPLICATIONS NOW REQUIRE LIBPATH			OPTIONS ARE:
TO INCLUDE LOCATION OF XM.DLL (USUALLY /USR/LIB) AFTER SR11			PRINT
STAT= INTRAN	FESN5NF0000-000	CTID= II0000	ISEV= 4
SB01/10/25	RC	CL	PD
		PE=	SEV= 4
			TYPE= I
RCOMP= INFOPALIB	PA LIB INFO	ITE	RREL= R001
FCOMP=			PFREL= F
ACTION=	SEC/INT=		TREL= T
USPTF=	PDPTF=		DUP/
			DUPS 0

DW01/10/25	RT	SC	FT	RE	
PT	UP	LP	PV	AP	FULL PAGE= ON
EN	FL	LC01/11/09	RU01/10/25	OT	SUMMARY TEXT
CT	FR	TD	TYPE OF SOLUTION=		PG 1, 2 OF 3
PROJECTED CLOSE CODE=		CUST INST LVL/SU=			
FAILING MODULE:		FAILING LVL/SU=			
SYSROUTE OF:		RET APAR=	PS=	APAR= II13000	
STATUS DETAIL=		RELIEF AVAILABLE=			
COMP OPER ENV=					
ERROR DESCRIPTION:					
Java 1.3 PTF SR11 (APAR PQ52841/PTF UQ99325) makes a change					
to AWT operation. At this service refresh, and for later service					
levels, the X libraries are no longer statically bound with					
the JVM libawt.so dll.					
This means that for correct JVM operation, the X libraries must					
be available to the JVM via the LIBPATH envvar.					

The launcher adds the normal location (/usr/lib) to the LIBPATH but if your installation moves the X libraries from this

FULL PAGE= ON

default location, you will need to modify LIBPATH accordingly.
LOCAL FIX:

SUBMITTER
PG 1, 2 OF 2

Additional Information:
It may be required that these dlls are program controlled.

APAR= II13000

Example 6-4 APAR PQ52841

APAR= PQ52841 SER= NF FUNCTION ACTIVE
JAVA 2 - SDK - UPGRADE TO 1.3.1 INCL. PERSISTENT REUSABLE JVM, OPTIONS ARE:
SECURITY, STORAGE MGMT, PERF. ENHANCEMENTS, SERVICE REFRESH (SR11) PRINT
STAT= CLOSED UR1 FESN0506095- CTID= HU0200 ISEV= 4
SB01/09/28 RC01/09/28 CL01/10/31 PD SEV= 4
PE= SPEC/ATTN/Y TYPE= D
RCOMP= 5648C9801 JAVA2(1.3) OS/3 RREL= R130
FCOMP= 5648C9801 JAVA2(1.3) OS/3 PFREL= F TREL= T999
ACTION= SEC/INT= N DUP/
USPTF= PDPTF= DUPS 1

DW01/09/28 RT01/09/28 SC FT RE
PT UP LP PV01/10/31 AP FULL PAGE= ON
EN FL LC01/11/13 RU01/11/13 OT01/12/13 SUMMARY TEXT
CT01/12/13 FR TD TYPE OF SOLUTION= PG 1, 2 OF 3
PROJECTED CLOSE CODE= CUST INST LVL/SU= 130
FAILING MODULE: AJVTAR13 FAILING LVL/SU= 130
SYSROUTE OF: RET APAR= PS= S APAR= PQ52841
STATUS DETAIL= SHIPMENT RELIEF AVAILABLE=
COMP OPER ENV= 130
Stop Press: AWT applications require LIBPATH to Xm.dll see OPTIONS ARE:
information APAR: II13000

Index of Information APARs relating to Java on OS/390 & z/OS
is contained in information APAR II12998
PROBLEM CONCLUSION:
TEMPORARY FIX:
COMMENTS:
APARs fixed by applying SR11 APAR PQ52841

***** PE01/11/05 PTF IN ERROR. SEE APAR PQ54336 FOR DESCRIPTION
MODULES/MACROS: AJVTAR13 FULL PAGE= ON
SRLS: SC34603401 RESPONDER
RTN CODES: PG 9,10 OF 10
APPLICABLE COMPONENT LEVEL/SU:
R130 PSY UQ99325 UP01/11/02 I 1000
CIRCUMVENTION: APAR= PQ52841
MESSAGE TO SUBMITTER:

Important: All customers upgrading WebSphere Application Server from 4.0.0 to 4.0.1 on OS/390 V2.10, and at the same time upgrading Java from 1.3.0 to 1.3.1, will have the same problem, that WebSphere Application Server will not find Xm.dll.

Also, it is worth mentioning that the PSP bucket discourages customers from using JDK 1.3.1S at this time because of PQ54336 and PQ54725. The recommended level is SR10 (UQ58208).

To quickly test if this was the problem all along, we updated the /WebSphere390/CB390/controlinfo/envfile/RDLPAR/SYSMGT01/current.env LIBPATH statement to include /usr/lib and recycled the BBOSMSS instance.

This did not resolve the problem, so we decided to put /usr/lib in the libpath for all our servers, and recycled DAEMON01 to ensure the change was picked up. However, BBONMS would get the following error:

```
BBOU0539E LDAP INITIALIZATION FAILED OR CONNECTION TO LDAP SERVER COULD NOT BE ESTABLISHED.
```

Also, we would get the message "+BBOU0539E LDAP INITIALIZATION FAILED OR CONNECTION TO LDAP SERVER COULD NOT BE ESTABLISHED" when we attempted to connect with the SMEUI.

When running BBOIVPE, the job fails and the following is seen in the job output:

Example 6-5 BBOIVPE Job Output

```
***** bmp bean will be run$
Look up policy session home
Obtaining polycysession bean jndi name...
The polycysession bean jndi name is: /WebSphere390/CB390/apps/BB0ASR2/
PolicyIVP/polycysession_deploy/ivp.polycysession/com.ibm.ws390
.samples.ivp.ejb.PolicySessionHome
***** cmp bean will be run$
Look up policy session home
Obtaining polycysession bean jndi name...
The polycysession bean jndi name is: /WebSphere390/CB390/apps/BB0ASR2/
PolicyIVP/polycysession_deploy/ivp.polycysession/com.ibm.ws390
.samples.ivp.ejb.PolicySessionHome
IVP has failed
READY
BBOHFSWR '/SYSTEM/tmp/ejbivp.err'
java.lang.NullPointerException
.at java.util.Hashtable.put(Hashtable.java:386)
.at java.util.Properties.setProperty(Properties.java:113)
.at com.ibm.ws390.samples.ivp.client.TestClient.main(TestClient.java:84)
java.lang.NullPointerException
.at java.util.Hashtable.put(Hashtable.java:386)
.at java.util.Properties.setProperty(Properties.java:113)
.at com.ibm.ws390.samples.ivp.client.TestClient.main(TestClient.java:84)
```

We backed out the change to /NAMING01/current.env and recycled DAEMON01.

This solved the LDAP connection failure from the SMEUI, so we were able to start our application servers and the naming registrations succeeded on all of them.

So, by removing the /usr/lib from BBONM's current.env we corrected the problem; however, we have no idea why this fixed the problem.

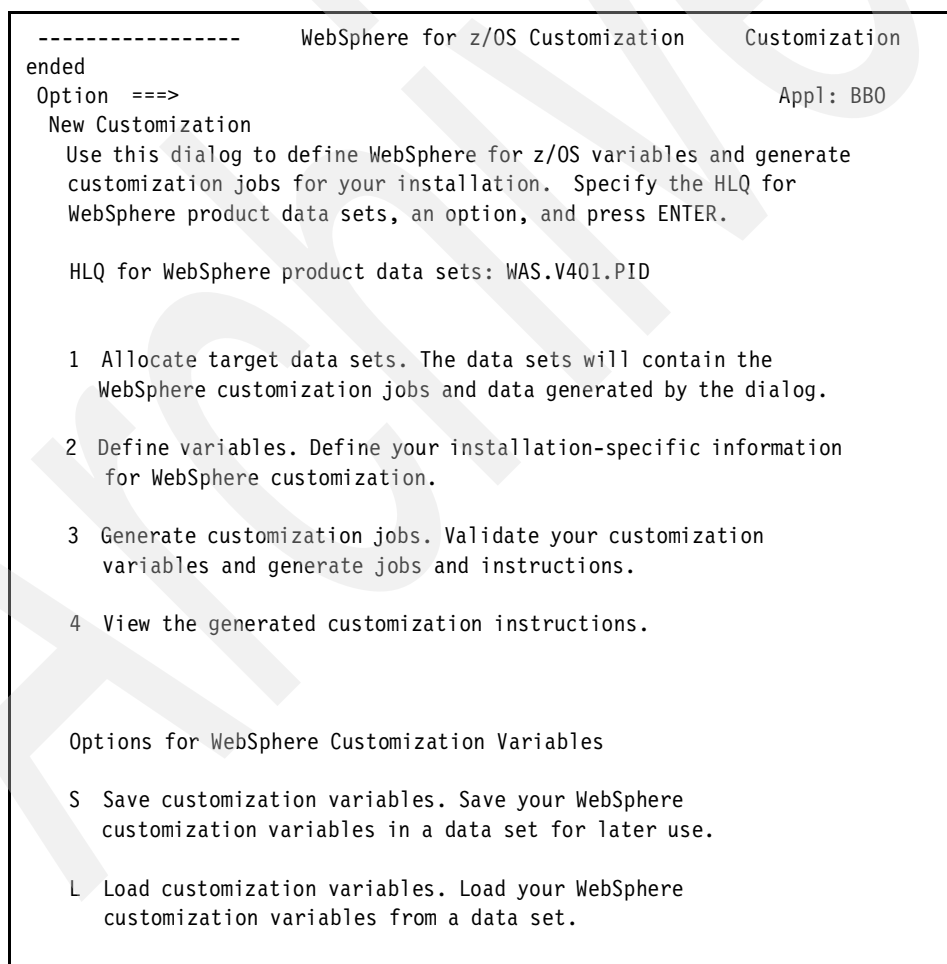
We planned on updating the global current.env on the sysplex definition via the SMEUI to include the /usr/lib in the LIBPATH statement as recommended by APAR II13000, but at the current level this requires some manual updates in the current.env files to remove it from the NAMING and IR servers.

Important: Our conclusions are as follows:

If you choose to go with JDK SR10, then you won't need to change your LIBPATH in your environmental variables. But if you stay with JDK SR11 you will need to configure the LIBPATH of various WebSphere Application Server and appl.servers. Adding usr/lib to LIBPATH in the global SYSPLEX env.variables via the SMEUI is the first step.

The next step is to go to the Naming and IR server definitions, via the SMEUI, and add the same LIBPATH variable, but edit the value to remove the usr/lib. (The previously described LDAP INITIALIZATION FAILED message is an example of what happens if the usr/lib is in the Naming LIBPATH.)

The installation dialog related to LDAP is accessed from option **2** (define variables) on panel BBOWMAIN, shown in Figure 6-7.



```
----- WebSphere for z/OS Customization Customization
ended
Option ==> Appl: BBO
New Customization
  Use this dialog to define WebSphere for z/OS variables and generate
  customization jobs for your installation. Specify the HLQ for
  WebSphere product data sets, an option, and press ENTER.

  HLQ for WebSphere product data sets: WAS.V401.PID

  1 Allocate target data sets. The data sets will contain the
  WebSphere customization jobs and data generated by the dialog.

  2 Define variables. Define your installation-specific information
  for WebSphere customization.

  3 Generate customization jobs. Validate your customization
  variables and generate jobs and instructions.

  4 View the generated customization instructions.

Options for WebSphere Customization Variables

S Save customization variables. Save your WebSphere
  customization variables in a data set for later use.

L Load customization variables. Load your WebSphere
  customization variables from a data set.
```

Figure 6-7 Panel BBOWMAIN

Select option **5** LDAP customization on panel BBOWCUS1 (see Figure 6-8 on page 193).

```

. . . . .
BBOWCUS1 ----- WebSphere for z/OS Customization -----
Option ===>

Define Variables

Specify a number and press ENTER to define the WebSphere variables.
You should review all of the variables in each of the sections, even
if you are using all of the IBM-supplied defaults.
Once you complete all sections, press PF3 to return to the main menu.


1 - System Locations (directories, HLQs, etc)
2 - WebSphere Customization
3 - Server Customization
4 - IVP Customization
5 - LDAP Customization
6 - Security Customization

Changed?
Y
Y

```

Figure 6-8 Panel BBOWCUS1: Select option 5

You should use the proposed default values.

Important: When using option 2 WebSphere customization, you are asked to supply some RACF groups and user IDs that are common throughout WebSphere for z/OS. The dialog creates the RACF commands to define these new user IDs and groups for your security system.

The job will be created as member BBOWBARK in data set hlq.WAS.DATA.

Check this job carefully to make sure it will fit your security requirements and rules. The job may fail because of the ALU commands to reset the passwords. By default the passwords are reset to the userid value; this may not be allowed in the SETROPTS RACF rules in many shops, so make the necessary adjustments to the ALU commands.

```

EDIT          PRICHAR.WAS.DATA(BBOWBRAK) - 01.00          Columns 00001
00072
000062 "ALU CBADMIN PASSWORD(CBADMIN) NOEXPIRED"
000063 /* Adding users to run IVP1.
000064 "ADDUSER CBIVP DFLTGRP(CBIVPGP) OMVS(UID(2109) HOME(/tmp)
PROGRAM(/bin/sh) |
000065 "n/sh)) NAME('WAS IVP1 USER') "
000066 "PW USER(CBIVP) NOINTERVAL"
000067 "ALU CBIVP PASSWORD(CBIVP) NOEXPIRED"
000068 /* Adding users to run IVP2.
000069 "ADDUSER CBIVP2 DFLTGRP(CBIVPGP2) OMVS(UID(2117) HOME(/tmp)
PROGRAM(/bin/sh) |
000070 "bin/sh)) NAME('WAS IVP2 USER') "
000071 "PW USER(CBIVP2) NOINTERVAL"
000072 "ALU CBIVP2 PASSWORD(CBIVP2) NOEXPIRED"

```

Figure 6-9 Job BBOWBRAK

We ran into two problems:

1. A Java problem with JDK 1.3.1 (/usr/lib in LIBPATH)
2. Security definition problems because of the ALU commands in job BBOWBRAK

If you follow the installation and customization guide very carefully, then you should be able to get through the installation without any major problem, and you should be able to run the IVPs successfully.

```
. . . . .
BBOWCFG4 ----- WebSphere for z/OS Customization -----
Option ==>

WebSphere Customization (3 of 4)

WebSphere Common Groups and User IDs

Control region group for ALL servers... CBCTL1
Control region GID for ALL servers.... 2211
Server region group for base servers... CBSR1
Server region GID for base servers.... 2201

Unauthenticated User Definitions for Base Servers

Userid...: CBGUEST      UID...: 2102
Group....: CBCLGP      GID...: 2202

WebSphere Application Installer Group Information

Group....: CBCFG1      GID...: 2300

WebSphere Administrator Information

Userid...: CBADMIN      UID...: 2103
Password.: CBADMIN
Group....: CBADMGP      GID...: 2203
```

Figure 6-10 BBCHCFG4 screen

6.3 Migrating WebSphere Application Server 4.0.1 from RDBM to TDBM

The LDAP server comes with a TDBM backend database based on DB2. The TDBM database is a more scalable database implementation than the RDBM database which was shipped in prior releases. This change affects the amount of data the directory can store.

6.3.1 Migrating existing RDBM data to a TDBM database

Because TDBM has a different DB2 table design than RDBM, existing RDBM data must be unloaded and reloaded when migrating to TDBM. For the same reason, DB2 data sharing between TDBM and existing RDBM tables is not possible.

A high-level view of the steps to migrate from RDBM to TDBM follows. (Refer to the appropriate sections of *z/OS V1R2.0 Security Server LDAP Server Administration and Use*, SC24-5923 for the specific details of each step.)

1. Create a backup of the existing RDBM directory data using native DB2 utilities.

2. Unload the RDBM directory information using **db2ldif** and specifying the distinguished name of the root of the directory tree that is stored in DB2. As an example:

```
db2ldif -f /etc/ldap/slapd.conf -o /tmp/directorydata.unload -s "o=IBM,c=US"
```

 assuming this command was run from the z/OS shell prompt.
3. Obtain the DB_VERSION setting for the existing RDBM DB2 tables. If the DB_VERSION is 8.0 (you were already in OS/390 V2R8), skip to step 4. Otherwise, if you migrate from a pre-OS/390 V2R8 release, evaluate all comments (lines beginning with a pound sign (#)) that begin with AF, AP, DF, or DP in the LDIF file created in the previous step. If the attribute value or distinguished name described in this comment is acceptable to you, no further updates are necessary for that attribute value. If the attribute value is not acceptable to you, modify the uncommented LDIF line below the comments to contain an attribute value acceptable to you.
4. Optionally, remove the old DB2 tables. To do this, run the following SQLProcessor Using File Input (SPUFI) command:

```
DROP DATABASE databasename
```

 This step is optional because RDBM and TDBM databases can coexist.
5. Create the TDBM database and table spaces in DB2.
6. Make appropriate configuration changes.
7. Load the schema associated with the TDBM backend, which conforms to the LDAP Version 3 (V3) protocol. New schema that are consistent with earlier schema files have been shipped with z/OS in the V3 format. If you have added to or modified your existing schema files, you will need to migrate these files to the new format, either by using the migration utility or by making the same changes to the z/OS V3 schema files.
8. Load the data from the LDIF file just created using either the **ldif2tdbm** utility or the **ldapadd** utility.

A migration example

In this section we describe the actual steps we took to migrate from an RDBM to a TDBM configuration for a z/OS V1R2 system with WebSphere Application Server 4.0.1. The steps are numbered to correspond to those in the previous section.

1. Create a backup of the existing RDBM directory data using native DB2 utilities.
 You may want to run an imagecopy or any DB2 utility to make a backup of your LDAP DB2 tables.
 Figure 6-11 on page 196 is an example of a job that performs an image copy for each of the DB2 LDAP tablespaces created by the WebSphere Application Server installation dialog job (BBOLDTBC)

```

***** Top of Data *****
//PRICHAR2 JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHAR      29 NOV 2001      12:52:53
//*      -LIB- PRICHAR.LDAP.CNTL(IMGCPY)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//JOBLIB DD DISP=SHR,
//          DSN=DSN710.SDSNLOAD
//* ***** */
//* SYSUTILX IMAGE COPY STEP */
//* ***** */
//BBOENT EXEC PGM=DSNUTILB,PARM='DB1T,IMAGCOPY',REGION=1024K
//SYSCOPYX DD DSN=PRICHAR.IMAGCOPY.BBOENT,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSALLDA,
//            SPACE=(CYL,(10,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE BBOLDAP.BBOENT COPYDDN SYSCOPYX
//BBO4K EXEC PGM=DSNUTILB,PARM='DB1T,IMAGCOPY',REGION=1024K
//SYSCOPYX DD DSN=PRICHAR.IMAGCOPY.BBO4K,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSALLDA,
//            SPACE=(CYL,(10,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE BBOLDAP.BBO4K COPYDDN SYSCOPYX
//BBO32K EXEC PGM=DSNUTILB,PARM='DB1T,IMAGCOPY',REGION=1024K
//SYSCOPYX DD DSN=PRICHAR.IMAGCOPY.BBO32K,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSALLDA,
//            SPACE=(CYL,(10,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE BBOLDAP.BBO32K COPYDDN SYSCOPYX
//BBOMUTX EXEC PGM=DSNUTILB,PARM='DB1T,IMAGCOPY',REGION=1024K
//SYSCOPYX DD DSN=PRICHAR.IMAGCOPY.BBOMUTX,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSALLDA,
//            SPACE=(CYL,(10,10),RLSE)
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
COPY TABLESPACE BBOLDAP.BBOMUTX COPYDDN SYSCOPYX

```

Figure 6-11 Job BBOLDTBC image copy job

2. Unload the directory information using **db2ldif** and specifying the distinguished name of the root of the directory tree that is stored in DB2.

```

//PRICHARL JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//      NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1,USER=PRICHAR,
//      PASSWORD=V9ERONI
//*
//*
//*      -DOC- PRICHAR      29 Nov 2001      13:21:25
//*      -LIB- PRICHAR.LDAP.CNTL(DB2LDJ)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//*      -PURPOSE -run tdbm2ldif to unload the schema entry
//*                      or the whole directory
//*
// INCLUDE MEMBER=$SYSINCL
// SET SYSCONE=SYSTEM
//*
//STEP01 EXEC PGM=BPXBATCH,COND=(4000,LT),
//* the following is to unload the whole directory
// PARM='SH /usr/local/ldap/ldapunload.sh'
//STDOUT DD PATH='/&SYSCONE./tmp/out',
//          PATHOPTS=(OCREAT,OTRUNC,OWRONLY),PATHMODE=SIRWXU
//STDERR DD PATH='/&SYSCONE./tmp/error',
//          PATHOPTS=(OCREAT,OTRUNC,OWRONLY),PATHMODE=SIRWXU
//STDENV DD *
DSNAOINI=/WebSphere390/CB390/PORTINS0/etc/ldap/MVS0.dснаoini
_BPX_SPAWN_SCRIPT=YES
_BPX_SHAREAS=YES
//*
//STDIN DD PATH='/dev/null',
//          PATHOPTS=ORDONLY
//*
//ERRS04 EXEC PGM=IKJEFT1B,COND=(4000,LT)
//JES DD SYSOUT=*,DCB=(RECFM=V,LRECL=256)
//ERRORF DD PATH='/&SYSCONE./tmp/error',
//          PATHOPTS=ORDONLY
//OUT DD PATH='/&SYSCONE./tmp/out',
//          PATHOPTS=ORDONLY
//SYSTSPRT DD DUMMY
//SYSTSIN DD *
OCOPY INDD(ERRORF) OUTDD(JES)
OCOPY INDD(OUT) OUTDD(JES)
//*

```

Figure 6-12 Job to run DB2LDIF command

This sample job will run the **db2ldif** command in a shell script in batch mode. The second step of this job allows you to print the content of the STDOUT of the BPXBATCH step into the JES2 SYSOUT file of the job.

Figure 6-13 on page 198 shows the content of the script file `/usr/local/ldap/ldapunload.sh`.

```

BROWSE -- /usr/local/ldap/ldapunload.sh ----- Line 00000000 Col 0
Command ==>                                     Scroll ==
***** Top of Data *****
export DSNA0INI="/WebSphere390/CB390/PORTINS0/etc/ldap/MVS0.dsnaoini"
db2ldif -f /WebSphere390/CB390/PORTINS0/etc/ldap/MVS0.bboslapd.conf \
-o /usr/local/ldap/ldapload.ldif
***** Bottom of Data *****

```

Figure 6-13 Contents of script file `usr/local/ldap/ldapunload.sh`

The job output is shown in Figure 6-14.

```

GLD4005I Environment variable file not found. Environment variables not
set. C
GLD0010I Reading configuration file
/WebSphere390/CB390/PORTINS0/etc/ldap/MVS0.b
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.at.system.
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.cb.at.conf.
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.at.conf.
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.oc.system.
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.cb.oc.conf.
GLD0010I Reading configuration file /usr/lpp/ldap/etc/slapd.oc.conf.
GLD0181I The 'maxThreads' parameter is no longer supported.
GLD0129I The value for the 'maxConnections' option is out of range (30,
65535).
GLD0181I The 'waitingThreads' parameter is no longer supported.
GLD2062E The LDAP program will operate in single-server mode.
GLD2099I db2ldif: 2687 entries have been successfully read from the LDAP
directo

```

Figure 6-14 Job output from `db2ldif` command

Figure 6-15 shows the first lines of the ldif file generated by the **db2ldif** command, containing the 3 suffixes defined in the ldap configuration file.

```
BROWSE -- /usr/local/ldap/ldapload.ldif ----- Line 00000000 Col 0
Command ==>                                     Scroll ==
***** Top of Data *****
version: 1

dn: CN=LOCALHOST
objectclass: container
objectclass: TOP
cn: localhost
aclentry: cn=anybody:NORMAL:RSC:SYSTEM:RSC
aclpropagate: TRUE

dn: o=WASNaming,c=FR
o: WAS_ROOT
o: WASNaming
objectclass: organization
objectclass: TOP
description: WAS Naming over LDAP Name Tree Root
userpassword:: e25vbmV9c2VjcmV0
entryowner: access-id:CN=WASADMIN,O=WASNAMING,C=FR
ownerpropagate: TRUE
aclpropagate: TRUE
aclentry: group:CN=ANYBODY:normal:rsc
aclentry: access-id:CN=WASADMIN,O=WASNAMING,C=FR:normal:rwsc:object:ad

dn: o=BOSS,c=FR
o: CB_ROOT
o: BOSS
objectclass: top
objectclass: organization
description: CBServer Name Tree Root
```

Figure 6-15 Lines of the LDIF file generated by DB2LDIF Command

3. Obtain the DB_VERSION setting for the existing RDBM DB2 tables.
Run the SPUFI request shown in Figure 6-16 on page 200.


```

BROWSE      PRICHAR.LDAP.CNTL(TDBMSPUF) - 01.00          Line 00
***** Top of Data *****
//PRICHARH JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHAR      29 NOV 2001      17:05:03
//*      -LIB- PRICHAR.LDAP.CNTL(TDBMSPUF)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//*****
//LOADTB EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
PROFILE NOPREF
DSN SYSTEM(DB1T)
RUN PROGRAM(DSNTIAD) PLAN(DSNTIA71) -
    LIB('DSNDBOT.RUNLIB.LOAD')
END
//SYSIN DD *
CREATE DATABASE BBOLDAPT STOGROUP BBOLDSTO;
CREATE TABLESPACE ENTRYTS IN BBOLDAPT
    USING STOGROUP BBOLDSTO
    PRIQTY 14400
    SECQTY 7200
    BUFFERPOOL BP0;

.....
.....
.....
insert here the TDBMDB and TDBMINDX members of SGLDSAMP
.....
.....
CREATE UNIQUE INDEX CBLDAP.DIR_CHANGE1
    ON CBLDAP.DIR_CHANGE( ID )
    USING STOGROUP BBOLDSTO
    DEFER YES;

CREATE UNIQUE INDEX CBLDAP.DIR_LONGCHANGE1
    ON CBLDAP.DIR_LONGCHANGE( ID, SEQ )
    USING STOGROUP BBOLDSTO
    DEFER YES;

GRANT EXECUTE ON PLAN DSNACLI TO CBLDAP;

GRANT SELECT ON SYSIBM.SYSCOLUMNS TO CBLDAP;

GRANT DBADM ON DATABASE BBOLDAPT TO CBLDAP;

```

Figure 6-17 Batch version of SPUFI commands

Note: The database name was changed from BBOLDAP to BBOLDAPT in order to keep the old RDBM database. We also changed the prefix of the tables and indexes to match the RACF userid associated with the LDAP server address space, namely CBLDAP.

Also, if you decide to change the database name, as we did, don't forget to rerun part of the BBOCBGRT job from the WAS.CNTL data set, in order to make the necessary grants for the database. (See Figure 6-18)

```

EDIT          PRICHAR.LDAP.CNTL(BBOCBGRT) - 01.02          Columns 000
***** ***** Top of Data *****
000001 //BBOCBGRT JOB (ACCTNO,ROOM),'PRICHAR',CLASS=A,REGION=OM,
000002 // NOTIFY=&SYSUID,MSGCLASS=X
000003 //*
000004 //*
000005 //BBOCBGRT EXEC PGM=IKJEFT01,DYNAMNBR=20
000006 //DBRMLIB DD DISP=SHR,DSN=DSNDBOT.DBRMLIB.DATA
000007 //SYSTSPRT DD SYSOUT=*
000008 //SYSUDUMP DD SYSOUT=*
000009 //SYSPRINT DD SYSOUT=*
000010 //SYSTSIN DD *
000011 DSN SYSTEM(DB1T)
000012 RUN PROGRAM(DSNTIAD) PLAN(DSNTIA71) -
000013 LIB('DSNDBOT.RUNLIB.LOAD')
000014 END
000015 //SYSIN DD *
000016 GRANT DBADM ON DATABASE BBOLDAP TO CBNAMSRI;
000017 GRANT DBADM ON DATABASE BBOLDAP TO CBINTSRI;
000018 /*
000019 //

```

Figure 6-18 Grants for DB2 database

6. Make appropriate configuration changes. See "Configuration File Checklist" in *SecureWay Security Server LDAP Server Administration and Use*, SC24-5923 for information on the TDBM configuration options.

Basically, we need to modify the LDAP server configuration file with the changes shown in Figure 6-19 on page 203.


```

BROWSE -- MVS0.bboslpad.conf                               Line 0000002
Command ==>                                                Scr
#-----
# Standard stuff...
#-----
listen ldap://:1389
maxconnections 300
#sysplexGroupName BBOPORTINSO
#sysplexServerName PORTINSO
adminDN        "cn=CBAAdmin"
adminPW        secret
#-----
#
# DB2 database access definitions
#
# Note: LDAP code doesn't handle the dsnaoini living in the
#       HFS until R10. So we will include a //DSNAOINI DD PATH=
#       statement in the LDAP server, BBONMS and BBOIRS procs.
#-----
database tdbm GLDBTDBM
#database rdbm GLDBRDBM
#multiserver y
#dsnaoini not used check JCL
servername RDBNDBOT
databasesname BBOLDAPT
#databasesname BBOLDAP

dbuserid CBLDAP
pwEncryption none
extendedgroupsearching on
#tbspacemutex BBOMUTX
#tbspaceentry BB0ENT
#tbspace32k BB032K
#tbspace4k BB04K
suffix "cn=localhost"
suffix "o=WASNaming,c=FR"
suffix "o=BOSS,c=FR"

```

Figure 6-19 Job to change LDAP server configuration file

We made the following modifications to bboslpad.conf:

- Commented out all the include statements because we are going to use dynamic schema definitions with the TDBM backend. The following statements were commented out:
 - include /usr/lpp/ldap/etc/slapd.at.system
 - include /usr/lpp/ldap/etc/slapd.cb.at.conf
 - include /usr/lpp/ldap/etc/slapd.at.conf
 - include /usr/lpp/ldap/etc/slapd.oc.system
 - include /usr/lpp/ldap/etc/slapd.cb.oc.conf
 - include /usr/lpp/ldap/etc/slapd.oc.conf
- Changed the database statement to reflect the TDBM (not RDBM).
- Changed the database name (our initial choice, but you may use the same name).
- Change dbuserid to CBLDAP (the userid associated with the LDAP server procedure) instead of BBOLDAP, because that is what is usually recommended in the LDAP

documentation, and also because we had changed the prefix of the tables and indexes to that userid.

- Commented out the following statements (no longer supported with TDBM):

```
#tbspacemutex  BB0MUTX
#tbspaceentry  BB0ENT
#tbspace32k    BB032K
#tbspace4k     BB04K
```

After making the necessary changes to the LDAP server configuration file, we restarted our LDAP server.

As part of the LDAP server log, we received the messages shown in Figure 6-20, indicating that the TDBM backend has been configured.

```
GLD0022I z/OS Version 1 Release 2 Security Server LDAP Server
Starting slept.
GLD0010I Reading configuration file //DD:CONFIG.
GLD3047E OCSF setup failed, but preinduction method 'none' is available.
GLD3062I Password encryption method 'none' is enabled in the TDBM backend.
GLD0163I Backend capability listing follows:
GLD0166I Backend type: ICBM, Backend ID: TDBM BACKEND, Backend suffix:
CN=LOCALH
GLD0165I Capability: LDAP_Backend_ID      Value: TDBM BACKEND
GLD0165I Capability: LDAP_Backend_BldDateTime Value:
2001-11-09-15.01.48.000
GLD0165I Capability: LDAP_Backend_APARLevel Value: LDAP
GLD0165I Capability: LDAP_Backend_Release Value: R 2.0
GLD0165I Capability: LDAP_Backend_Version Value: V 1.0
GLD0165I Capability: LDAP_Backend_Dialect Value: DIALECT 1.0
GLD0165I Capability: LDAP_Backend_BerDecoding Value: BINARY
GLD0165I Capability: LDAP_Backend_ExtGroupSearch Value: YES
GLD0165I Capability: LDAP_Backend_krbIdentityMap Value: YES
GLD0165I Capability: supportedControl Value: 2.16.840.1.113730.3.4.2
GLD0165I Capability: supportedControl Value: 1.3.18.0.2.10.2
GLD0167I End of capability listing for Backend type: tdbm, Backend ID: TDBM
BACK
```

Figure 6-20 LDAP Server log showing that a TDBM backend has been configured

7. Load the schema associated with the TDBM backend, which conforms to the LDAP Version 3 (V3) protocol. New schema that are consistent with earlier schema files have been shipped with z/OS in the V3 format.

Beginning with OS/390 Release 8, a greatly enhanced set of LDAP schema is shipped along with the LDAP server. The new schema files are installed into the installation directories of the z/OS LDAP server installation for customer use. The new schema files are in the /usr/lpp/ldap/etc directory and are named schema.*.

Note that the new schema files represent a superset of what has been developed and shipped with releases of the LDAP server on OS/390 prior to OS/390 Release 8. Thus, once the new files are used, the old attribute type and object class definition configuration files must not be included in the configuration.

Also, the LDAP server allows the schema to be changed dynamically through the LDAP protocol when using a TDBM backend. This will reduce server outages.

The minimum schema contains the minimum schema attributes necessary to start the z/OS LDAP server and create a schema. These minimum definitions are shipped internal to the z/OS LDAP server and may be viewed by performing a search of the schema anytime after the server is started. If the LDAP search is performed immediately after the server is started at Release 10 level, the minimum schema will be the complete schema entry shown. When additional schema elements are loaded, the minimum schema elements will appear as part of the entire schema entry.

With z/OS LDAP, the schema definitions which the server uses to manage the LDAP directory are loaded using the LDAP modify function. There are two types of schema files shipped with the z/OS LDAP server that may be used to load the directory schema.

The schema.user.ldif and schema.IBM.ldif files are the LDAP protocol V3 dynamic schema versions of the schema.system.at, schema.system.oc, schema.user.at, schema.user.oc, schema.IBM.at, and schema.IBM.oc files that were used in previous releases of the z/OS LDAP server. The new versions of these files have changed in the following ways:

- ▶ They are in LDIF format rather than configuration file format.
- ▶ The files are defined in terms of the LDAP protocol Version 3 (IETF RFC 2252 format).
- ▶ They use of new syntaxes and separate specification of matching rules.
- ▶ The syntaxes supported in previous OS/390 LDAP servers (bin, ces, cis, dn, tel) have been converted to a combination of a syntax and matching rule. These changes are consistent with the LDAP protocol Version 3 definitions of the attribute types.
- ▶ New attribute types and object classes have been added.
- ▶ Superior attribute types and object classes are used in the definitions.

The schema.IBM.ldif file requires the schema.user.ldif file to be loaded first.

The second type of schema files that are shipped with the z/OS LDAP server are collections of schema attributes that relate to specific uses. These schema files may be loaded by the system administrator to satisfy particular schema needs. The schema files are provided so that the system administrator can load a smaller set of schema, if desired. The schema definitions contained in these files are subsets of the schema definitions found in schema.user.ldif or schema.IBM.ldif.

The schema files that are shipped with the z/OS LDAP server are based on industry- and product-defined schemas. As such, they should not be modified since existing products and applications use the schema elements as defined.

The schema files identified in Table 33 of z/OS Security Server *LDAP Server Administration and Use*, SC24-5923-02, are shipped with the z/OS LDAP server and are located in the /usr/lpp/ldap/etc directory. The “Included in” column shows which of the schema.user.ldif or schema.IBM.ldif files also include the attribute type and object class definitions contained in the specific file.

6.3.2 Setting up the schema for TDBM

The LDAP Server is shipped with a set of predefined schema files representing schemas which the user may want to load as the LDAP schema.

Schema.IBM.ldif, shown in Figure 6-21 on page 206, contains the schema definitions for WebSphere naming. Schema.user.ldif, shown in Figure 6-22, contains the set of schema definitions for ComponentBroker.

```

BROWSE -- /usr/lpp/ldap/etc/schema.IBM.ldif ----- Line 0000001
Command ==> Scr

# -----
# This is the z/OS LDAP Server general IBM-defined schema
# definition file. Do not alter the definitions in this file.
# -----
# This file is a compilation of the following schema ldif files:
# CommServer.ldif
# DB2.ldif
# DMTF.ldif
# IBM.ldif
# Kerberos-V1.ldif
# ManagedSystemInfrastructure.ldif
# MCI.ldif
# MetaDirectory.ldif
# NativeAuthentication.ldif
# NFI.ldif
# OnDemandServer.ldif
# RACF.ldif
# RACF.2.ldif
# RegisteredSoftware.ldif
# UNIX.ldif
# WebSphereNaming.ldif
# -----

```

Figure 6-21 *Schema.IBM.LDIF contains schema definitions for WebSphere naming*

```

BROWSE -- /usr/lpp/ldap/etc/schema.user.ldif ----- Line 00000011 Col
Command ==> Scroll =

# -----
# This is the z/OS LDAP Server general externally-defined schema
# definition file. Do not alter the definitions in this file.
# -----
# This file is a compilation of the following schema ldif files:
# ChangeLog.ldif
# ComponentBroker.ldif
# EntrustPKIV4.ldif
# EntrustPKIV5.ldif
# MS.ActiveDirectory.ldif
# Netscape.ldif
# Netscape-V2.ldif
# nisSchema.ldif
# OtherStandard.ldif
# RFC2252.ldif
# RFC2256.ldif
# RFC2587.ldif
# RFC2713.ldif
# RFC2714.ldif
# SecurityIdentities.ldif
# System.ldif
# System-V2.ldif
# UniversalMessaging.ldif
# X.520.ldif

```

Figure 6-22 *Schema.user.ldif contains schema definitions for component broker*

You should use both schema files to represent the data stored in the TDBM database. Copy the files from the /usr/lpp/ldap/etc directory to a working directory, for instance /usr/local/ldap.

For each file, find the line dn:cn=schema,<suffix> and replace <suffix> with one of the suffixes defined for TDBM in the configuration file. Remember that WebSphere Application Server provides 3 different suffix entries.

You can establish your schema DN under any suffix managed by TDBM.

It doesn't really matter which suffix you use. You could use your own installation-specific one or you could use "cn=localhost"; both will work just fine. From a documentation perspective, it's probably easier to use "cn=localhost" as the root entry to define the schema entry.

However, it's probably clearer to describe the suffix using the WebSphere Application Server-preferred name (i.e: suffix "o=WASNaming,c=XX").

There are places in the WebSphere Application Server documentation where it is indicated that "cn=localhost" is no longer a required suffix, so you should not be confused with conflicting documentation.

In our installation we decided to use "cn=localhost."

To know under which suffix to define the schema definition, you can run the LDAP command shown in Figure 6-23.

```
BROWSE    PRICHAR.LDAP.CNTL(CONTEXT) - 01.00          Line 0000000
***** Top of Data *****
//PRICHAR6 JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHAR      29 Nov 2001      17:52:36
//*      -LIB- PRICHAR.LDAP.CNTL(SUBSCHEM)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//STEP011 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
/* find context          */
osshell +
/bin/ldapsearch -h 9.100.203.110 -p 1389 +
-D "cn=CBAAdmin" -w secret +
-s base -V 3 "objectclass=*"
//
```

Figure 6-23 LDAP command to determine the suffix to define schema definition

When we did this, we got the output shown in Figure 6-24 on page 208.

```

supportedcontrol=2.16.840.1.113730.3.4.2
supportedcontrol=1.3.18.0.2.10.2
namingcontexts=cn=localhost
namingcontexts=o=WASNaming,c=FR
namingcontexts=o=BOSS,c=FR
namingcontexts=sysplex=MVS0
subschemasubentry=CN=SCHEMA,cn=localhost
supportedsaslmmechanisms=EXTERNAL
supportedldapversion=2
supportedldapversion=3
ibmdirectoryversion=z/OS V1R2

```

Figure 6-24 Output from LDAP command

Therefore, we thought it would be advisable to use the subschemasubentry as reported by the LDAP database search.

For both schema.user.ldif and schema.IBM.ldif, find the line dn:cn=schema,<suffix> and replace <suffix>. Figure 6-25 is an example for schema.IBM.ldif.

```

EDIT      /usr/local/ldap/schema.IBM.ldif      Columns 00001
00072
Command ===>      Scroll ===>
CSR
000032 # WebSphereNaming.ldif
000033 #
-----
000034 dn: cn=schema,cn=localhost
000035 changetype:modify
000036 add:attributetypes
000037 attributetypes: (
000038   1.3.18.0.2.4.554
000039   NAME 'acceleratorCapabilities'
000040   DESC 'Based on CIM. Indicates the graphics and 3D capabilities of
t
000041   SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
000042   USAGE userApplications
000043   )

```

Figure 6-25 Example for schema.IBM.ldif

Then, run the **ldapmodify** command from the z/OS shell, specifying the host, port, bind DN, password, and schema file. This will load the schema into the directory. Remember that the schema.IBM.ldif file requires the schema.user.ldif file to be loaded first.

You can use, as an example, the 2-step job shown in Figure 6-26 on page 209. After running these jobs, check the subschema entry to make sure it was correctly updated.

```

BROWSE    PRICHAR.LDAP.CNTL(SCHEMLDI) - 01.05          Line
/**
//STEP010 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM   DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN  DD *
/* finalize set-up of ldap server*/
  oshell +
/bin/ldapmodify -h 9.100.203.110 -p 1389 +
-D "cn=CBAdmin" -w secret +
-f /usr/local/ldap/schema.user.ldif
//STEP011 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM   DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN  DD *
/* finalize set-up of ldap server*/
  oshell +
/bin/ldapmodify -h 9.100.203.110 -p 1389 +
-D "cn=CBAdmin" -w secret +
-f /usr/local/ldap/schema.IBM.ldif

```

Figure 6-26 Job to run Ldapmodify and bind dn password and schema file

Displaying the schema entry

The following command shows how to search for the schema entry:

```
ldapsearch -h hostname -b "cn=schema,<naming context >" -s base "objectclass=subschema"
```

The job shown in Figure 6-27 on page 210 can help you list the content of the schema entry.

```

BROWSE    PRICHAR.LDAP.CNTL(SCHMLST) - 01.02          Line 00000000
***** Top of Data *****
//PRICHARB JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHAR      29 Nov 2001      18:30:04
//*      -LIB- PRICHAR.LDAP.CNTL(SCHMLST)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//STEP011 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
/* list schema entries */
  oshell +
/bin/ldapsearch -h 9.100.203.110 -p 1389 +
-D "cn=CBAdmin" -w secret +
-s base +
-b "cn=schema,cn=localhost" "objectclass=subschema"
//

```

Figure 6-27 Search for schema entry

Loading the data

8. The last step of our migration process involves loading the data from the LDIF file created way back in step 2, described on page 196, using either the **ldif2tdbm** utility or the **ldapadd** utility.

We used the **ldapadd** command to perform the load function to repopulate the TDBM DB2 tables.

The JCL we used to perform this in batch mode is shown in Figure 6-28 on page 211.


```

BROWSE    PRICHAR.LDAP.CNTL(LDIF2TDB) - 01.02          Line 00000000 Co
***** Top of Data *****
//PRICHARG JOB (PRICHAR),'PRICHAR',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHAR,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHAR      29 Nov 2001      18:40:25
//*      -LIB- PRICHAR.LDAP.CNTL(LDIF2TDB)
//*
//*      -PURPOSE - _____
//*                  reload the bboldap rdbm ldif file to ____
//*                  tdbm backend_____
//*
//*
//*****
//* This job:
//* 1. Runs OSHELL under the TSO TMP to issue UNIX cmds from batch.
//*****
//S1      EXEC PGM=IKJEFT1B
//SYSTSPRT DD SYSOUT=*
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTEM  DD DUMMY
//SYSTSIN DD *
          oshell +
          /bin/ldapadd -h 9.100.203.110 -p 1389 +
          -D "cn=CBAAdmin" -w secret +
          -f /usr/local/ldap/ldapload.ldif

```

Figure 6-28 JCL to repopulate TDBM DB2 tables

6.3.3 Restarting WebSphere Application Server after TDBM migration

After going through the migration steps, we restarted all our WebSphere Application Server servers successfully and were able to run the IVPs.

However, we encountered a problem trying to start the SMEUI administration tool. The logs that included the problem messages are shown in Figure 6-29 on page 212 and Figure 6-30 on page 213.

```

+BB0U0243E IMPLEMENTATION OF CLASS NamingServiceDO::NamingContextDO 748
  EITHER CAUSED OR RAISED EXCEPTION CORBA::PERSIST_STORE
  DURING INVOCATION OF METHOD retrieveFromDataStore.

From the BBONMS SYSOUT trace (more than 220K lines of output!) , i found the
following exceptions:

): HashableValue::hash, = sign missing from HashableValue key
==>LDAPException::throwLdapException,rc=0x22,rsncode=2001,logstatus=NOTLOGGE
D,lineno=114,from >@(#)09 1.3
src/servers//slapd/libutils/hashable_value.c,ldap.slapp, os390r12ldap,
11276.0W51987

<==LDAPException::throwLdapException, throwing LDAPException
<==> LDAPException::getLogStatus(), returning NOTLOGGED
<==> LDAPException::getReasonCode(), returning 2001
<==> LDAPException::getReturnCode(), returning 0x22
<==> LDAPException(const LDAPException& le)
<==> ~LDAPException
<==>HashableValue::~HashableValue key=CBADMIN, data =NULLSTRBUF,

<=! SchemaImpl::dnNormalize, caught LDAPException, rethrowing
<==>LDAPException::rethrow,lineno=2561,from >@(#)91 1.75
src/servers/slapd/libutils/schemaimpl.c, ldap.slapp,
os390r12ldap, 11276.0W51987 6/28/01 07:24:43<

<==> LDAPException::getReasonCode(), returning 2001
<==> LDAPException::getReturnCode(), returning 0x22
<==> LDAPException(const LDAPException& le)
<==> LDAPException::getReturnCode(), returning 0x22
==>LDAPException::getMsg()
<==LDAPException::getMsg(), returning >R002001 Missing equal sign in DN
(schemaimpl.c|1.75|2561)<
component: CBADMIN (hashable_value.c|1.3|114), (schemaimpl.c|1.75|2561)<
<=! getPrincipalDN: unable to create PrincipalDN, rc = 34, errorText =
R002001 Missing equal sign in DN component: CBADMIN
(hashable_value.c|1.3|114),
:rethrow,lineno=134,from >@(#)15 1.17
.....
.....
: <==> LDAPException::getLogStatus(), returning NOTLOGGED
: <==> LDAPException::getReasonCode(), returning 2001
: <==> LDAPException::getReturnCode(), returning 0x22
: <==> LDAPException(const LDAPException& le)
: <==> LDAPException::getReturnCode(), returning 0x22
: Unable to check ACLs or send entry, continuing ...
: <==>LDAPException::rethrow,lineno=1134,from >@(#)79 1.74.2.2
p.tdbm, os390r12ldap, 11276.0W51987 8/16/01 16:02:22<

```

Figure 6-29 Log after restart of WebSphere Application Server after TDBM migration

```

<==> LDAPException::getLogStatus(), returning NOTLOGGED
<==> LDAPException::getReasonCode(), returning 2001
<==> LDAPException::getReturnCode(), returning 0x22
<==> LDAPException(const LDAPException& le)
<==> LDAPException::getReturnCode(), returning 0x22
<==> LDAPException::getReasonCode(), returning 2001
level_search: terminal LDAPException catch; rc=0x22, rsnocode=2001
<==> LDAPException::getReturnCode(), returning 0x22
==>LDAPException::getMsg()
<==LDAPException::getMsg(), returning >R002001 Missing equal sign in DN
component: CBADMIN (hashable_value.c|1.3|114), (schemaimpl.c|1.75|2561),
(aclsuobject.c|1.17|134), (tdbm_search.c|1.74.2. 2|1134)<

<==> LDAPException::getReturnCode(), returning 0x22
<==> ~LDAPException
<==> ~LDAPException
<==> ~LDAPException
<==> ~LDAPException
-- ~ACLSubject destructor called
-- ~ACLObject destructor called
-- ~ACLRights destructor called
=> FreeTDBMRequest, old = 0x17998f48.
==> freeCachesForTDBMRequest()
==> TDBMCacheManager::releaseCaches()
<== TDBMCacheManager::releaseCaches()
<== freeCachesForTDBMRequest()
7664857      20 usec SQLFreeStmt(2,SQL_DROP) => 0
<= FreeTDBMRequest.
<= level_search, rc = 34
local_send_ldap_result 34:NULLSTRBUF:R002001 Missing equal sign in DN
(schemaimpl.c|1.75|2561), (aclsuobject.c|1.17|134), (tdbm_search.c|1.74.2.

```

Figure 6-30 ACL problem log

Those messages seemed to point to ACL-type problems for the CBADMIN.

At this point we were quite at a loss, until we went to the IBM intranet support Web site (<http://www-1.ibm.com/support/techdocs/atsmastr.nsf/>). We found useful information in an IBM white paper, “Migrating LDAP RDBM to TDBM” document number WP100223, revised 06/11/2001. The abstract is as follows:

“This document describes how to migrate from LDAP using RDBM to TDBM for WebSphere Application Server V4.0 for z/OS and OS/390.

This document written by the Washington System Center deals with RDBM to TDBM step-by-step migration recommendations. Every customer should be advised to request a copy of this flash and review it carefully before starting the migration process.”

Essentially, the paper summarizes the migration steps we have described, but it also advises to update the ACLs as follows:

```

aclentry: group:CN=ANYBODY:normal:rsc
aclentry: access-id:racfid=CBYSMSR1,profiletype=user,o=WSLPLEX:normal:rw
sc:sensitive:rwsc:critical:rwsc:object:ad
aclentry: access-id:racfid=CBADMIN,profiletype=user,o=WSLPLEX:normal:rw
sc:sensitive:rwsc:critical:rwsc:object:ad
aclentry: access-id:CN=BOSSADMIN,O=CB390_WSLPLEX:normal:rwsc:object:ad
aclentry: group:CN=ANYBODY:normal:rsc
aclentry: access-id:racfid=CBYSMSR1,profiletype=user,o=WSLPLEX:normal:rw
sc:sensitive:rwsc:critical:rwsc:object:ad

```

```

aclentry: access-id:racfid=CBADMIN:,profiletype=user,o=WSLPLEX:normal:rw
sc:sensitive=rwsc:critical:rwc:object:ad
aclentry: access-id:CN=WASADMIN,O=WAS390_WSLPLEX:normal:rwc:object:ad

```

This was the piece of information we were missing. So, we run the **ldapmodify** job shown in Figure 6-31.

```

BROWSE    PRICHAR.LDAP.CNTL(ACLUPDT) - 01.02          Line 000
***** Top of Data *****
//PRICHA1D JOB (PRICHA1),'PRICHA1',CLASS=A,MSGCLASS=X,
//          NOTIFY=PRICHA1,REGION=OM,MSGLEVEL=1
//*
//*
//*      -DOC- PRICHA1      5 Dec 2001      15:57:11
//*      -LIB- PRICHAR.LDAP.CNTL(ACLUPDT)
//*
//*      -PURPOSE - _____
//*      _____
//*      _____
//*
//STEP011 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
/* list schema entries */
  oshell +
/bin/ldapmodify -h 9.100.203.110 -p 1389 +
-D "cn=CBAdmin" -w secret +
-V 3 -f /usr/local/ldap/ldapac1.ldif
//

```

Figure 6-31 LDAPMODIFY Job

The input file `/usr/local/ldap/ldapac1.ldif` contains the ACL input shown in Example 6-6.

Example 6-6

```

dn: o=WASNaming,c=FR
changetype: modify
replace: x
aclentry: group:CN=ANYBODY:normal:rsc
aclentry:
access-id:racfid=CBADMIN,profiletype=user,sysplex=MVS0:normal:rwc:sensitive=rwc:critical
=rwc:object:ad
aclentry:
access-id:racfid=CBADMIN,profiletype=user,sysplex=MVS0:normal:rwc:sensitive=rwc:critical=
rwc:object:ad
aclentry: access-id:CN=WASADMIN,O=WASNAMING,C=FR:normal:rwc:object:ad

dn: o=BOSS,c=FR
changetype: modify
replace: x
aclentry: group:CN=ANYBODY:normal:rsc
aclentry:
access-id:racfid=CBADMIN,profiletype=user,sysplex=MVS0:normal:rwc:sensitive=rwc:critical
=rwc:object:ad

```

```
aclentry:  
access-id:racfid=CBADMIN,profiletype=user,sysplex=MVS0:normal:rwsc:sensitive=rwsc:critical=  
rwsc:object:adn  
aclentry: access-id:CN=CBADMIN,O=BOSS,C=FR:normal:rwsc:object:ad
```

After that, we were able to start all the WebSphere Application Server servers, as well as the SMEUI administration user tool, without any problem.

Notes about this flash: This WSC white paper contains a few mistakes, which at the time of writing this redbook had not been corrected yet.

- ▶ The SLAPD conf file contains a RACF database backend definition:
 - This is not required; the requirement is for the correct RDN-style names for the RACF userids that need access. In fact, adding SDBM to the mix causes a number of problems with program-controlled libraries. So you should not configure the LDAP server with the GLDSDBM statement in the slapd.conf file.
- ▶ In the original RDBM backend there was no ACL entry for CBSYMSR1 and CBADMIN for dn: o=WASNaming,c=XX, only an ACL entry for access-id:CN=WASADMIN,O=WASNAMING,C=XX:normal:rwsc:object:ad, but the sample ACL file in the flash shows 2 ACL entries for those 2 RDNs. (See the ACL input file discussed previously.)
 - There is no reason to have either of these entries in the ACL list since these RDNs are not used to update the J2EE part of the namespace. Again, it doesn't hurt anything, it is just not needed.
 - Also, there is typo in the flash, it should read CBSYMCR1, not CBSYMSR1. This is a typo, and it does not effect anyone converting from RDBM to SDBM since this userid is not used to update the namespace from BBONMS or BBOIRS after the bootstrap is complete.

6.3.4 Tips and recommendations for debugging/tracing

Most of the problems you are likely to face are related to LDAP security-type problems.

Our experience on WebSphere Application Server migration has shown that the following traces/logs are very useful in trying to understand and debug problems.

JAVA TR - tracing in Systems Management

The Java tracing is managed by the settings in the jvm.properties file, of which there is one per server, and it is in the same directory as the current.env for that server. The jvm.properties file has one setting that points to where the actual trace settings are.

1. Create a jvm.properties file (if you don't already have one) in the same directory as the current.env for System Management. The jvm.properties file has one setting that points to where actual trace settings are. The following is a sample entry in the jvm.properties file that you can use as a model:

```
com.ibm.ws390.trace.settings=/WebSphere390/CB390/controlling/Neville/  
PLEX1/server/trace.dat
```

where *server* in this case would be System management.

2. Create a trace.dat file in the directory you identified in jvm.properties. In this trace.dat file put this line:

```
com.ibm.*=all=enabled
```

3. In the current.env for System Management, change your TRACEBUFFLOC from BUFFER to SYSPRINT.
4. Cancel BBOSMSS. This will cause a new BBOSMSS to be started and this one will “pick up” the tracing choices you just created in the jvm.properties and trace.dat file.

LDAP tracing: A new way to enable and disable LDAP tracing

1. From the operator console, enter this command:

```
fldapsrv,appl=debug=65535
```

(You can also do this from ISPF, but be sure to precede the command with a /.)

2. In the current.env for your appl.server change your TRACEBUFFLOC from BUFFER to SYSPRINT.
3. Cancel you appl.server and then start it again. This will cause it to again try to register the new IVP data to LDAP, and this time we'll be generating trace data to SYSPRINT.

Therefore, to activate the LDAP trace for the NAMING server, you should update the current.env file of the server and add the following parameter:

```
LDAP_DEBUG=1376255
```



```

BROWSE -- current.env
Command ==>
LDAPROOT=o=BOSS,c=FR
LDAP_DEBUG=65535
LIBPATH=/usr/lpp/db2/db2710/lib:/usr/lpp/java_jdk130/IBM/J1.3/bin:/usr/lp
LOGSTREAMNAME=WAS.ERROR.LOG
NM_GENERIC_SERVER_NAME=CBNAMING
NM_SPECIFIC_SERVER_NAME=NAMING01
NMPROC=BBONM
OTS_DEFAULT_TIMEOUT=300
OTS_MAXIMUM_TIMEOUT=300
RAS_MINORCODEDEFAULT=NODIAGNOSTICDATA
RESOLVE_IPNAME=wasv4.secrs.pssc
RESOLVE_PORT=900
SM_DEFAULT_ADMIN=CBADMIN
SM_GENERIC_SERVER_NAME=CBsysmgmt
SM_SPECIFIC_SERVER_NAME=SYSMGTO1
SMPROC=BBOSMS
SOMOOSQL=1
SRVIPADDR=None
SYS_DB2_SUB_SYSTEM_NAME=DB1T
TRACEALL=1
TRACEBUFFLOC=SYSPRINT
TRACEPARM=00
DB2SQLJPROPERTIES=/usr/lpp/db2/db2710/classes/db2sqljjdbc.properties
  
```

Figure 6-32 LDAP debug

Application servers tracing

Add the following line to the current.env of the application server that is trying to register:

```
JVM_DEBUG=1
```

This is Java trace. Typically the problem is a class not found. You can try to find that in the output produced (SYSOUT of job output) from your application server region.

You should also modify the current.env for your server regions with the following:

```
TRACEALL=3
```

```
TRACEDETAIL=(3,4)  
TRACEBUFF=SYSPRINT
```

The traces are written to the sysprint of the corresponding address spaces.

Archived

Archived

LDAP server on z/OS

This chapter describes the steps necessary to configure and start an LDAP server with TDBM and SDBM, using the LDAPCNF utility. It also documents the steps necessary to implement password encryption for the TDBM backend on the z/OS LDAP server.

7.1 Input file description

There are 4 read-only input files in the /usr/lpp/ldap/etc directory, which must be copied to a directory where you can customize them:

ldap.db2.profile
ldap.profile
ldap.racf.profile
ldap.slapped.profile

This can be done by the OMVS command:

```
cp /usr/lpp/ldap/etc/ldap.* /usr/local/alains/ldap/
```

Each of these files contains all parameters with comments and default values. If you need additional details, see *SecureWay Security Server: LDAP Server Administration and Use*, SC24-5923.

7.1.1 Customizing ldap.profile

This is the main configuration file. Changes have been highlighted in Figure 7-1 on page 221, which contains only parameters; all comments have been removed. Below is a list of the changes we did to the ldap.profile:

- ▶ **OUTPUT_DATASET='GLD.CNFOUT5'** specifies the output pds for LDAPCNF utility. It will contain 10 members. It must be allocated and empty before starting the LDAPCNF batch job.
- ▶ Other parameters to check are h1q and volser (either SMS or SYSRES) for system libraries.
- ▶ **DSN_SSID='DB1T'** DB2 subsystem name obtained through D SSI command.
- ▶ **TDBM_SUFFIX='o=IBMMOP'**
- ▶ **SDBM_SUFFIX='sysplex=portinfo'**
- ▶ **ADMINDN='cn=LDAP Administrator,o=IBMMOP'**
 - LDAP Administrator and password are described in TDBM backend.
 - Since there is an SDBM backend, we could also define the administrator as a RACF userid: **ADMINDN="racfid=ALAINS,profiletype=user,sysplex=portinfo"**
- ▶ Jobcards skeletons for JCL build by LDAPCNF.
- ▶ Localization of ldap.slapped.profile, ldap.racf.profile, and ldap.db2.profile.

```

USR_LPP_ROOT='/'
OUTPUT_DATASET='GLD.CNFOUT5'
OUTPUT_DATASET_VOLUME='SMS'
TEMPORARY_DIR='/tmp'
GLDHLQ='SYS1'
SGLDLNKVOL='SYSRES'
GSKHLQ='GSK'
SGSKLOADVOL='SMS'
DSN_SDSNEXITHLQ='DSN710'
SDSNEXITVOL='SMS'
DSN_SDSNLOADHLQ='DSN710'
SDSNLOADVOL='SMS'
DSN_SDSNDBRMHLQ='DSN710'
DSN_SSID='DB1T'
CEEHLQ='SYS1'
SCEERUNVOL='SYSRES'
CBCHLQ='CBC'
CLBDLLVOL='SYSRES'
CSSLIBVOL='SYSRES'
LINKLIBVOL='SYSRES'
LDAPUSRID='GLDSRV'
LDAPUSRGRP='GLDGRP'
TDBM_SUFFIX='o=IBMMOP'
SDBM_SUFFIX='sysplex=portinfo'
ADMINDN='cn=LDAP Administrator,o=IBMMOP'
PROG_SUFFIX='00'
APF_JOB CARD_1="//JOBAPF1 JOB (),MSGLEVEL=(1,1),MSGCLASS=X"
PRGCTRL_JOB CARD_1="//JOBAPRG1 JOB (),MSGLEVEL=(1,1),MSGCLASS=X"
DB2_JOB CARD_1="//JOBDB21 JOB (),MSGLEVEL=(1,1),MSGCLASS=X"
RACF_JOB CARD_1="//JOB RAC1 JOB (),MSGLEVEL=(1,1),MSGCLASS=X"
${SOURCE_CMD} ${USR_LPP_ROOT}usr/local/alains/ldap/ldap.slapd.profile
${SOURCE_CMD} ${USR_LPP_ROOT}usr/local/alains/ldap/ldap.db2.profile
${SOURCE_CMD} ${USR_LPP_ROOT}usr/local/alains/ldap/ldap.racf.profile

```

Figure 7-1 Customized ldap.profile

7.1.2 Customizing ldap.db2.profile

We have modified only 2 parameters in this file:

- **TDBM_DB2_LOCATION='RDBNDBOT'**

This is the DB2 server location which can be obtained by running a DSNJU004 utility

```

//ALAINS JOB LDAP,LDAP,CLASS=A,
// MSGLEVEL=(1,1),MSGCLASS=X,
// NOTIFY=&SYSUID
//STEP010 EXEC PGM=DSNJU004
//SYSUT1 DD DISP=SHR,DSN=DSNDBOT.DB1T.BSDS01
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
/*

```

Figure 7-2 DSNJU004 utility JCL

- **TDBM_DB2_DBNAME='GLddb3'**

```

TDBM_DB2_PLAN='DSNACLI'
TDBM_DB2_LOCATION='RDBNDBOT'
TDBM_DB2_USERID='GLDSRV'
TDBM_DB2_DBNAME='GLDDB3'
TDBM_DB2_STORAGEGROUP='SYSDEFLT'
TDBM_DB2_ENTRYTABLESPACE='ENTRYTS'
TDBM_DB2_ENTRYBUFFERPOOL='BP0'
TDBM_DB2_LONGENTRYTABLESPACE='LENTRYTS'
TDBM_DB2_LONGENTRYBUFFERPOOL='BP0'
TDBM_DB2_LONGATTRTABLESPACE='LATTRTS'
TDBM_DB2_LONGATTRBUFFERPOOL='BP0'
TDBM_DB2_SEARCHTABLESPACE='SEARCHTS'
TDBM_DB2_SEARCHBUFFERPOOL='BP0'
TDBM_DB2_MISCTABLESPACE='MISCTS'
TDBM_DB2_MISCBUFFERPOOL='BP0'
TDBM_DB2_DESCTABLESPACE='DESCTS'
TDBM_DB2_DESCBUFFERPOOL='BP0'
TDBM_DB2_REPLICATABLESPACE='REPTS'
TDBM_DB2_REPLICABUFFERPOOL='BP0'
TDBM_DB2_SEARCHTRUNCsize='32'
TDBM_DB2_DNTRUNCsize='32'
TDBM_DB2_DNMAXsize='512'
TDBM_DB2_ENTRYTS_PRIQTY='14400'
TDBM_DB2_ENTRYTS_SECQTY='7200'
TDBM_DB2_LONGENTRYTS_PRIQTY='14400'
TDBM_DB2_LONGENTRYTS_SECQTY='7200'
TDBM_DB2_LONGATTRTS_PRIQTY='14400'
TDBM_DB2_LONGATTRTS_SECQTY='7200'
TDBM_DB2_SEARCHTS_PRIQTY='14400'
TDBM_DB2_SEARCHTS_SECQTY='7200'
TDBM_DB2_MISCTS_PRIQTY='14400'
TDBM_DB2_MISCTS_SECQTY='7200'
TDBM_DB2_DESCTS_PRIQTY='14400'
TDBM_DB2_DESCTS_SECQTY='7200'
TDBM_DB2_REPLICATS_PRIQTY='14400'
TDBM_DB2_REPLICATS_SECQTY='7200'
TDBM_DB2_ENTRYX2_PRIQTY='14400'
TDBM_DB2_ENTRYX2_SECQTY='7200'
TDBM_DB2_ENTRYX3_PRIQTY='14400'
TDBM_DB2_ENTRYX3_SECQTY='7200'
TDBM_DB2_DESCX1_PRIQTY='14400'
TDBM_DB2_DESCX1_SECQTY='7200'
TDBM_DB2_SEARCHX1_PRIQTY='14400'
TDBM_DB2_SEARCHX1_SECQTY='7200'
TDBM_DB2_SEARCHX2_PRIQTY='14400'
TDBM_DB2_SEARCHX2_SECQTY='7200'
DSN_ATTACH_TYPE='CAF'

```

Figure 7-3 Customized ldap.db2.profile

7.1.3 Customizing ldap.racf.profile

This file contains only two parameters used by LDAPCNF to create a RACF userid (GLDSRV) for the LDAP Started task. Default values have been accepted.

```
LDAPGID='2'  
LDAPUID='1'
```

Figure 7-4 Customized ldap.racf.profile

7.1.4 Customizing ldap.slapd.profile

Following are the options that require customization in the ldap.slapd.profile dataset when used by the LDAPCNF utility.

- ▶ `PORT='3389'` is the non-secure port used by LDAP.
- ▶ `PORT='3390'` is the secure port.
- ▶ `SSL_AUTH='serverAuth'` server authentication only.
- ▶ `SSL_CERTIFICATE='gld ldap server'`
`SSL_CIPHERSPECS='15104'`
`SSL_KEYRINGFILE='/usr/local/alains/ldap/gldclient'`
`SSL_KEYRINGPWSTASHFILE='/usr/local/alains/ldap/gldclient.sth'`
information about the server certificate created by gskkyman
- ▶ `TDBM_USENATIVEAUTH='selected'`
`TDBM_NATIVEUPDATEALLOWED='on'`
`TDBM_NATIVEAUTHSUBTREE='o=IBMMOP'`

Native Authentication can be used for entries in TDBM subtree o=IBMMOP. Password update is allowed.

```
LDAP_LANG='En_US.IBM-1047'  
LDAP_HOSTNAME=''   
PORT='3389'  
MAXCONNECTIONS='60'  
SECURITY='SSL'  
SECUREPORT='3390'  
SSL_AUTH='serverAuth'  
SSL_CERTIFICATE='gld ldap server'  
SSL_CIPHERSPECS='15104'  
SSL_KEYRINGFILE='/usr/local/alains/ldap/gldclient'  
SSL_KEYRINGPWSTASHFILE='/usr/local/alains/ldap/gldclient.sth'  
TDBM_ATTROVERFLOWSIZE='255'  
TDBM_PWENCRYPTION=''   
TDBM_EXTENDEDGROUPSEARCHING='on'  
TDBM_USENATIVEAUTH='selected'  
TDBM_NATIVEUPDATEALLOWED='on'  
TDBM_NATIVEAUTHSUBTREE='o=IBMMOP'
```

Figure 7-5 Customized ldap.slapd.profile

7.2 Starting the ldapcnf utility

This utility can be started from the OMVS shell by the command:

```
/usr/lpp/ldap/sbin/ldapcnf -i profile_file
```

where **profile_file** specifies the input file that contains statements necessary to configure the LDAP server.

This utility can also be started as a batch job since there is no interactivity if the output dataset is allocated and empty. This method allows for easier checking of the process.

```
//ALAINS2 JOB LDAP,LDAP,CLASS=A,
//          MSGLEVEL=(1,1),MSGCLASS=X,
//          NOTIFY=&SYSUID
//STEP010 EXEC PGM=IKJEFT1B
//SYSEXEC DD DISP=SHR,DSN=SYS1.SBPXEXEC
//SYSTSPRT DD SYSOUT=*
//SYSTEM DD DUMMY
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
/* EXECUTE LDAPCNF */
   oshell +
   cd /usr/lpp/ldap/sbin +
   && +
   ldapcnf -i /usr/local/alains/ldap/ldap.profile
/*
```

Figure 7-6 Starting LDAPCNF as a batch job

After successful execution of LDAPCNF JOB, 10 members are written in the file specified in the output_dataset parameter of ldap.profile.

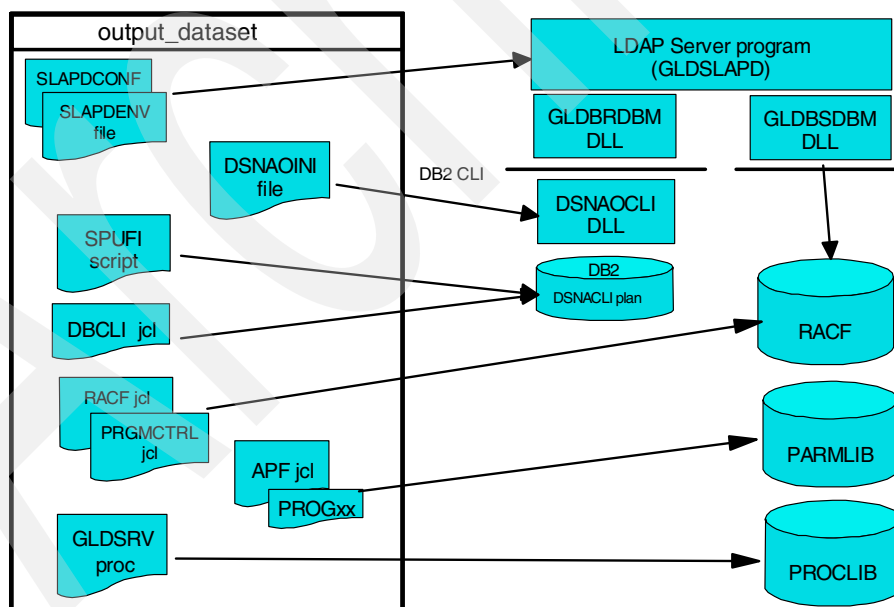


Figure 7-7 Members produced by LDAPCNF in output_dataset

7.3 Executing system and RACF jobs

Following are activities that must be performed for executing system and RACF jobs.

7.3.1 Update proclib

- ▶ Member GLDSRV must be copied into your PROCLIB.
- ▶ Ensure that gldhlq.SGLDLNK is in LINKLIST, otherwise you must manually update the GLDSRV procedure with a STEPLIB locating this library.

7.3.2 Update PARMLIB

- ▶ The PROGxx member must be copied into your PARMLIB, or added to an existing PROGyy member.
- ▶ Run job APF to activate these changes. To make the APF changes permanent, the PROG member must be added to the APF list created at IPL time.

7.3.3 Execute RACF jobs

These jobs must be run by a security administrator.

- ▶ A RACF job creates all necessary definitions in RACF for the LDAP server:
 - Add user, group, and started task definition.
 - Give the necessary authorizations to this started task in class dataset and facility:
 - BPX.DEAMON and BPX.SERVER
 - IRR.RUSERMAP (used if SDBM backend)
- ▶ PRGMCTRL defines, in PROGRAM class, all necessary profiles to protect libraries containing programs which must be protected. This job must be run only if password encryption or SDBM is configured for this LDAP server, and program control (WHEN(PROGRAM) is active.
- ▶ These members should be updated:
 - ADDMEM is added to the ** definition. Be sure the started task user (GLDSRV) has read access to ** in the PROGRAM class.
 - RALTER produced by LDAPCNF specifies volumes, which are no longer necessary:
RALTER PROGRAM ** ADDMEM('SYS1.SGLDLNK'/'*****'/NOPADCHK)
You can update all these RACF commands:
RALTER PROGRAM ** ADDMEM('SYS1.SGLDLNK'//NOPADCHK)

7.4 Executing DB2 jobs

Before executing these two members produced by LDAPCNF, be sure that DB2 is started; if not, issue the command:

```
prefix START DB2
```

where **prefix** for this DB2 is obtained by the **D OPDATA** command.

- ▶ DBCLI JCL binds the CLI packages to DB2 and the DSNACLI plan. Therefore, this job must be executed under a userid which has SYSADM authority.

- The SPUFI member must be executed through the DB2 SPUFI interactive tool.

7.5 Start the LDAP server

To start the LDAP server in SDSF, if LDAPUSRID='GLDSRV' enter:

```
/S GLDSRV
```

```
z/OS Version 1 Release 2 Security Server LDAP Server
Reading configuration file //DD:CONFIG.
The useNativeAuth configuration option SELECTED has been enabled.
OCSF setup failed, but pwEncryption method 'none' is available.
Password encryption method 'none' is enabled in the TDBM backend.
Backend capability listing follows:
Backend type: tdbm, Backend ID: TDBM BACKEND, Backend suffix: O=IBMMOP::
Capability: LDAP_Backend_ID      Value: TDBM BACKEND
Capability: LDAP_Backend_BldDateTime Value: 2001-10-30-17.21.51.000000
Capability: LDAP_Backend_APARLevel Value: LDAP
Capability: LDAP_Backend_Release Value: R 2.0
Capability: LDAP_Backend_Version Value: V 1.0
Capability: LDAP_Backend_Dialect Value: DIALECT 1.0
Capability: LDAP_Backend_BerDecoding Value: BINARY
Capability: LDAP_Backend_ExtGroupSearch Value: YES
Capability: LDAP_Backend_krbIdentityMap Value: YES
Capability: supportedControl Value: 2.16.840.1.113730.3.4.2
Capability: supportedControl Value: 1.3.18.0.2.10.2
End of capability listing for Backend type: tdbm, Backend ID: TDBM BACKEND, Backend
suffix: O=IBMMOP.
Backend type: sdbm, Backend ID: SDBM BACKEND, Backend suffix: SYSPLEX=PORTINFO::
Capability: LDAP_Backend_ID      Value: SDBM BACKEND
Capability: LDAP_Backend_BldDateTime Value: 2001-07-10-13.39.57.000000
Capability: LDAP_Backend_APARLevel Value: LDAP
Capability: LDAP_Backend_Release Value: R 2.0
Capability: LDAP_Backend_Version Value: V 1.0
Capability: LDAP_Backend_Dialect Value: DIALECT 1.0
Capability: LDAP_Backend_BerDecoding Value: STRING
Capability: LDAP_Backend_ExtGroupSearch Value: YES
Capability: LDAP_Backend_krbIdentityMap Value: YES
Capability: supportedControl Value: 2.16.840.1.113730.3.4.2
Capability: supportedControl Value: 1.3.18.0.2.10.2
End of capability listing for Backend type: sdbm, Backend ID: SDBM BACKEND, Backend
suffix: SYSPLEX=PORTINFO.
Backend capability listing ended.
Configuration file successfully read.
Nonsecure communication is active for IP: INADDR_ANY, nonsecure port: 3389.
Slapd is ready for requests.
```

Figure 7-8 Sysout for GLDSRV

7.6 Finalize setup of LDAP server

The tasks remaining to complete the LDAP server setup are described in this section.

7.6.1 Load schemas

Multiple schemas may need to be loaded before applications that use the directory will work.

In addition to schemas loaded statically in SLAPDCONF, *schema.user.ldif* must be dynamically loaded, and it's recommended also to load *schema.IBM.ldif*

- ▶ Copy *schema.user.ldif* and *schema.IBM.ldif* from /usr/lpp/ldap/etc into your local directory. This was /usr/local/alains/ldap for our tests.
- ▶ For both schema files, modify the TDBM suffix in the line:
dn: cn=schema, <TDBM_suffix>
according to TDBM_SUFFIX='o=IBMMOP' in ldap.profile.
- ▶ Use the **ldapmodify** utility to modify the schema entry, as shown in Figure 7-9.

```
/bin/ldapmodify -h 9.100.203.110 -p 3389 +  
-D "cn=LDAP Administrator,o=IBMMOP" -w secret +  
-f /usr/local/alains/ldap/schema.user.ldif  
/bin/ldapmodify -h 9.100.203.110 -p 3389 +  
-D "cn=LDAP Administrator,o=IBMMOP" -w secret +  
-f /usr/local/alains/ldap/schema.IBM.ldif
```

Figure 7-9 ldapmodify for schema modification

7.6.2 Load the suffix entry for the TDBM backend

- ▶ Create a suffix.ldif file in your home directory according to TDBM_SUFFIX='o=IBMMOP' in ldap.profile.

```
dn: o=IBMMOP  
objectclass: top  
objectclass: organization  
o: IBMMOP
```

Figure 7-10 suffix.ldif hfs file

- ▶ Load this suffix with an **ldapadd** command.

```
/bin/ldapadd -h 9.100.203.110 -p 3389 +  
-D "cn=LDAP Administrator,o=IBMMOP" -w secret +  
-f /usr/local/alains/ldap/suffix.ldif
```

Figure 7-11 ldapadd command

7.6.3 Verify configuration

This **ldapsearch** utility can be run as an IVP to confirm that the LDAP server configuration was successful.

```
/bin/ldapsearch -h 9.100.203.110 -p 3389 -V 3 +  
-s base -b "" "objectclass=*"
```

Figure 7-12 ldapsearch: List of naming context

The result of this command is shown in Figure 7-13 on page 228.

```
supportedcontrol=2.16.840.1.113730.3.4.2
supportedcontrol=1.3.18.0.2.10.2
namingcontexts=o=IBMMOP
namingcontexts=sysplex=portinfo
subschemasubentry=CN=SCHEMA,o=IBMMOP
supportedsaslm mechanisms=EXTERNAL
supportedldapversion=2
supportedldapversion=3
ibmdirectoryversion=z/OS V1R2
```

Figure 7-13 list of naming context

7.7 Implementing password encryption with z/OS LDAP

This section describes the steps for implementing password encryption for the TDBM backend of the z/OS LDAP server. Password encryption can be used for the userPassword attribute, which is stored in the TDBM and RDBM backend. The use of password encryption will help prevent unauthorized access to users' passwords by storing the passwords in an encrypted fashion while they are within the LDAP directory or when they are being backed up.

To implement password encryption, the decision must first be made as to which type of encryption is required to meet the business and security needs. The options are:

- ▶ None (no password encryption)
- ▶ Crypt, MD5, or SHA (one-way hash stored passwords)
- ▶ DES (two-way encryption)

The one-way hashed formats (crypt, MD5, or SHA) allow passwords to be encrypted once and they stay in that format. If an application needs to bind to these passwords, the password to be verified is hashed and checked against the hashed value stored in the LDAP directory. In some cases, an application will require the password in the clear. In these cases, two-way encryption is required. With two-way encryption, the userPassword attribute is encrypted and stored within the TDBM or RDBM. When the password is required for a bind operation, the password to be verified is encrypted and checked against the encrypted value stored in the LDAP directory, similar to the one-way hashing method. But if the client application is authorized, the two-way encrypted password could be passed back to the application in the clear for processing. In this fashion middle-tier authentication servers and reverse proxies have the potential of using this password for their authentication purposes.

Once the desired encryption method is determined, then the system requirements are identified. No special system requirements are needed to implement either the none or crypt options. For MD5 and SHA, z/OS Open Cryptographic Services Facility (OCSF) must be set up and functional. For DES, both OCSF and z/OS Integrated Cryptographic Service Facility (ICSF) must be set up and functional. On our test system ICSF was not available, so we did not work with the DES option. The implementation and description of OCSF and ICSF are explained in *OS/390 Open Cryptographic Services Facility Application Developer's Guide and Reference*, SC24-5875-02 for OCSF and the *OS/390 Integrated Cryptographic Service Facility Administrator's Guide*, SC23-3975-08 for ICSF.

Pay careful attention to the following hints and guidelines during the implementation of OCSF.

- ▶ Define all the CDS.** profiles within the RACF FACILITY class, and give the LDAP server (or its RACF group) READ access to the required profiles. Be sure to refresh the RACLISTed FACILITY class. Our RACF commands are shown in Example 7-1.

Example 7-1

```
//JONESD1 JOB CLASS=H,MSGCLASS=0,NOTIFY=???????
//STEP01 EXEC PGM=IKJEFT01
//SYSLBC DD DSN=SYS1.BROADCAST,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *

RDEF FACILITY CDS.CSSM UACC(NONE)
RDEF FACILITY CDS.CSSM.CRYPTO UACC(NONE)
RDEF FACILITY CDS.CSSM.DATALIB UACC(NONE)

PE CDS.CSSM.DATALIB CLASS(FACILITY) ID(JONES) ACC(READ)
PE CDS.CSSM.CRYPTO CLASS(FACILITY) ID(JONES) ACC(READ)
PE CDS.CSSM CLASS(FACILITY) ID(JONES) ACC(READ)
PE CDS.CSSM.DATALIB CLASS(FACILITY) ID(LDAPRACF) ACC(READ)
PE CDS.CSSM.CRYPTO CLASS(FACILITY) ID(LDAPRACF) ACC(READ)
PE CDS.CSSM CLASS(FACILITY) ID(LDAPRACF) ACC(READ)

SETR RACLIST(FACILITY) REFRESH
/*
/*
/* RALT PROGRAM * ADDMEM('SYS1.CRYPTO.SGSKLOAD'//NOPADCHK)
/* RALT PROGRAM * ADDMEM('SYS1.LDAP.SGLDLNK'//NOPADCHK)
/* RALT PROGRAM * ADDMEM('SYS1.LEMVS.SCEERUN'//NOPADCHK)
/* SETR WHEN(PROGRAM) REFRESH
/* PE BPX.SERVER CLASS(FACILITY) ID(LDAPRACF) ACC(READ)
/* SETR RACLIST(FACILITY) REFRESH
/*
```

Note: The last 6 RACF commands are commented out because they had already been done on our system. Leave them in if they have not already been done on your system.

- ▶ The LDAP server's RACF userid has to be permitted to the BPX.SERVER profile to use the OCSF functions. This means that datasets used by this server will need to be RACF PROGRAM protected. This is usually done in a normal z/OS LDAP server implementation but, in case it has not been done yet, here is an example of our RACF commands to permit the LDAP server to BPX.SERVER and to PROGRAM protect its datasets. The LDAP, DB2, and C runtime load libraries are probably already RACF protected. The other required load libraries to be protected are the System SSL and ICSF load libraries. (See the JCL presented in Example 7-1.)
- ▶ There are several executables in HFS files that are required for OCSF to run successfully. These must also be RACF PROGRAM protected, as well as APF authorized. The SMP/E installation turns on the extended attribute bits for these files. To check the setting of the files to ensure that they are set appropriately, use the UNIX commands shown in Example 7-2 from the OMVS shell environment.

Example 7-2

```
/usr/lpp/ocsf/lib: >ls -alE
.....
drwxr-xr-x      2 WEBSERV OMVSGRP      8192 Jul 26 18:20 IBM
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    49152 Jul 27 07:50 cdsrprt.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    90112 Jul 27 07:50 cdsibmut.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    98304 Jul 27 07:50 cdskwtf.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    65536 Jul 27 07:50 cdskwucs.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP   3600384 Jul 27 07:50 cdsnspsp.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    192512 Jul 27 07:50 cdsport.dll
-rwxr-xr-x  aps-  2 WEBSERV OMVSGRP    188416 Jul 27 07:50 cdsrandm.dll
```

```

-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP  1323008 Sep  6 16:10 cssm32.dll
-rw-r--r--  --s-  2 WEBSERV  OMVSGRP    12400 Jul 11 2001 cssm32.x
lrwxrwxrwx    1 WEBSERV  OMVSGRP    16 Aug 11 12:14 cssmmanp.dll -> cssmmanp_s13.dll
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    36864 Jul 27 07:50 cssmmanp_s13.dll
lrwxrwxrwx    1 WEBSERV  OMVSGRP    16 Aug 11 12:14 cssmusep.dll -> cssmusep_s13.dll
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    36864 Jul 27 07:50 cssmusep_s13.dll
/usr/lpp/ocsf/lib: >cd ..
/usr/lpp/ocsf: >cd bin
/usr/lpp/ocsf/bin: >ls -alE
...
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    61440 Jul 27 07:49 install_cssm
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    65536 Jul 27 07:49 install_ibmcca
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    65536 Jul 27 07:49 install_ibmcl
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    729088 Jul 27 07:49 install_ibmcl2
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    733184 Jul 27 07:49 install_ibmdl2
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    729088 Jul 27 07:49 install_ibmocepd1
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    69632 Jul 27 07:49 install_ibmoceptp
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    65536 Jul 27 07:49 install_ibmswcsp
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    65536 Jul 27 07:49 install_ibmtp
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    729088 Jul 27 07:49 install_ibmtp2
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    65536 Jul 27 07:49 install_ibmwkcsp
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    86016 Jul 27 07:49 install_ldapd1
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    233472 Jul 27 07:49 kwtfreport
-rwxr-xr-x  --s-  2 WEBSERV  OMVSGRP    1603 Jul 11 2001 ocep_install
-rwxr-xr-x  --s-  2 WEBSERV  OMVSGRP    1003 Jul 11 2001 ocep_uninstall
-rwxr-xr-x  --s-  2 WEBSERV  OMVSGRP    6008 Jul 11 2001 ocsf_install_crypto
-rwxr-xr-x  --s-  2 WEBSERV  OMVSGRP    2394 Jul 11 2001 ocsf_uninstall_crypto
-rwxr-xr-x  aps-  2 WEBSERV  OMVSGRP    53248 Jul 27 07:49 regdump

```

Note: The APF and Program Controlled bits should be set for all the OCSF DLLs and install scripts in the lib, addins, and bin directories. The DLLs in the addins directory have a file type of *.so).

- If, for some reason, the extended attribute bits are not set correctly, use the RACF TSO commands shown in Example 7-3 to turn on the access to the extended attributes. After the RACF environment is set up so the extended attributes can be adjusted, use the following UNIX commands from the OMVS environment to set the extended attributes for the following files.

Example 7-3

```

/usr/lpp/ocsf/bin: > extattr +a install*
/usr/lpp/ocsf/bin: > extattr +p install*
/usr/lpp/ocsf/lib: > extattr +p *.dll
/usr/lpp/ocsf/lib: > extattr +a *.dll
/usr/lpp/ocsf/addins: > extattr +a *.so
/usr/lpp/ocsf/addins: > extattr +p *.so

```

- The OCSF installation script must be run from a userid with READ access to the CDS.** profiles and the BPX.SERVER profile in the RACF FACILITY class. From the OMVS environment, issue the following UNIX commands. (See the JCL above.)
- Once the OCSF environment has been set up, the LDAP server needs to know where the DLLs are stored. The OCSF DLLs are pointed to by three symbolic links that have been placed in the /usr/lib directory. These are cssmmanp.dll, cssmusep.dll, and cssm32.dll. If the attributes of these symbolic links are reviewed, it will be noted that they really point to the DLLs in /usr/lpp/ocsf/ lib. The LIBPATH for the LDAP server needs to know where these symbolic links are. This can be done in one of two different methods. One way is to

update the LIBPATH environment variable in the ENVVARS file, similar to Example 7-4. If the ENVVARS file cannot be updated, then the JCL statements in the LDAP PROC can be updated. The syntax in the JCL must be exact.

Example 7-4

```
Our ENVVARS file:
LANG=En_US.IBM-1047
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/En_US.IBM-1047/%N:/usr/lpp/ldap/lib/nls/msg
/En_US.IBM-1047/%N
LIBPATH=/usr/lib
```

or, if needed, update your PROC JCL, change the EXEC statement:

```
//GO EXEC PGM=GLDSLAPD,REGION=&REGSIZE,TIME=1440,
//      PARM=('/&PARMS >DD:SLAPDOUT 2>&1')
to the following:
//GO      EXEC PGM=GLDSLAPD,REGION=&REGSIZE,TIME=1440,
// PARM=('ENVAR("LIBPATH=/usr/lib")/&PARMS >DD:SLAPDOUT 2>&1')
```

- Once you get OCSF set up, restart the LDAP server. There should be messages in the LDAP output which indicate if there is a problem. If more detailed messages are needed, then set the LDAP debug option (for this situation we suggest setting it to 32769).

Messages when starting LDAP server without OCSF setup and pwEncryption set to none.

```
GLD0022I z/OS Version 1 Release 2 Security Server LDAP Server
Starting slapd.
GLD0010I Reading configuration file //DD:CONFIG.
GLD0053I Configuration read security of none.
GLD0185I Connections allowed only on the nonsecure port.
GLD0187I Configuration parameter security is being ignored since it was specified with the
listen parameter.
GLD3047E OCSF setup failed, but pwEncryption method 'none' is available.
GLD3062I Password encryption method 'none' is enabled in the TDBM backend.
GLD0163I Backend capability listing follows:
GLD0166I Backend type: tdbm, Backend ID: TDBM BACKEND, Backend suffix: 0=WSC::
```

Messages when starting LDAP server with OCSF setup correctly and pwEncryption set to none. (Note that the OCSF failed message is not listed.)

```
GLD0022I z/OS Version 1 Release 2 Security Server LDAP Server
Starting slapd.
GLD0010I Reading configuration file //DD:CONFIG.
GLD0053I Configuration read security of none.
GLD0185I Connections allowed only on the nonsecure port.
GLD0187I Configuration parameter security is being ignored since it was specified with the
listen parameter.
GLD3062I Password encryption method 'none' is enabled in the TDBM backend.
GLD0163I Backend capability listing follows:
GLD0166I Backend type: tdbm, Backend ID: TDBM BACKEND, Backend suffix: 0=WSC::
```

Messages when starting LDAP server without OCSF setup correctly and pwEncryption set to SHA. (Note that this will fail configuration for this backend. If other backends have been configured for this LDAP server, this will be processed independently and may configure successfully.)

```
Starting slapd.  
GLD0010I Reading configuration file //DD:CONFIG.  
GLD0053I Configuration read security of none.  
GLD0185I Connections allowed only on the nonsecure port.  
GLD0187I Configuration parameter security is being ignored since it was specified with the  
listen parameter.  
GLD3046E OCSF setup failed, encryption method 'SHA' is not available.  
GLD0141I A backend (A7D2790) of type tdbm failed to configure.
```

Messages when starting LDAP server with OCSF setup and pwEncryption set to SHA (or MD5).

```
Starting slapd.  
GLD0010I Reading configuration file //DD:CONFIG.  
GLD0053I Configuration read security of none.  
GLD0185I Connections allowed only on the nonsecure port.  
GLD0187I Configuration parameter security is being ignored since it was specified with the  
listen parameter.  
GLD3062I Password encryption method 'SHA' is enabled in the TDBM backend.  
GLD0163I Backend capability listing follows:  
GLD0166I Backend type: tdbm, Backend ID: TDBM BACKEND, Backend suffix: O=WSC::
```

For the z/OS LDAP server to use password encryption, set the `pwEncryption` parameter in the `slapd.conf` file. After the z/OS LDAP server has been refreshed, all entries that have a `userPassword` attribute either added or modified will have the `userPassword` attribute encrypted. For entries that already exist, the `db2pwdcn` utility can be used to convert existing `userPassword` attributes.

When the LDAP directory is backed up using the `tdbm2ldif` utility, the `userPassword` attribute is displayed in LDIF (LDAP Data Interchange Format) format, in encrypted format with a tag identifying the method of encryption.

LDAP with RACF digital certificates

This chapter is divided into three sections. The first section describes the background setup that is required to use RACF digital certificate support. The second section walks through the steps to set up an existing LDAP server using RACF digital certificates for server authentication. The third section extends the RACF digital certificate setup so that the LDAP client can use their RACF digital certificates for authentication. All the steps in the second and third sections use the RACDCERT command so that the individual steps can be highlighted. The Web application, PKISERV, which RACF introduced with APARs OW45211 and OW45212, would reduce the setup process and the required knowledge of RACDCERT commands. PKISERV and its documentation is available at:

<http://www-1.ibm.com/servers/eserver/zseries/zos/racf/webca.html>

8.1 Preparing to use RACDCERT commands

The LDAP server and whoever is building the digital certificates must be allowed to maintain and review the RACF keyrings and RACF digital certificates. To allow these entities access to the RACF digital certificate environment, there are several profiles within the FACILITY that must be set up. A sample of this setup is shown in Example 8-1. The person who was creating and maintaining the RACF digital certificate environment is represented by the RACF GROUP of SYSPROG. For the appropriate levels of authority required in your installation, review the *RACF Security Administrator's Guide*, SC28-1915. The LDAP server only needs to be able to interrogate the RACF keyrings and digital certificates, so they will need READ access to the IRR.DIGTCERT.LISTRING and the IRR.DIGTCERT.LIST profiles. For more details on the LDAP server's and client's requirements, see the *z/OS Security Server LDAP Client Programming*, SC24-5924 and *z/OS Security Server LDAP Server Administration and Use*, SC24-5923.

Example 8-1

```
/* ***** */
/* Define the profiles to protect certificate maintenance */
/* ***** */
RDEFINE FACILITY IRR.DIGTCERT.*          UACC(NONE)
RDEFINE FACILITY IRR.DIGTCERT.LISTRING  UACC(NONE)
RDEFINE FACILITY IRR.DIGTCERT.LIST      UACC(NONE)
/* ***** */
/* Permit the appropriate groups to the profiles */
/* ***** */
PE IRR.DIGTCERT.*          CLASS(FACILITY) ID(SYSPROG) ACCESS(CONTROL)
PE IRR.DIGTCERT.LISTRING  CLASS(FACILITY) ID(SYSPROG) ACCESS(CONTROL)
PE IRR.DIGTCERT.LIST      CLASS(FACILITY) ID(SYSPROG) ACCESS(CONTROL)
PE IRR.DIGTCERT.LISTRING  CLASS(FACILITY) ID(LDAPSRV) ACCESS(READ)
PE IRR.DIGTCERT.LIST      CLASS(FACILITY) ID(LDAPSRV) ACCESS(READ)
/* ***** */
/* Refresh the FACILITY class. */
/* ***** */
SETR RACLST(FACILITY) REFRESH
```

8.2 Server authentication

Once the RACF protection is set up, the Certificate Authority's digital certificate can be built. In our example, this will be a self-signed certificate that will be used to sign all the other certificate requests.

```
/* ***** */
/* Step 1 - Create the CertAuth certificate */
/* ***** */
RACDCERT CERTAUTH GENCERT                +
  SUBJECT ( CN('RACF CERTAUTH CERTIFICATE') +
            T('MASTER CERTIFICATE')        +
            OU('zSERIES')                    +
            O('ETPTEST')                     +
            L('POKTOWN')                     +
            SP('NEW YORK')                    +
            C('US')                          +
            )                                +
  NOTBEFORE(DATE(2001/12/31))                +
  NOTAFTER(DATE(2010/12/31))                 +
  SIZE(1024)                                +
  WITHLABEL('RACF CERTAUTH CERT')           +
  +
```



```
KEYUSAGE(CERTSIGN)
```

```
RACDCERT CERTAUTH LIST(LABEL('RACF CERTAUTH CERT'))
```

The Certificate Authority's digital certificate could be (and usually is) a "trusted" digital certificate imported from a company such as Entrust, Verisign, Thawte, etc. As mentioned, in our case, it is a self-signed trusted RACF-generated digital certificate that we will use as our trusted root. The first RACDCERT command generates the self-signed trusted root. Anytime the trusted root needs to be referred to in a RACDCERT command, this will be done by referring to its LABEL of 'RACF CERTAUTH CERT'. The usage of the trust root is to sign certificate requests, as identified by the KEYUSAGE parameter. The second RACDCERT in the example will display the newly created certificate.

After the trusted root certificate is created, then the LDAP server's digital certificate needs to be created and signed with the trusted root certificate (that is, the Certificate Authority certificate). Since the trusted root was created within the same RACF environment as this certificate, both the generation of the certificate request and the signing of the certificate request can be done in one RACDCERT command. In other cases, these might have to be performed in two or more commands.

```
/* *****  
/* Step 2 - Create the LDAP Server Cert and Sign with the CA Cert */  
/* *****  
RACDCERT ID(CBLDAP) GENCERT +  
  SUBJECT ( CN('LDAP SERVER CERTIFICATE') +  
            T('SERVER CERTIFICATE') +  
            OU('zSERIES') +  
            O('ETPTEST') +  
            L('POKTOWN') +  
            SP('NEW YORK') +  
            C('US') ) +  
  NOTAFTER(DATE(2010/12/31)) +  
  NOTBEFORE(DATE(2001/12/31)) +  
  SIZE(1024) +  
  WITHLABEL('LDAP SERVER CERT') +  
  SIGNWITH( CERTAUTH LABEL('RACF CERTAUTH CERT'))  
  
RACDCERT ID(CBLDAP) LIST(LABEL('LDAP SERVER CERT'))
```

The first RACDCERT command creates the LDAP server digital certificate. The RACF userid, CBLDAP, is the userid associated with the LDAP server started task. This certificate has a label of 'LDAP SERVER CERT' and is signed with the self-signed trusted root that was created in the previous step. The second RACDCERT command will list the newly created LDAP server certificate.

Now that the certificates have been created, they have to be stored in a keyring so they can be used by the servers. To do this, a keyring for the LDAP server must be built.

```
/* *****  
/* Step 3 - Create the LDAP Server Keyring. */  
/* *****  
RACDCERT ID(CBLDAP) ADDRING(LDAP_Server_RACF_Keyring)  
  
RACDCERT ID(CBLDAP) LISTRING(LDAP_Server_RACF_Keyring)  
  
RACDCERT ID(CBLDAP) LISTRING(*)
```

The first RACDCERT command creates a RACF keyring that is associated with the CBLDAP userid. CBLDAP is the RACF userid that the LDAP server runs with. The RACF keyring that is created is labelled 'LDAP_Server_RACF_Keyring'. The second RACDCERT command will list all the digital certificates within the newly created keyring. The third RACDCERT command will list all the keyrings associated with the CBLDAP userid.

The next step is to place the appropriate digital certificates within the LDAP server's RACF keyring. The first RACDCERT (in step 4a) connects our self-signed trusted root, which is labelled 'RACF CERTAUTH CERT' to the LDAP server's keyring which is labelled 'LDAP_Server_RACF_keyring'. The usage of this trusted root digital certificate is identified as the Certificate Authority or the base of our certificate chain. Note that LDAP server's RACF userid, CBLDAP, is the owner of this keyring.

The second RACDCERT, which is in step 4b, is to connect the LDAP server's digital certificate to the LDAP server's keyring. CBLDAP is the owner of both the RACF keyring and the digital certificate. The usage of this digital certificate is for the personal use of this server: in other words, this digital certificate cannot be used for other functions such as signing other certificate requests. The LDAP server's digital certificate is also marked as the DEFAULT digital certificates. By marking it as the default digital certificate, it allows other digital certificates to be added to this keyring if needed. These additional digital certificates can be referenced by their labels if needed.

The last RACDCERT command will list all the digital certificates that have been connected to the 'LDAP_Server_RACF_Keyring'. Two different digital certificates, with their labels and appropriate usages, should be listed.

```

/*****
/* Step 4a - Connect the RACF CA Cert to LDAP Server keyring */
/*****
RACDCERT ID(CBLDAP) CONNECT          +
  ( CERTAUTH                        +
    LABEL('RACF CERTAUTH CERT')    +
    USAGE(CERTAUTH)                 +
    RING(LDAP_Server_RACF_Keyring) )
/*****
/* Step 4b - Connect the LDAP Server Cert to the Server keyring */
/*****
RACDCERT ID(CBLDAP) CONNECT          +
  ( ID(CBLDAP)                      +
    LABEL('LDAP SERVER CERT')      +
    RING(LDAP_Server_RACF_Keyring) +
    DEFAULT                         +
    USAGE(PERSONAL)                 )
RACDCERT ID(CBLDAP) LISTRING(LDAP_Server_RACF_Keyring)

```

The output from the last RACDCERT command should look like:

Digital ring information for user CBLDAP:
Ring:
 >LDAP_Server_RACF_Keyring<

Certificate Label Name	Cert Owner	USAGE	DEFAULT
-----	-----	----	-----
RACF CERTAUTH CERT	CERTAUTH	CERTAUTH	NO
LDAP SERVER CERT	ID(CBLDAP)	PERSONAL	YES

The RACF digital certificate environment is set up, but the LDAP server must be configured to use digital certificates and the LDAP client must be set up to use RACF digital certificates. The next couple of steps will set up the LDAP client to be able to use RACF digital certificates for server authentication. The setup for client authentication will be done in the next section.

In our case, the LDAP client is going to be a RACF userid who is doing LDAP commands, such as LDAPSEARCH, from either TSO or OMVS. The RACF user must have a RACF keyring that contains the trusted root. This is what is used to validate the LDAP server's digital certificate for server authentication.

```

/*****
/* Step 5 - Create the LDAP Client's Keyring.      */
/*****
RACDCERT ID(KARIN) ADDRING(KARIN_RACF_Keyring)

RACDCERT ID(KARIN) LISTRING(KARIN_RACF_Keyring)

RACDCERT ID(KARIN) LISTRING(*)

```

First a RACF keyring must be created for this RACF userid, the LDAP client. In this example, KARIN is the RACF userid and the keyring has a label of 'KARIN_RACF_Keyring'. The second RACDCERT command will list all the digital certificates in this newly created keyring, which will be empty. The third RACDCERT will list all the keyrings that KARIN owns and any digital certificates that might be connected to those keyrings.

The last step in this set up is to connect the self-signed trusted root certificate to the LDAP client's keyring.

```

/*****
/* Step 6 - Connect the RACF CA Cert to the LDAP client keyring */
/*****
RACDCERT ID(KARIN) CONNECT          +
( CERTAUTH                        +
  LABEL('RACF CERTAUTH CERT')    +
  USAGE(CERTAUTH)                 +
  RING(KARIN_RACF_Keyring) )

RACDCERT ID(KARIN) LISTRING(KARIN_RACF_Keyring)

```

In this RACDCERT command, the digital certificate with the label of 'RACF CERTAUTH CERT' is connected to the RACF keyring with the label 'KARIN_RACF_Keyring'. The second RACDCERT command will list the digital certificates with KARIN's keyring and their usage.

Now everything is set up to use server authentication within the RACF environment. Next, the LDAP server's configuration file has to be updated to indicate that server authentication and SSL are going to be used. To do this, the appropriate statements are added to the slapd.conf file. More details on these statements are found in *z/OS Security Server LDAP Server Administration and Use*, SC24-5923.

```

verifySchema on
sizeLimit 500
timeLimit 3600
listen ldap://:389
listen ldaps://:636
adminDN "cn=karin,ou=pd,o=etptest,c=us"
sslAuth serverAuth
sslCipherspecs 15104

```

```

sslKeyRingFile LDAP_Server_RACF_Keyring
validateincomingV2strings yes
sendV3stringsoverV2as UTF-8
# -----
database sdbm GLDBSDBM
suffix "racfdb=p24"
# -----
database tdbm GLDBTDBM
suffix "o=etptest,c=us"
suffix "secAuthority=Default"
servername DBVG
dbuserid CBLDAP
databasename BBOTLDAP
pwEncryption none
useNativeAuth selected
nativeUpdateAllowed yes
nativeAuthSubtree "ou=pd,o=etptest,c=us"
nativeAuthSubtree "ou=na,o=etptest,c=us"

```

This is the slapd.conf that was used for these examples. This slapd.conf file was the same one that was used for the testing of Policy Director 3.8 and Native Authentication. It also has the SDBM and TDBM environments set up for the LDAP server. The statements within this slapd.conf that are in bold print are the statements that were added to configure the LDAP server for server authentication and SSL. The highlighted 'listen' statement identifies the secure port. The 'sslAuth' statement indicates whether server authentication and/or client authentication is going to be used. The 'sslCipherspecs' statement denotes which algorithms and strengths are supported by this LDAP server when using SSL connections. And the 'sslKeyRingFile' will indicate the keyring that this LDAP server will use to house its digital certificates. Since the 'sslKeyRingFile' is pointing at a RACF keyring, there is no need to specify a keyring password or stash file, as you would if you were using a keyring stored in the HFS. The RACF keyring must be owned by the RACF userid that is associated with the LDAP server for this setup to work successfully.

Once the LDAP server is configured and started, SSL and server authentication is now supported. Now consider what happens when KARIN logs onto TSO or gets into OMVS, and issues the following command (this example was issued from OMVS):

```

ldapsearch -h 127.0.0.1 \
-D racfid=karin,profiletype=user,racfdb=p24 -w ???????? \
-Z -K KARIN_RACF_Keyring \
-b "o=etptest,c=us" "objectclass=*"

```

In this example, the LDAP server authenticates itself to the LDAP client by sending the LDAP server's digital certificate to the client. The client (KARIN, in this case) validates the LDAP server's digital certificate by checking the client's keyring to see if the signing certificate (the self-signed trusted root) is viewed as trusted. Since the RACF self-signed Certificate Authority certificate is connected to the client's keyring as a trusted CertAuth, the LDAP server's digital certificate is viewed as valid and trusted. Once the digital certificate is validated, then the SSL connection can be established and KARIN's RACF userid and password can pass through the connection securely. In this example, the -Z indicates that SSL is required for this communication. The -K indicates that KARIN_RACF_Keyring is the client keyring that is to be used for the validation of the LDAP server's digital certificate.

Another example of what can and cannot be done in this environment is that RACF keyrings can only be used by their associated RACF userids. For example, consider that Chris logs onto the system and issues the following OMVS command:

```
ldapsearch -h 127.0.0.1 \
-D racfid=chris,profiletype=user,racfdb=p24 -w ???????? \
-Z -K KARIN_RACF_Keyring \
-b "o=etptest,c=us" "objectclass=*"

```

Instead of getting the listing of the etptest organization, Chris would get a message that looks similar to:

```
ldap_ssl_client_init failed! rc == 113, failureReasonCode == 2
reason text: Initialization call failed

```

This indicates that the SSL initialization failed because Chris was trying to use KARIN's RACF keyring. That is, Chris does not have a keyring called KARIN_RACF_Keyring or access to Karin's keyring which is called KARIN_RACF_Keyring.

Native Authentication can be used with RACF keyring. For example, KARIN has been defined in the TDBM with Native Authentication. So now KARIN can issue the following command:

```
ldapsearch -h 127.0.0.1 \
-D cn=karin,ou=pd,o=etptest,c=us -w ?racfpw? \
-Z -K KARIN_RACF_Keyring \
-b "o=etptest,c=us" "objectclass=*"

```

In this example, KARIN's TDBM DN, which is the same as the DN in the digital certificate, is authenticated with the associated RACF password. KARIN can use the RACF keyring that is owned by KARIN in this example.

8.3 Client Authentication

Building upon the work completed in the server authentication section, the LDAP server can be set up to support client authentication. This will allow LDAP clients to authenticate themselves with digital certificates instead of userid and password combinations. In our examples, RACF digital certificates and keyrings will be used, but digital certificates could be from other sources and the keyrings could be stored in HFS files as well.

To set up the LDAP client's RACF digital certificate environment, a digital certificate for the LDAP client must be created and signed with a trusted Certificate Authority certificate.

```

/*****
/* Step 7 - Create the LDAP Client Cert and Sign with the CA Cert */
/*****
RACDCERT ID(KARIN) GENCERT +
SUBJECT ( CN('KARIN') +
          T('MASTER LDAP USER') +
          OU('zSERIES FTSS') +
          O('GI Team') +
          L('North Cold') +
          SP('Minneapolis') +
          C('US') ) +
NOTAFTER(DATE(2010/12/31)) +
NOTBEFORE(DATE(2001/12/31)) +

```

```

SIZE(1024)
WITHLABEL('KARIN Personal Cert')
SIGNWITH( CERTAUTH LABEL('RACF CERTAUTH CERT'))

```

```

RACDCERT ID(KARIN) LIST(LABEL('KARIN Personal Cert'))

```

In this example, the LDAP client's (the RACF user, KARIN, in this case) certificate request is signed by the self-signed trusted root that was created in the server authentication section. This self-signed trusted root certificate has the label of 'RACF CERTAUTH CERT'. The LDAP client's digital certificate is identified with the label of 'KARIN Personal Cert'. Since both the LDAP client's certificate request and the trusted root are within the same RACF system, the generation of the LDAP client's digital certificate can be completed with one RACDCERT command. If these are separate systems, then the RACF certificate request will have to be shipped to the system containing the trusted root certificate for signing. Then the signed LDAP client certificate will have to be returned to the LDAP client's system and imported into the LDAP client's keyring.

In our example, the signed LDAP client certificate now has to be connected to the LDAP client's keyring that was created in the server authentication section. The trusted root certificate was already connected to the LDAP client's keyring in the previous section.

```

/*****
/* Step 8a - RACF CA Cert already connected to client's keyring */
/*           in previous step - otherwise this must be connected */
/*****
/*****
/* Step 8b - Connect the Client's Cert to the Client's keyring */
/*****
RACDCERT ID(KARIN) CONNECT
( ID(KARIN)
  LABEL('KARIN Personal Cert')
  RING(KARIN_RACF_Keyring)
  DEFAULT
  USAGE(PERSONAL) )

```

```

RACDCERT ID(KARIN) LISTRING(KARIN_RACF_Keyring)

```

The first RACDCERT command connects the digital certificate labelled 'KARIN Personal Cert' to the LDAP client's keyring labelled 'KARIN_RACF_Keyring'. This digital certificate is marked as the default digital certificate and is to be used for personal use; that is, not as a certificate authority. The second RACDCERT command will display the LDAP client's keyring and all the digital certificates in that keyring. The output of that command will look like:

```

RACDCERT ID(KARIN) LISTRING(KARIN_RACF_Keyring)

```

Digital ring information for user KARIN:

```

Ring:
>KARIN_RACF_Keyring<

```

Certificate Label Name	Cert Owner	USAGE	DEFAULT
RACF CERTAUTH CERT	CERTAUTH	CERTAUTH	NO
KARIN Personal Cert	ID(KARIN)	PERSONAL	YES

The last thing that needs to be changed is the LDAP configuration file, slapd.conf. The only change that needs to be made is to set the sslAuth statement to serverClientAuth. This indicates that the LDAP server will use server authentication and then, if requested by the client, it will accept a client digital certificate for authentication. If the client does not want to use digital certificates, then userid and password combinations can still be used.

```
verifySchema on
sizeLimit 500
timeLimit 3600
listen ldap://:389
listen ldaps://:636
adminDN "cn=karin,ou=pd,o=etptest,c=us"
sslAuth serverClientAuth
sslCipherspecs 15104
sslKeyRingFile LDAP_Server_RACF_Keyring
validateincomingV2strings yes
sendV3stringsoverV2as UTF-8
# -----
database sdbm GLDBSDBM
suffix "racfdb=p24"
# -----
database tdbm GLDBTDBM
suffix "o=etptest,c=us"
suffix "secAuthority=Default"
servername DBVG
dbuserid CBLDAP
databasename BBOTLDAP
pwEncryption none
useNativeAuth selected
nativeUpdateAllowed yes
nativeAuthSubtree "ou=pd,o=etptest,c=us"
nativeAuthSubtree "ou=na,o=etptest,c=us"
```

Once the LDAP server has been restarted, the LDAP client can request to use its digital certificate for authentication. Below is the OMVS example that KARIN would issue.

```
ldapsearch -h 127.0.0.1 \
  -S EXTERNAL -V 3 -Z -K KARIN_RACF_Keyring \
  -b "ou=pd,o=etptest,c=us" \
  "objectclass=*"

```

The LDAP client request must be to an LDAP version 3 server, therefore the LDAP client must indicate this with the -V 3 option. The -Z and -K options are the same ones that were used in the server authentication request (from the previous section). They indicate that SSL is desired and identify the LDAP client's keyring. The -S EXTERNAL option indicates that the LDAP client's digital certificate is to be used for authentication, instead of a userid and password combination. The output of this command is the list of the 'pd' organizational unit.

The DN that is authenticated in this case is the DN in the certificate. For example, since the authenticated DN is the one in the digital certificate, there is no way to authenticate with digital certificates and get access to the SDBM's data. The following is an example of what will happen if that is attempted.

```
ldapsearch -h 127.0.0.1 \
  -S EXTERNAL -V 3 -Z -K KARIN_RACF_Keyring \
  -b "profiletype=group,racfdb=p24" \
  "objectclass=*"

```

ldap_search: Insufficient access
ldap_search: additional info: R000137 'CN=KARIN,T=MASTER LDAP USER,OU=ZSERIES FT
SS,O=GI TEAM,L=NORTH COLD,ST=MINNEAPOLIS,C=US' is not a valid RACF DN for bind.
Check that the syntax is correct and that it is a DN for a RACF user.

Overview of security on Linux

Although an entire book could easily be devoted to the subject of security, in this chapter we provide only an overview of the basics of security on Linux.

And while you can obtain very powerful tools to increase your Linux system's security, it's important to keep in mind that you will not automatically arrive at a security solution that is 100% effective, regardless of the operating system and the software. Why? Because effective security is not achieved simply through the use of tools. Rather, it has to be carefully planned in advance by means of a thoughtful *security policy*. In this chapter we discuss tools and hints you can utilize to increase the security level of your system after you install Linux for zSeries and S/390.

The basic security techniques can be summarized as follows:

- ▶ Disable unneeded services
- ▶ Use Secure Shell for remote access
- ▶ Use shadow password utilities
- ▶ Use the Pluggable Authentication Module (PAM)
- ▶ Monitor security news and alerts
- ▶ Use hardening tools
- ▶ Integrate z/VM security

In the following sections, we describe these points in more detail.

9.1 Disable unneeded services

Many network requests are handled by either the Internet Daemon (inetd), or the Extended Internet Daemon (xinetd). The Red Hat system uses xinetd as its default Internet daemon. All other distributions use inetd.

The inetd daemon listens on several ports on the network and starts the program which handles the connection. It has its own configuration file, where all services served by inetd are listed. However, some services are not handled by inetd; instead, they run their own daemon and listen on their agreed-upon port (an example is a Web server).

We recommend that you check all services running on the system *before* you connect Linux to a insecure network. All services that are not used should be disabled.

The services activated by default will differ, depending on which distribution is used. Table 9-1 lists which services are activated, by default, by each distribution.

Table 9-1 Active/Inactive services after default installation

Service	SuSE	Turbo	Red Hat	Caiman	Served by
telnet	active	- *	-	active	inetd/xinetd
ftp	active	- *	-	active	inetd/xinetd
smtp	active *	-	active	-	daemon **
time	active	-	-	-	inetd/xinetd
finger	active	-	-	-	inetd/xinetd
http	active	- *	-	-	daemon **
pop-3	active	-	-	-	inetd/xinetd
login	active	-	-	-	inetd/xinetd
shell	active	-	-	-	inetd/xinetd
printer	active	-	-	-	daemon **
nfs	active *	-	-	-	daemon **
ssh	-	active	active	-	daemon **
Notes: * Activation setting asked during installation ** Runs as a daemon, starts/stops via a separate script					

Inetd

Services can be easily deactivated by editing the file /etc/inetd.conf. To disable services in inetd, place a hash character (#) before each service you want to disable.

If you want to disable telnet, for example, change this line:

```
telnet stream tcp      nowait root    /usr/sbin/tcpd in.telnetd
```

to this:

```
#telnet stream tcp      nowait root    /usr/sbin/tcpd in.telnetd
```

After making any changes to the file `/etc/inetd.conf`, you then have to force `inetd` to reload this information. The most basic way to accomplish this is to determine the process ID (pid) of `inetd` via the `ps` command, and send it the hangup (HUP) signal via the `kill` command, as shown:

```
# ps aux | grep inetd
root      233  0.0  0.0 1188 516 ?        S   Jun05   0:00 /usr/sbin/inetd
root      8861 0.0  0.1 1364 528 pts/3    S   15:39   0:00 grep inetd
# kill -HUP 233
```

Several scripts in the distributions will do this for you, as shown in Table 9-2.

Table 9-2 Reloading `inetd` configuration file after any changes

Distribution	Command
SuSE	<code>rcinetd reload</code>
Turbo	<code>/etc/rc.d/init.d/inetd reload</code>
Caiman	<code>/etc/init.d/inetd reload</code>

xinetd

You can use a similar procedure to manipulate `xinetd`. To disable a service in `/etc/xinetd.conf`, add a line with the `disabled=<service>` value for disabling the respective service.

However, on Red Hat-based systems, besides the file `/etc/xinetd.conf`, there is a directory named `/etc/xinetd.d/`. Under that directory, there is a configuration file for each service. In the appropriate file (`/etc/xinetd.d/<service>`), you have to add a line such as:

```
disable=yes
```

Example 9-1 provides an example of the `xinetd.conf` configuration files.

Example 9-1 `xinetd.conf` - Extended Internet daemon

```
#
# xinetd.conf
#
# Copyright (c) 1998-99 SuSE GmbH Nuernberg, Germany.
#
defaults
{
    log_type          = FILE /var/log/xinetd.log
    log_on_success    = HOST EXIT DURATION
    log_on_failure    = HOST ATTEMPT RECORD
    # only_from       = localhost
    instances         = 2
#
# The specification of an interface is interesting, if we are on a firewall.
# For example, if you only want to provide services from an internal
# network interface, you may specify your internal interfaces IP-Address.
#
# interface = 127.0.0.1
#
# If you want to enable one of the following services, you only have to
# comment it out. After that, send SIGUSR1 to xinetd to force a
# reload of it's configuration
#
# disabled= ftp
#           disabled      = rstatd
# disabled= telnet
```

```

disabled= shell
# disabled= login
disabled= finger
disabled= pop3
disabled= comsat
disabled= ntalk
disabled= talk
disabled= discard
disabled= chargen
disabled= daytime
disabled= time
disabled= echo
disabled= daytime
disabled= time
disabled= smtp
disabled= ident
}

##
## Now the definitions of the different services
##

service telnet
{
    socket_type      = stream
    protocol         = tcp
    wait             = no
    user             = root
    server           = /usr/sbin/in.telnetd
    server_args= -n
# only_from= localhost
    no_access=
}
...

```

Send **xinetd** the hangup signal after any changes in the file `/etc/xinetd.conf`:

```

# ps aux | grep inetd
root      233  0.0  0.0 1188  516 ?        S   Jun05   0:00 /usr/sbin/xinetd
root      8861 0.0  0.1 1364   528 pts/3    S   15:39   0:00 grep xinetd
# kill -SIGUSR1 233

```

The insecurity of Telnet

Most TCP/IP network traffic is sent without any encryption. Every computer on a local area network (LAN) segment is able to read all TCP/IP packets coming through. Sensitive information like passwords can be easily *sniffed* and stored by a program.

The common way to connect to a remote machine is via telnet. Although the typed-in password is hidden on the screen during the login process, it is still transferred over the network in clear text. Example 9-2 shows the very insecure password of *password* being sniffed (some characters are doubled because of console echo):

Example 9-2 A sniffed telnet session

```

ÿÿÿÿÿÿÿÿ ÿÿ!ÿÿ"ÿÿ'ÿÿÿÿ#ÿÿÿÿÿ ÿÿ#ÿÿ'ÿÿÿÿÿÿ!ÿÿ"ÿÿÿÿ ÿÿÿÿ#ÿÿÿÿ'ÿÿÿÿÿÿÿÿ P ÿÿÿÿ
38400,38400ÿÿÿÿ# linux.local:0ÿÿÿÿ' PRINTERlp DISPLAYlinux.local:0ÿÿÿÿ
KVTÿÿÿÿÿÿÿÿWelcome to SuSE Linux 7.0 (s390) - Kernel 2.2.16 (0).

```

```
ÿÿ
vmlinux2 login: rroooott
```

Password: password

```
You have new mail in /var/spool/mail/root.
Last login: Mon May 21 14:37:58 from 9.12.2.171
Have a lot of fun...
# llaa
```

```
[Omtotal 28
drwx--x--x  3 root    root      4096 May 21 14:36 [01;34m.[0m
drwxr-xr-x 19 root    root      4096 May 18 16:01 [01;34m..[0m
-rw-----  1 root    root        476 May 21 15:59 [0m.bash_history[0m
-rw-r--r--  1 root    root      1124 Jun 20  2000 [0m.exrc[0m
-rw-r--r--  1 root    root     1301 Jun 20  2000 [0m.xinitrc[0m
drwxr-xr-x  2 root    root      4096 May 18 16:01 [01;34mbin[0m
# eexxiitt
logout
```

To avoid sending passwords over the network in clear text, several encryption methods can be used. Furthermore, TCP/IP version 6 (IPv6) is able to do encryption at the network layer, but this goes beyond the scope of this chapter.

9.2 Use Secure Shell for remote access

With Secure Shell (SSH), you can connect as telnet does, but the connection is encrypted. An SSH daemon must be running on the target host in order to enable establishing the connection.

SSH protects you from:

- ▶ IP spoofing
- ▶ DNS spoofing
- ▶ Interception of cleartext passwords

Authentication can be done in several ways:

- ▶ By password
- ▶ By key authentication
- ▶ Via host authentication
- ▶ Via Kerberos

Password

Authentication by password is the common way to log in via SSH.

Key authentication

Using key authentication, you can connect to a remote host without using a password. But using no password is a security risk because if someone obtains the private key, he will have access to the remote machine.

To avoid this scenario, you can protect your key with a *passphrase*. Users will then need both the key and the passphrase to connect to the remote machine.

Generating and using SSH keys

Before you can use the authentication method with keys, you have to generate the keys. The command **ssh-keygen** can be used by users to generate their SSH keys, or it can be used by the administrator to generate keys for the host.

```
$ ssh-keygen
Generating RSA keys: Key generation complete.
Enter file in which to save the key (/home/tot12/.ssh/identity):
Created directory '/home/tot12/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/tot12/.ssh/identity.
Your public key has been saved in /home/tot12/.ssh/identity.pub.
The key fingerprint is:
a1:44:ec:33:7e:c5:89:aa:e8:e1:1a:dd:8e:e0:4a:7d tot12@vmlinux2
$
```

The key generation routine generates two files in the `.ssh/` directory of the user:

identity The private key

identity.pub The public key

You should ensure that the file `identity` is read/writeable only by the user.

The directory structure would then look as follows:

```
$ ls -la .ssh/
drw----- 2 tot12 users 4096 Mai 21 09:14 .
drw----- 30 tot12 users 16384 Mai 29 13:50 ..
-rw----- 1 tot12 users 524 Mai 21 09:14 identity
-rw-r--r-- 1 tot12 users 328 Mai 21 09:14 identity.pub
```

To connect to another host without the usual password authentication, the file `identity.pub` has to be attached to the file `.ssh/authorized_keys` in the home directory of the remote user.

The directory structure of the remote `.ssh/` directory will be:

```
$ ls -la .ssh/
drw----- 2 user13 users 4096 Mai 21 09:14 .
drw----- 30 user13 users 16384 Mai 21 09:50 ..
-rw-r--r-- 1 user13 users 328 Mai 22 10:00 authorized_keys
-rw----- 1 user13 users 524 Mai 15 19:04 identity
-rw-r--r-- 1 user13 users 328 Mai 15 19:04 identity.pub
```

You can attach your own public key file. The following command will attach a public key file to your own `authorized_keys` file:

```
$ cat identity.pub >> .ssh/authorized_keys
```

Attention: Give only your *public* key to other users!

The first time you establish a connection to a remote host, you will be prompted to accept the fingerprint of the host key of the remote machine. This ensures that the remote host identifies itself to you, so you can be sure that the host isn't replaced by someone (known as a "man in the middle attack").

The remote host fingerprint is stored in the file `.ssh/known_hosts`. If the host changes because a new system install was performed, you have to delete the line which includes the fingerprint of the host. Otherwise, `ssh` will refuse to connect to the host, as shown:

```
$ ssh vmlinux1
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!    @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
Please contact your system administrator.
Add correct host key in /home/tot12/.ssh/known_hosts to get rid of this message.
RSA host key for vmlinux1 has changed and you have requested strict checking.
```

The first time you connect to a remote host, the host's fingerprint will be saved for future connections and verification:

```
$ ssh vmlinux1
The authenticity of host 'vmlinux1' can't be established.
RSA key fingerprint is 7e:53:7d:54:99:47:34:47:8d:8d:85:23:0a:f5:d2:f1.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'vmlinux1,9.12.6.98' (RSA) to the list of known hosts.
tot12@vmlinux1's password:
Have a lot of fun...
```

You can also connect to a different user on the remote host by using the `-l` parameter:

```
$ ssh -l poke vmlinux2
poke@vmlinux2's password:
Last login: Thu Jun  7 17:08:58 2001 from vmlinux1.itso.ibm.com
Have a lot of fun...
```

Example 9-3 shows the many options that `ssh` can take.

Example 9-3 `ssh` options

```
# ssh -h
Usage: ssh [options] host [command]
Options:
-l user      Log in using this user name.
-n          Redirect input from /dev/null.
-A          Enable authentication agent forwarding.
-a          Disable authentication agent forwarding.
-X          Enable X11 connection forwarding.
-x          Disable X11 connection forwarding.
-i file      Identity for RSA authentication (default: ~/.ssh/identity).
-t          Tty; allocate a tty even if command is given.
-T          Do not allocate a tty.
-v          Verbose; display verbose debugging messages.
            Multiple -v increases verbosity.
-V          Display version number only.
-P          Don't allocate a privileged port.
-q          Quiet; don't display any warning messages.
-f          Fork into background after authentication.
-e char     Set escape character; "none" = disable (default: ~).
-c cipher   Select encryption algorithm: "3des", "blowfish"
-p port     Connect to this port. Server must be on the same port.
-L listen-port:host:port Forward local port to remote address
-R listen-port:host:port Forward remote port to local address
            These cause ssh to listen for connections on a port, and
            forward them to the other side by connecting to host:port.
-C          Enable compression.
-N          Do not execute a shell or command.
-g          Allow remote hosts to connect to forwarded ports.
-4          Use IPv4 only.
```

- 6 Use IPv6 only.
 - 2 Force protocol version 2.
 - o 'option' Process the option as if it was read from a configuration file.
-

Use secure copy for copying files

It is common to use FTP to transfer files over the network. But with FTP, as with telnet, the password is transferred in clear text and can be sniffed by an attacker. Therefore we recommend that rather than using FTP, you use the secure copy command **scp** for file transfers because the connection will be encrypted and you can take advantage of the other authentication methods.

The following example shows how to copy the file `openssh.rpm` from `vmlinux2` to `vmlinux1`:

```
$ scp openssh.rpm vmlinux1:
tot12@vmlinux1's password:
openssh.rpm      100% |*****| 1264 KB  00:01
```

This is an example of copying in the reverse direction, from the remote host `vmlinux1` to the local machine:

```
$ scp vmlinux1:/tmp/openssh.rpm ./
tot12@vmlinux1's password:
openssh.rpm      100% |*****| 1264 KB  00:01
```

To copy a whole directory recursively, use the **-r** flag:

```
$ scp -r * vmlinux1:/tmp/
tot12@vmlinux1's password:
openssh.rpm      100% |*****| 1264 KB  00:01
archiv.tar.gz    100% |*****| 3874 KB  00:05
...
```

The connection is closed after the transfer has finished. Transfer with **scp** is somewhat slower than normal FTP because of the encryption.

Secure login (slogin)

To execute a command or program on a remote machine, use the **slogin** command. The syntax is:

```
slogin <host> <command>
```

Following is an example of running the command **uptime** on the remote host **vmlinux1**:

```
$ slogin vmlinux1 uptime
tot12@vmlinux1's password:
8:28pm up 1 day, 23 min, 2 users, load average: 0.00, 0.07, 0.10
```

SSH daemon configuration

Now let's look at the major settings of the **sshd** daemon. For proper operation of the **ssh** connection, several settings can be changed in the **ssh** configuration file, as shown in Example 9-4.

Example 9-4 /etc/ssh/sshd_config - ssh daemon configuration file

```
# This is ssh server systemwide configuration file.

Port 22
#Protocol 2,1
ListenAddress 0.0.0.0
#ListenAddress ::
HostKey /etc/ssh/ssh_host_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600

This enables root login if set to yes: (should be no)
PermitRootLogin no
#
# Don't read ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRSAAuthentication
#IgnoreUserKnownHosts yes
StrictModes yes
This enables you to run X-sessions over the encrypted connection
X11Forwarding yes
X11DisplayOffset 10
PrintMotd yes
KeepAlive yes

# Logging
SyslogFacility AUTH
LogLevel INFO
#obsoletes QuietMode and FascistLogging

This enables remote host authentication: (should be no)
RhostsAuthentication no
#
# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
RhostsRSAAuthentication no
#
Enables the key authentication when set to yes:
RSAAuthentication yes

# To disable tunneled clear text passwords, change to no here!
```

Enables password authentication if set to yes (should be no if you only want key authentication, but turn RSAAuthentication!!):

PasswordAuthentication yes

Allows empty passwords if set to yes: (should be no)

PermitEmptyPasswords no

Uncomment to disable s/key passwords

#SkeyAuthentication no

#KbdInteractiveAuthentication yes

To change Kerberos options

#KerberosAuthentication no

#KerberosOrLocalPasswd yes

#AFSTokenPassing no

#KerberosTicketCleanup no

Kerberos TGT Passing does only work with the AFS kaserver

#KerberosTgtPassing yes

CheckMail no

#UseLogin no

Note: We recommend that you turn off root login.

Differences between SSH protocol version 1 and version 2

There are two protocol versions of SSH: SSH1 and SSH2.

SSH1 has the following attributes:

- ▶ It uses the RSA algorithm, which is patented until September 2000.
- ▶ It is well-tested, and free.

SSH2 has the following attributes:

- ▶ It has been totally rewritten.
- ▶ It has improved privacy.
- ▶ It has a more restrictive license than SSH1; it is free for non-commercial use only.

SSH client/server sources

There are quite a few sources of SSH. All of the distributions include OpenSSH, which is on the Web at:

<http://www.openssh.org>

If you are using a Windows client, there are also quite a few SSH clients:

TeraTerm Pro is on the Web at:

<http://hp.vector.co.jp/authors/VA002416/teraterm.html>

Putty is on the Web at:

<http://www.chiark.greenend.org.uk/~sgtatham/putty>

SSH-win32 is on the Web at:

<http://guardian.htu.tuwien.ac.at/therapy/ssh>

9.3 Use shadow password utilities

Passwords and information about users are usually stored in the file `/etc/passwd`. Although the password is stored in an encrypted format, with some tools it's possible to *crack* the password if you have read access to this file.

The shadow password utility stores the encrypted password in the file `/etc/shadow`, separated from `/etc/passwd`, because some applications have to read the user/group ID of users, so `/etc/passwd` must be readable to all.

The access privileges to `/etc/passwd` are:

```
$ ls -l /etc/passwd
-rw-r--r-- 1 root root 3534 Jun 6 09:52 /etc/passwd
```

The access privileges to `/etc/shadow` do not allow world read, and only allow group read to members of the shadow group:

```
$ ls -l /etc/shadow
-rw-r----- 1 root shadow 2083 Jun 7 10:02 /etc/shadow
```

Therefore, a normal user cannot read the shadow password file, as shown:

```
$ head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
$ head -1 /etc/shadow
head: /etc/shadow: Permission denied
```

To avoid this, you should use shadow password utilities instead of normal password files. With this utility, the file `/etc/passwd` does not contain the encrypted password from the user, but it does contain all other necessary information, such as the user's user/group ID, shell, and so on. The encrypted passwords are stored in the file `/etc/shadow`, which is only readable to the user root. The `/etc/passwd` file has an entry with the following format:

```
username:password:UID:GID:name:home_directory:shell
```

Table 9-3 Description of `/etc/passwd`

Attribute	Description
username	The character login name of the user
Password	Password ('x', if shadow password is used)
UID	The integer user ID
GID	The integer group ID
Home directory	Full path to home directory
Default shell	The executable shell

Now we log in as root and observe the root entry in the shadow password file, `/etc/shadow`. The letter x in the second field of the `/etc/passwd` file informs the system that the root password information is stored in the shadow password file:

```
# head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
# head -1 /etc/shadow
root:U2TpkoRQ7d1o:11446:0:10000:::
```

The `/etc/shadow` file has many more attributes than `/etc/passwd`:

```
username:password:last:may:must:warn:expire:disable:reserved
```

Table 9-4 lists and describes these attributes:

Table 9-4 Description of /etc/shadow

Attribute	Description
username	User name (login)
password	Encoded password
last	Days since Jan 1, 1970 password was last changed
may	Days before password may be changed
must	Days after which password must be changed
warn	Days before password is to expire that user is warned
expire	Days after password expires that account is disabled
disable	Days since Jan 1, 1970 that account is disabled
reserved	Reserved for future use

The first two characters of the encrypted password are called the *SALT* value. Following are shadow utility commands:

- useradd** Adds a user to the system; can only be run by root.
- userdel** Deletes a user from system; can only be run by root.
- usermod** Shows/modifies user-related data like name, UID, and so on; can only be run by root.
- passwd** Changes the password of a user; if run by root, it can change the password of any user.
- chage** Changes the user password expiry information; can only be run by root.

Following are examples of tasks that can be accomplished via the shadow password utilities.

- To set the validity of a user ID for a short period of time, add the **-e** flag to the **useradd** command:

```
# useradd -m -e 2001-06-15 user23
```

The account will become invalid on June 15, 2001.
- To force the user to change his password regularly, add the **-M** flag to the **chage** command:

```
# chage -M 7 user23
```

The user will have to change his password every 7 days (of course, he will only be able to change it once, because after that it will be expired).
- Use a password generator to avoid weak passwords:

```
# head --bytes=16 /dev/urandom | md5sum  
230ca11cfe3acd5cc6645121e3db52c5 -
```

Note, however, that overly complex passwords are often written down, which adds a new security exposure.
- Force the user to change his password on the first login by adding the **-m** flag to the **useradd** command:

```
# useradd -m user23  
# passwd user23  
...  
# chage -m 0 -M 99999 -d 0 user23
```

In the preceding example, we added user23 to the system and set the password. We then changed the age at which he has to change the password on his first login. The initial password can be used for the first login, but then it must be reset; see the **change** man page for details.

When the user logs into the system, he will be prompted to use a new password, as follows:

```
$ ssh vmlinux2 -l user23
The authenticity of host 'vmlinux2 (9.12.6.99)' can't be established.
RSA1 key fingerprint is 52:ad:c2:36:4c:08:78:0b:34:a7:54:ae:64:9f:a8:69.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'vmlinux2,9.12.6.99' to list of known hosts.
user23@vmlinux2's password: <-- assigned password
Password changing requested. Choose a new password.
Warning: Password has expired, please change it now <-- user must change
...
```

9.4 Use the Pluggable Authentication Module (PAM)

The traditional way to authenticate users is by password; see Figure 9-1. Many programs write their own authentication code. Sometimes the code is well-written, and sometimes it is not. But whenever authentication code exists in many programs, it is duplicated. If you want to use another authentication method, you have to recompile *each* program.

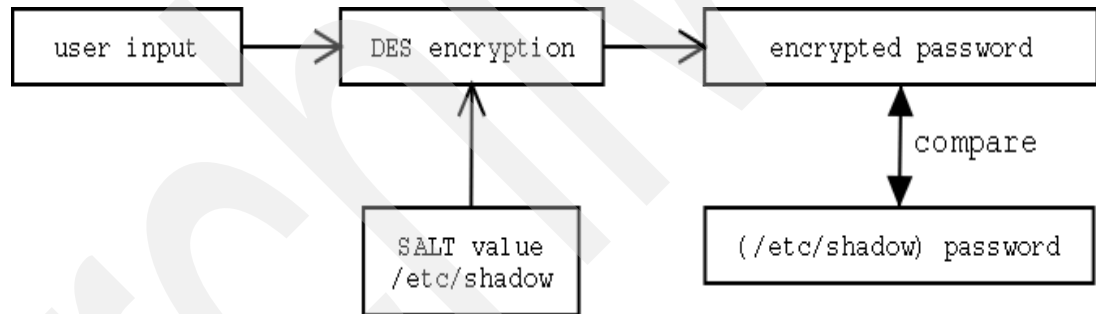


Figure 9-1 Traditional password authentication

In contrast, however, if the program is compiled with Pluggable Authentication Module (PAM) support, you can switch between several authentication methods without having to recompile each program. This uses the power of Linux's share libraries.

The advantages of using PAM lie in its *transparent authentication* method. In a heterogeneous network, for example, the system administrator wants to authenticate the users against an NT box, or an NIS service. If there is a PAM module for such authentication, the administrator can perform the authentication easily, without having to maintain several different databases where users are listed for different systems—one authentication method can be used for different systems. And the authentication can be done with a magnetic card, retina scan, or whatever.

Figure 9-2 on page 256 illustrates a PAM authentication block; PAM is installed by default on all distributions.

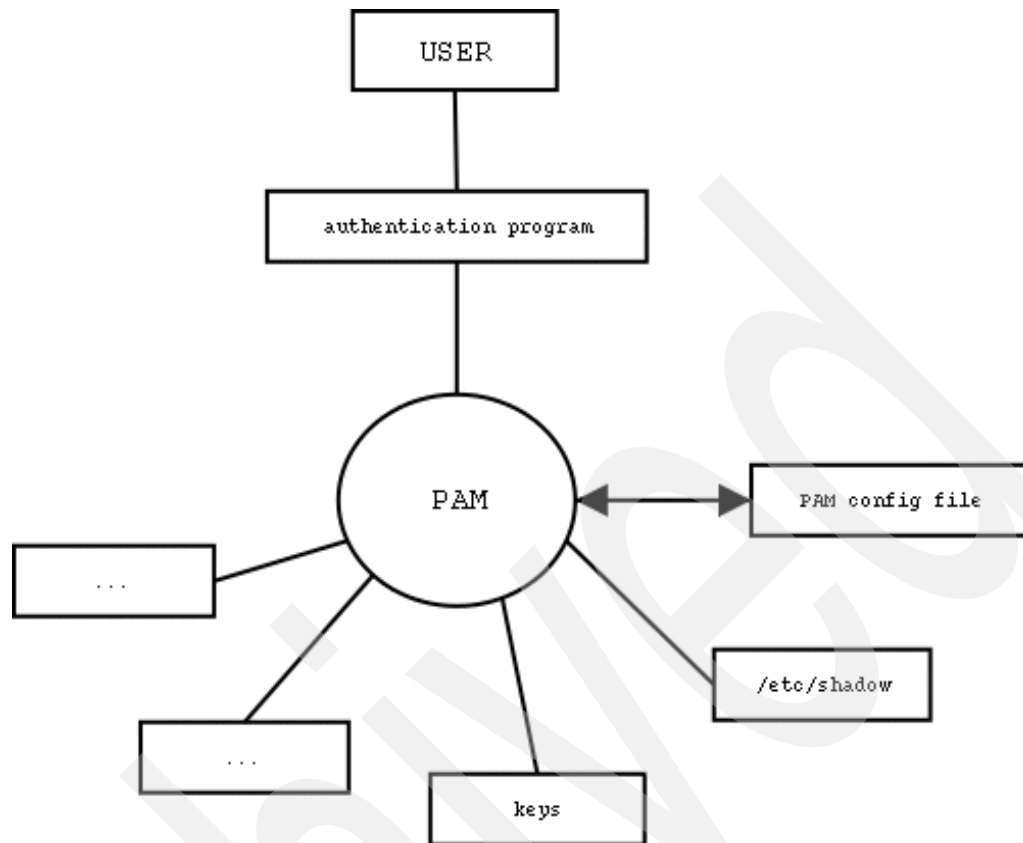


Figure 9-2 PAM authentication block diagram

Configuring PAM

The services are listed in the directory `/etc/pam.d/`. Each service has its own configuration file, which means that every service listed in the directory can have different security policies; see Example 9-5.

Example 9-5 `/etc/pam.d/login`

```

#%PAM-1.0
#type    control    module-path          module-arguments
auth      requisite /lib/security/pam_unix.so      nullok #set_secrcp
#auth     required  /lib/security/pam_securetty.so
auth      required  /lib/security/pam_nologin.so
#auth     required  /lib/security/pam_homecheck.so
auth      required  /lib/security/pam_env.so
auth      required  /lib/security/pam_mail.so
account   required  /lib/security/pam_unix.so
password  required  /lib/security/pam_pwcheck.so      nullok
password  required  /lib/security/pam_unix.so      nullok use_first_pass use_authtok
session   required  /lib/security/pam_unix.so          none # debug or trace
session   required  /lib/security/pam_limits.so
  
```

The configuration file contains four columns: type, control, module-path, and module-arguments. Entries in these columns have the following meanings:

type	Tells PAM what type of authentication is used for the module; can be one of the following values:
auth	Checks user ID and password.
account	Determines if the user has access to the service.
password	What to do if the user changes his authentication (password).
session	What to do after/before successful authentication.
control	Tells PAM what to do if the authentication by the module fails; can be one of the following values:
requisite	If authentication with this module fails, the authentication will be denied.
required	If authentication fails, PAM still calls the other modules before denying authentication.
sufficient	If authentication is successful, PAM grants authentication even if a previous module failed.
optional	Whether the module succeeds or fails is only significant if it is the only module of its type for this service.
module-path	The path to the PAM module.
module-arguments	The arguments for the PAM module.

Following are examples using the PAM-module pam_unix. This module can be used in the following management groups (with examples):

► **auth**

```
auth requisite /lib/security/pam_unix.sonullok
```

The module gets the user ID and password. Disable the NULL password by removing the nullok argument.

► **account**

```
account required /lib/security/pam_unix.so
```

The module gets information such as password expire, last_change, and so on from /etc/shadow.

► **password**

```
password required /lib/security/pam_unix.sonullok use_first_pass
```

This is used for updating/changing passwords. Disable the nullok argument so it will not allow NULL passwords.

► **session**

```
session required /lib/security/pam_unix.so
```

This logs the username to the syslog.

Table 9-5 lists which module comes with each distribution, and also provides a brief description of each module.

Table 9-5 Available modules: /lib/security/

Module	Description	SuSE	Turbo	Red Hat	Caiman	Think Blue
pam_access	Provides logdaemon-style login access control	yes	yes	yes	yes	yes
pam_chroot	Puts user into chroot environment	-	-	yes	-	yes
pam_console	Gives users access at the physical console	-	yes	yes	-	yes
pam_cracklib	Checks strength for passwords	yes	yes	yes	-	yes
pam_deny	Denies everything	yes	yes	yes	yes	yes
pam_env	Allows you to set/unset environment variables	yes	yes	yes	yes	yes
pam_filter	Applies filter rules to input/output stream	yes	yes	yes	yes	yes
pam_ftp	Provides anonymous ftp	yes	yes	yes	yes	yes
pam_group	Provides group settings	yes	yes	yes	yes	yes
pam_issue	Displays the issue file /etc/issue	yes	yes	yes	yes	yes
pam_krb5	Performs Kerberos verification	-	-	yes	-	yes
pam_krb5afs	Kerberos	-	-	yes	-	yes
pam_lastlog	Writes last login into /var/log/lastlog	yes	yes	yes	yes	yes
pam_ldap	Authentication with LDAP	*	-	yes	-	yes
pam_limits	Sets some system limits to the user	yes	yes	yes	yes	yes
pam_listfile	Deny or allow services based on an arbitrary file	yes	yes	yes	yes	yes
pam_localuser	User must be listed in /etc/passwd	-	-	yes	-	yes
pam_mail	Indicates if user has mail	yes	yes	yes	yes	yes
pam_mkhomedir	Creates home directory on the fly	-	yes	yes	yes	yes
pam_motd	Displays the motd file /etc/motd	-	yes	yes	-	yes
pam_nologin	Prevent from login if file /etc/nologin exists	yes	yes	yes	yes	yes
pam_permit	Permits without confirmation	yes	yes	yes	yes	yes
pam_pwdb	Give status of user account	yes	yes	yes	-	yes
pam_rhosts_auth	Authenticate via rhosts	yes	yes	yes	yes	yes
pam_rootok	Rootaccess to a service without a password	yes	yes	yes	yes	yes
pam_securetty	Checks in file /etc/securetty if the TTY is valid for login	yes	yes	yes	yes	yes
pam_shells	Checks in file /etc/shells if shell is valid	yes	yes	yes	yes	yes
pam_stack	Propagates login and password to underlying module	-	-	yes	-	yes
pam_stress	Stress test applications	yes	yes	yes	yes	yes

Module	Description	SuSE	Turbo	Red Hat	Caiman	Think Blue
pam_tally	Counts login attempts and can deny access	yes	yes	yes	yes	yes
pam_time	Gives users restricted permissions depending on time and date to login	yes	yes	yes	yes	yes
pam_userdb	Checks username/password against Berkeley DB	yes	yes	yes	yes	yes
pam_unix	Checks passwords and more	yes	yes	yes	yes	yes
pam_warn	Logs information about a proposed authentication to update a password	yes	yes	yes	yes	yes
pam_wheel	Root access if user is member of the group wheel	yes	yes	yes	yes	yes
pam_xauth	Forwards xauth keys	-	yes	yes	-	yes
pam_homecheck	Checks if homedirectory of user exists	yes	-	-	yes	-
pam_radius	Authenticate via radius	yes	-	-	-	-
pam_pwcheck	Additional checks upon password changes	yes	-	-	-	-
pam_smb_auth	Authenticate against NT Server	*				
* Not in default installation - extra package						

The file `/etc/pam.d/others` is for all unknown services that are not handled by a PAM configuration file. You can edit the file to make the default policy stronger, but be careful with the module `pam_deny`, because it denies every attempt to login, even the user ID and the password suits.

9.5 Monitor security news and alerts

Software on your system might have bugs or vulnerabilities that are not known at a certain point in time. When such vulnerabilities are discovered, it is important that they be fixed as soon as possible, because they can quickly be exploited by crackers.

Many Web sites exist where you can check for the latest news on vulnerabilities in software you plan to use, or are currently using. The following are useful security-related sites:

<http://securitytracker.com>
<http://www.securityportal.com>
<http://www.securitysearch.net>
<http://www.securityfocus.com>
<http://www.lwn.net>

Monitor your own log files

It's important for you to know where your log files are being written so you can monitor them. For example, nearly all applications write to log files in the directory `/var/log/`. By default, warning messages are sent to the file `/var/log/warn`, and all other messages are written to the file `/var/log/messages`.

However, many applications write to other directories and this may vary by distribution. For example, Samba typically writes log files to the directory `/var/log/samba`.

9.6 Use hardening tools

The kernel itself is able to handle incoming and outgoing network traffic in several ways. For each TCP or UDP packet, you can specify what to do with it, and although a discussion of how to set up a complete firewall is outside the scope of this redbook, we want to give an overview of how the utilities can be used.

IPchains

IPchains is used to set up, maintain, and inspect the IP firewall rules in the Linux kernel. Following are some of the basic IPchains commands:

ipchains -L	List all chains and the applied rules
ipchains -A <chain> <rule>	Add <rule> to chain named <chain>
ipchains -D <chain> <rule-number>	Delete <rule-number> in chain named <chain>
ipchains -P <chain> <policy>	Set default policy for chain named <chain> to <policy>

IPchains is used not only to act as a firewall between several networks, it can also be used to restrict network access to your machine. For example, if you want to give only dedicated hosts access to services such as http, use the following command:

```
# ipchains -A input -p tcp -s vmlinux1.itso.ibm.com/32 -d vmlinux2.itso.ibm.com/32 80
# ipchains -L
Chain input (policy DENY):
target    prot opt      source                destination            ports
-         tcp  -----  vmlinux1.itso.ibm.com vmlinux2.itso.ibm.com  any ->    http
Chain forward (policy DENY):
Chain output (policy ACCEPT):
```

Note: Take care with your configuration, because it's possible to cut yourself off the host with an ipchains rule! If this happens, networking will effectively be down and you'll only be able to get to the host to fix the problem via the local console under VM or the HMC.

If you want to use ip-forwarding in order to use ipchains with several networking devices, you have to enable it in the kernel. (We recommend you only enable ip-forwarding if you use ipchains to control network traffic and access, especially if you have more than one networking device.)

```
$ echo 1 > /proc/sys/net/ipv4/ip_forward
```

IPtables is the replacement for IPchains in the 2.4 kernel, but is not discussed further in this redbook.

Tripwire

If attackers manage to infiltrate a system, it is possible that they could install *trapdoors* or other utilities for sniffing passwords. This may not only be difficult to detect in your system, but it may also be difficult to determine *how* your system has been compromised.

One safe approach is to totally reinstall the operating system, but again, you may not even know your system is compromised. Because of this exposure, tools like **tripwire** have been developed to maintain an overview of all files and their modification in the system.

Tripwire is able to detect the following file/directory modifications:

- ▶ Added files/directories
- ▶ Deleted files/directories
- ▶ Changed files/directories

The latest version of Tripwire is available on the Web at:

<http://www.tripwire.org>

In the SuSE distribution that comes with Tripwire version 1.2, the files are installed in directory `/var/adm/tripwire/`. However, it is not installed with all installations. If you don't have tripwire installed, look for the rpm on CD1 under the `sec` directory, as shown:

```
# file /suse/cd1/suse/sec1/tripwire.rpm
/suse/cd1/suse/sec1/tripwire.rpm: RPM v3 bin tripwire-1.2-262
```

The `tw.config` file is an important configuration file. In it, you add all directories and files that you want tripwire to monitor. You can also exclude files from a directory; as follows:

```
#
# Tripwire config-file
#
/ R
!/proc
!/var
!/root
/root/bin
!/dev
!/tmp
!/etc/mtab
!/etc/ntp.drift
!/etc/ld.so.cache
!/etc/snmpd.agentinfo
!/etc/ssh_random_seed
!/etc/mail/sendmail.st
```

Port scanning detection

An attacker can figure out what services are running on the target host by scanning a server's ports. That's why we recommend that you disable unnecessary services; the fewer "doors" on your system, the harder it is for a cracker to get in.

nmap

Most port scanners can guess the operating system, and the attacker could have information about vulnerable services on the host. Some tools are able to recognize if a host is scanned by port scanners. Example 9-6 shows a host being scanned with the **nmap** tool.

Example 9-6 Scanning a host with nmap

```
# nmap -O vmlinux2
```

```
Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/ )
```

```
Interesting ports on vmlinux2.itso.ibm.com (9.12.6.99):
```

```
(The 1506 ports scanned but not shown below are in state: closed)
```

Port	State	Service
21/tcp	open	ftp
22/tcp	open	ssh
23/tcp	open	telnet
25/tcp	open	smtp
37/tcp	open	time
79/tcp	open	finger
80/tcp	open	http
110/tcp	open	pop-3
111/tcp	open	sunrpc
513/tcp	open	login
514/tcp	open	shell
515/tcp	open	printer
789/tcp	open	unknown
901/tcp	open	samba-swat

```
1507/tcp  open      simplex
1533/tcp  open      virtual-places
2049/tcp  open      nfs
```

```
TCP Sequence Prediction: Class=random positive increments
                        Difficulty=5525585 (Good luck!)
```

```
Remote operating system guess: Linux 2.1.122 - 2.2.14
```

```
Nmap run completed -- 1 IP address (1 host up) scanned in 1 second
vmlinux1:~ #
```

scanlog

The **scanlog** daemon can be used to recognize a scan if hosts request more than a specific amount of ports per second. In this example, we have **scanlog** running and the scan is detected and written to the file `/var/log/messages`:

```
# grep scanlogd /var/log/messages
```

```
Jun  6 16:35:24 vmlinux2 scanlogd: 9.12.6.98 to 9.12.6.99 ports 36886, 19, 258, 348, 424,
205, 406, ..., f??pauxy, TOS 00 @16:35:23
```

9.7 Integrate VM and z/VM security

In this section we describe the security available in VM and z/VM. VM guest security can be used in addition to Linux security.

Operating system failures that occur in virtual machines do not normally affect the VM operating system running on the real processor. If the error is isolated to a virtual machine, only that virtual machine fails, and the user can re-IPL without affecting the testing and production work running in other virtual machines.

The Control Program (CP) common to VM and z/VM has integrity such that programs running in a virtual machine are unable to do the following:

- ▶ Circumvent or disable the CP real or auxiliary storage protection.
- ▶ Access a resource protected by RACF (resources protected by RACF include virtual machines, minidisks, and terminals)
- ▶ Access a CP password-protected resource
- ▶ Obtain control in a real supervisor state, or with a privilege class authority or directory capabilities greater than those it was assigned
- ▶ Circumvent the system integrity of any guest operating system that itself has system integrity as the result of an operation by any VM CP facility

Following are more detailed explanations of these terms:

Real storage protection Refers to the isolation of one virtual machine from another. CP accomplishes this by hardware dynamic address translation, start interpretive-execution guest storage extent limitation, and the Set Address Limit facility.

Auxiliary storage protection

Refers to the disk extent isolation implemented for minidisks/virtual disks through channel program translation.

Password-protected resource

Refers to a resource protected by CP logon passwords and minidisk passwords.

Guest operating system Refers to a CP (such as Linux for S/390 and zSeries) that operates under the CP.

Directory capabilities Refers to those directory options that control functions intended to be restricted by specific assignment, such as those that permit system integrity controls to be bypassed, or those not intended to be generally granted to users.

VM includes several facilities to enhance or improve the security and integrity of the system:

- ▶ Each guest and CMS user runs in a unique virtual machine definition which, in combination with hardware features, prohibits one user from accessing another's data in storage (unless specifically allowed through shared segments, communication vehicles such as IUCV and APPC/VM, or ESA/XC data sharing services).
- ▶ VM, in combination with hardware features, provides protection against channel programs accessing another user's virtual addresses.
- ▶ A password facility provides minidisk security to control both read-only and read-write access.
- ▶ Both user ID and password checking are provided to minimize unauthorized system access.
- ▶ User class restructure provides customers with the ability to control access to commands and DIAGNOSE codes more precisely through customer-defined classes.
- ▶ Journaling is supported on VM. In addition, RACF provides customers with many of these facilities, as well as other security capabilities.

In addition to these features for VM, z/VM offers the following function:

- ▶ Directory control statements and system configuration file statements provide controls for certain POSIX-related functions, such as the ability to change another virtual machine's POSIX security values.

The TCP/IP feature for z/VM offers:

- Kerberos authentication service
- Secure Socket Layer (SSL) support

RACF under z/VM

The Resource Access Control Facility (RACF) licensed program is a strategic security facility that provides comprehensive security capabilities. RACF controls user access to the system, checks authorization for use of system resources, and audits the use of system resources.

In the z/VM environment, RACF verifies logon passwords and checks access to minidisks, data in spool files, and RSCS nodes. You can use RACF commands to audit security-relevant events and prevent users from entering the CP DIAL and MSG commands before they log on.

The events you can audit include:

- ▶ Any CP command or DIAGNOSE code (including privileged commands and DIAGNOSE codes)
- ▶ Creating, opening, and deleting spool files
- ▶ Dumping and loading spool files through the SPXTAPE and SPTAPE commands

- ▶ IUCV CONNECT and SEVER operations, and certain VMCF functions
- ▶ APPC/VM CONNECT and SEVER operations
- ▶ Creating and deleting logs

Notes about using SDBM with the LDAP Server

This appendix provides some additional information about SDBM, the z/OS LDAP server's access to RACF information, and describes what it is and what it is not. This material goes behind the SDBM and describe its functionality and usage.

When the LDAP server is set up to use the SDBM (the RACF interface) as the backend, there are a few basic items that must be understood.

1. The SDBM goes directly against the RACF database. It does not use a copy of the RACF database. It does not copy RACF data. When the LDAP client issues an **ldapadd** command to add a RACF userid, this will be translated by the LDAP server into an **ADDUSER** command and the userid will be added to the RACF database. Therefore, all access of the SDBM for updating or displaying of RACF data requires the client to have the appropriate RACF attribute of either RACF SPECIAL or AUDITOR.
2. Since the LDAP server is using the RACF database, the LDAP schema definitions must match the RACF database layout. The LDAP schema definitions are in either
 - a. The schema.IBM.ldif file
 - b. The slapd.racf.at and slapd.racf.oc files
 - c. The schema.IBM.at and schema.IBM.oc files

This is based upon what the installation is using to define the LDAP server.

3. Because the LDAP server is using a zSeries security feature (RACF) for its SDBM directory, this functionality is only valid for the z/OS LDAP server. A distributed (a non-z/OS) LDAP server cannot directly use an SDBM backend; that is, an LDAP server running on AIX cannot have as its directory the SDBM or RACF database. Also, the z/OS LDAP server that is using the SDBM must be set up on the same z/OS image as the RACF database that it is using; that is, when the z/OS LDAP server is configured to use the SDBM, there is no way to point to a RACF database that is not the currently active RACF database on the z/OS system where the LDAP server is running. If the z/OS system(s) are configured in a parallel sysplex, the RACF database can be accessed by multiple LDAP servers within that parallel sysplex as long as the same RACF database is the active RACF database on all images.

4. When configured with the SDBM, the z/OS LDAP server uses the active RACF database. For this reason, there are several common LDAP functions and features that are not allowed for the SDBM part of the LDAP tree.
 - a. The SDBM data cannot be replicated to, or from, other LDAP servers. The SDBM environment will not allow RACF data to be replicated from the z/OS LDAP server to a distributed LDAP server. Nor can a distributed LDAP server replicate its data into the z/OS SDBM LDAP environment. This includes password changes, adding or deleting users or groups, etc. Some of this can be implied by an installation by writing exit code, but care must be taken in these cases. For example, if passwords are to be kept synchronized with a distributed LDAP environment, then the synchronization function must be written into the RACF exits (most likely the ICHPWX01 exit). In this fashion, when a RACF password changes, the change can be passed (probably via LDAP client code) to the distributed LDAP server. The distributed environment will probably have to do a similar synchronization function when the password change is initiated on the distributed system. If the LDAP servers that are being synchronized are both z/OS LDAP servers using SDBM (separate RACF databases), then set up RRSF between the RACF environments.
 - b. Another common feature that is not allowed with the SDBM is the “unauthenticated search” capabilities that several LDAP servers allow. An example of this is when an application client provides a partial user name and a password, and the application searches the directory to find a match (or bind/authenticate) for the DN and password. This is not allowed in RACF and is not allowed with z/OS LDAP server using SDBM.
 - c. The required DN to access the SDBM portion of the LDAP namespace demands some attention. The RACF DN has two naming attributes that are required. These are RACFID= and PROFILETYPE=. These are usually in place of the more commonly used CN= part of the DN. Some applications cannot handle this SDBM format of the DN. If the applications are not flexible in their handling of their formatting of the DN in their requests, then other mechanisms, including writing code, are needed. One solution that might work is the use of *native authentication*.

Note: Native authentication is mentioned here as an example. There is no requirement for SDBM when using native authentication. Native authentication's implementation is contained within the TDBM and can be used with any SAF-enabled z/OS security product. The implementation of native authentication is discussed in Chapter 5, “IBM Host On-Demand Version 6” on page 143, and in *z/OS Security Server LDAP Administration and Use*, SC24-5923. A white paper explaining native authentication can be found at http://www.ibm.com/servers/esdd/articles/zos_ldap/index.html

5. It must be remembered that the z/OS LDAP server, like any other LDAP server, is only an LDAP server. It is not the answer to single sign-on, nor the front end to the universal enterprise administration tool, nor the best mouse trap ever invented. LDAP servers are interfaces into distributed directories. This is also true of the z/OS LDAP server using SDBM. While RACF is viewed as the most secure user registry by many people, and the z/OS LDAP server using SDBM provides distributed access to this information, it is really the application's ability to use the LDAP features that adds the flexibility to enterprise

security. For example, in Figure A-1, it is the capabilities of the *applications* using LDAP to authenticate users that provides the single user security registry environment.

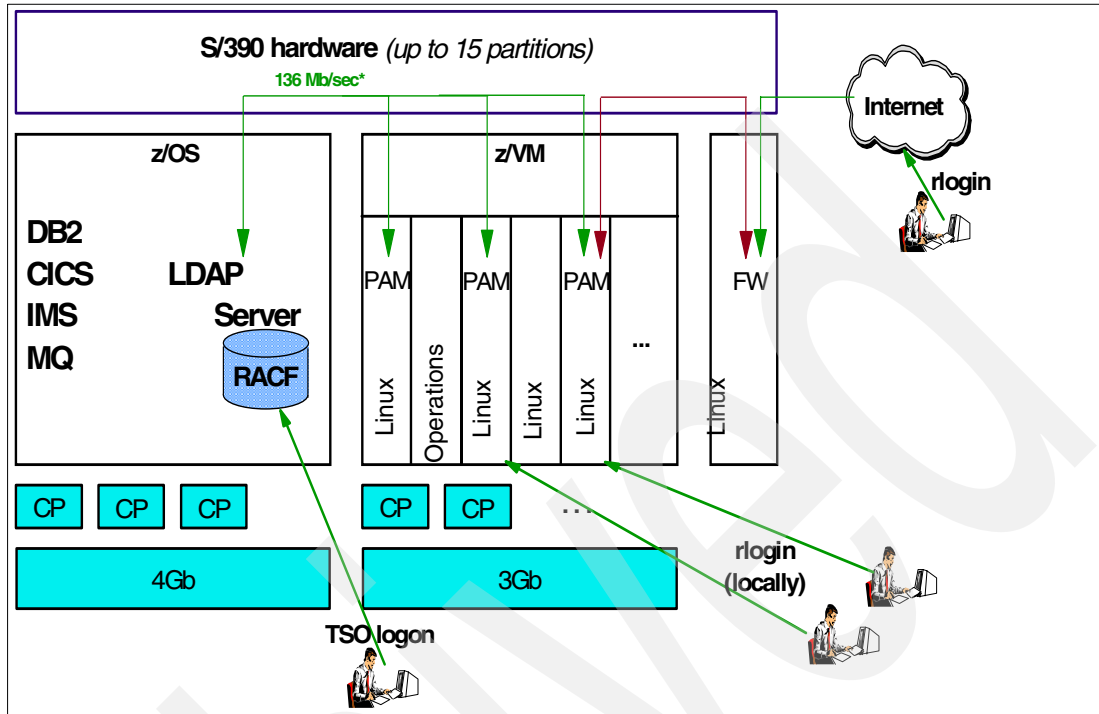


Figure A-1

LDAP provides the infrastructure, and the applications (HTTP server using either `%%CLIENT%%` or the LDAP server, TSO, Linux with PAM, etc.) use this tool. But it must be remembered, that the LDAP server only does the authentication of the user identity and does not build the user's security infrastructure required for authorization. In z/OS terms, this means that the LDAP server will check that the user/password combination or the digital certificate is valid, but the LDAP server will not build the ACEE (the z/OS security control block). That would be the responsibility of the application or system infrastructure.

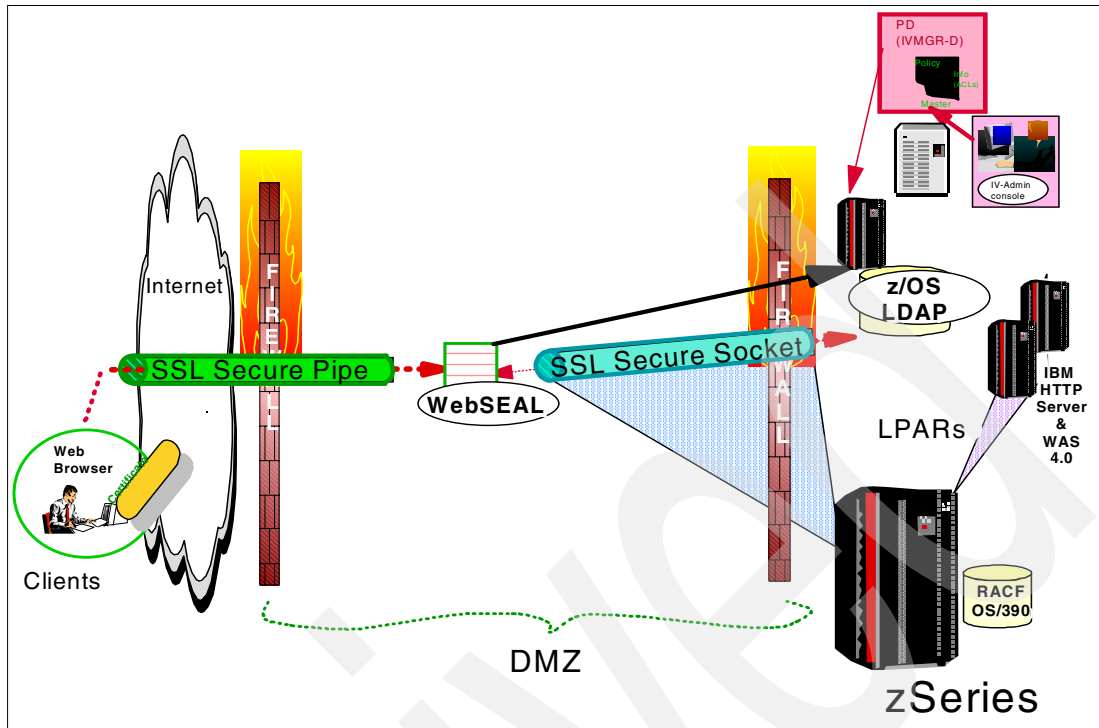


Figure A-2

In the example illustrated in Figure A-2, the user is authenticated by the LDAP server, which is called by the PD component, WebSeal, as the user enters the secure Web environment. As this authenticated identity is passed back to the WebSphere Application Server application, it is the responsibility of the application (in this example, either the Java servlet or EJB) to build and use the appropriate ACEE.

6. There are two basic usage scenarios for the SDBM backend of the z/OS LDAP server. The first is the BIND (authentication) functions of the LDAP server. The second function is access and update of user, group, and connect information within the RACF database.

Note: RACF connect information was introduced in OS/390 with APAR OW50971. This has been integrated into z/OS.

The BIND process is the LDAP server's method of authenticating the requestor. When the requestor binds to the LDAP server with a RACF userid and password (for example, RACFID=IBMUUSER,PROFILETYPE=USER,RACFDB=LOCAL), the LDAP server will use SAF services to ask RACF to validate the userid and password. This has several implications:

- a. Any security product that supports the SAF interface can be used by the LDAP server for this BIND process.
- b. The authenticated user DN can be used in the ACLs to control access to information stored within a TDBM. This function can be used by any security product that supports the SAF interface.
- c. If the DN is a RACF userid, then its associated RACF group DNs can be used in the ACLs to control access to information stored within a TDBM-backed tree. If this is not being used, then consider setting extendedgroupsearch to **No** to avoid performance considerations.

- d. The access to RACF data is protected and controlled by RACF. To issue RACF commands or to display RACF data, the authenticated DN must have either the RACF SPECIAL or AUDITOR attribute. The access to RACF data requires the SDBM to be configured in the slapd.conf file. The RACF data that can be accessed is user, group, and connect information. RACF information on general resources cannot be accessed. The `r_admin()` API is used to access the appropriate RACF information. This API is explained in detail in *RACF Callable Services Manual*, SA22-7691.
7. When attempting to access RACF data with the z/OS LDAP server, the strength and security of the RACF environment is not weakened. The z/OS LDAP server using SDBM does provide distributed access to RACF data, but it must also be remembered that the SDBM code itself has self-checking code to ensure that it is talking to the correct RACF database and that nobody is 'spoofing' either RACF or the SDBM code. Also, the LDAP client, the person/application issuing the RACF request, must bind to the z/OS LDAP server with a valid RACF DN that is verified by RACF (similar to them logging on to TSO). And the userid is checked to make sure that they have access to the RACF data, in much the same fashion that would happen if they were issuing the appropriate RACF command from TSO - that is, if it is an `ldapadd` to add a RACF user, then the requester must be a RACF GROUP SPECIAL or SPECIAL to complete the add successfully. Also if the request is going across a network, then the z/OS LDAP server does support SSL and it is strongly recommended that SSL support be set up and used for distributed access of RACF data.

Archived

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 272.

- ▶ *RACF Version 2 Release 2 Installation and Implementation Guide*, SG24-4580
- ▶ *IBM Host Access Client Package*, SG24-6182
- ▶ *IBM Communications Server for OS/390 TCP/IP Implementation Guide*, SG24-5227

Other resources

These publications are also relevant as further information sources:

- ▶ *OS/390 SecureWay Security Server Network Authentication and Privacy Service Administration*, SC24-5896
- ▶ *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776
- ▶ *DB2 UDB for OS/390 and z/OS: Application Programming and SQL Guide*, SC26-9933
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization*, GA22-7834
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Messages and Diagnosis*, GA22-7837
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Operations and Administration*, SA22-7835
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Assembling Java 2 Platform, Enterprise Edition (J2EE) Applications*, SA22-7836
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Assembling CORBA Applications*, SA22-7848
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: System Management User Interface*, SA22-7838
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: Migration*, GA22-7860
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390: System Management Scripting API*, SA22-7839
- ▶ *Building Business Solutions with WebSphere*, SC09-4432
- ▶ *Getting Started with WebSphere Application Server*, SC09-4581
- ▶ *Writing Enterprise Java Beans in WebSphere*, SC09-4431
- ▶ *WebSphere Application Server V4.0.1 for z/OS and OS/390 Program Directory*, GI10-0680
- ▶ *OS/390 V2R10.0 SecureWay Security Server RACF Macros and Interfaces*, SC28-1914

- ▶ *OS/390 V2R10.0 SecureWay Security Server RACF Security Administrator's Guide*, SC28-1915
- ▶ *z/OS V1R2.0 SecureWay Security Server LDAP Client Programming*, SC24-5924
- ▶ *z/OS V1R2.0 SecureWay Security Server LDAP Server Administration and Use*, SC24-5923
- ▶ *DB2 UDB for OS/390 and z/OS: ODBC Guide and Reference*, SC26-9941
- ▶ *WebSphere Application Server for OS/390: Application Server Planning, Installing, and Using V1.2*, GC34-4757
- ▶ *z/OS V1R3.0 Security Server RACF Callable Services*, SA22-7691

Referenced Web sites

These Web sites are also relevant as further information sources:

- ▶ For additional WebSphere for z/OS tools and supplements, go to:
http://www.ibm.com/software/webservers/appserv/zos_os390
- ▶ For information about the IBM Developer Kit for OS/390 Java 2 Technology Edition product: <http://www.s390.ibm.com/java>
- ▶ There are quite a few sources of SSH. All of the distributions include OpenSSH, which is on the Web at:
<http://www.openssh.org>
- ▶ The following are useful security-related sites:
<http://securitytracker.com>
<http://www.securityportal.com>
<http://www.securitysearch.net>
<http://www.securityfocus.com>
<http://www.lwn.net>

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Symbols

\$PSS.WD\$ 172–173
\$USR.ID\$ 170, 173
(TGT 87
/etc/passwd
 fields 253
/etc/rc 98
/etc/shadow
 fields 253
/var/skrb/creds 98

Numerics

390 KDC
 defining to Windows 2000 workstation 119

A

Abstract Syntax Notation 1 87
Add the certificate to the keyring 158
ADDUSER 20
AIM 20
ALTUSER 20
APAR
 OW31933 48
 OW38799 79
 OW39128 79
 OW42092 80
 OW45211 48, 50
 OW45212 48, 50
application ID 169
Application Identity Mapping 20
application name 157
Application servers tracing 216
ASN.1 87
Auditing 116
authentication server 86, 88
Authentication service exchange 87
authenticator 88, 90
Automatic local principal name mapping 106

B

base64cert 52
BLKUPD 31–32
BPX.DAEMON 16
BPX.SERVER 16
BXP014I 19
browsertype 52

C

cacert.der 50
cache files 98
Caching 4
CAGETCERT.REXX 58

CAMAIN.REXX 53
Cannot log on to the Administrator (LOG0001 error) 149
CAREQ.REXX 56
CARETRIEVE.REXX 58
CATMPL.REXX 55
CDSA 34
Certificate Authority 61
chage 254
chmod 98
chmount 79
chown 80
ck_priv 79
client 4
client authentication 162, 168, 239
client certificate 153, 162, 168
 setting up 162
client/server authentication exchange 90
Common Data Security Architecture 34
Configuring LDAP server for SDBM 131
Configuring LDAP server for TDBM 132
cron 98
Cryptographic Service Provider 55
CustomizedCAs.class file, creating 161

D

DB2 85
DB2 and LDAP
 guidelines for use 182
DB2 Connect 85
DB2 Connect Version 7.1 setup 119
DB2 jobs
 executing 225
DB2 Version 7 usage of Kerberos 119
debugging/tracing
 tips 215
Denial-of-service 92
Deployment Wizard 153–154
DES 86, 157
digital certificate 36, 38–39, 42, 45
DNS 4
DNS Service record
 defining 117
DNS TXT record
 defining 117
DoS 92

E

ELF, *See* express logon
Enabling native authentication 135
envvar 97
EUVF06014E 109
express logon 152–154, 157, 161–162, 168–169, 173
 macro 153, 157, 161, 168–170, 172–173
 TN3270 server 153, 168, 170, 172

F

finger 244
Foreign Kerberos principals 99
ftp 244

G

GID 20

H

hage 254
hardening tools 260
HFS for Kerberos cache files
 setting up 98
http 244

I

ICH01019I 80
ICH21035I 80
ICH420I 19
ICHRCX02 16
iecert 52
ikeyman 73
Implementing password encryption with z/OS LDAP 228
Import a Certificate 46
Incorrect password 150
inetd 244
Input file description 220
Install Windows 2000 DNS server 117
Installation dialog 184
Installing the LDAP server SDBM backend 130
Installing the WebSphere V4.0.1 runtime 180
Installing WebSphere Application Server 4.0.1 180
Internal Reorganization of Aliases 21
Internet Explorer 39, 48
inter-realm keys 91
Inter-realm operation 91
Introduction to Kerberos 86
IRR.DIGTCERT.ADD 70
IRR.DIGTCERT.GENCERT 70
IRR.DIGTCERT.LIST 74
IRR.DIGTCERT.LISTRING 74
IRR.RPKISERV.EXPORT 70
IRR.RPKISERV.GENCERT 70
IRR66007I 28
IRRADU00 79
IRRADULD 79
IRRADUTB 79
IRRD113I 36
IRRD00 25–26
IRRENS00 16–17
IRRGMAP 23–24, 29
IRRIRA00 20–21, 23, 27–29, 31, 33
IRRMIN00 24
IRROPT01 100
IRRSKP00 79
IRRSXPGL 56
IRRUMAP 23–24, 29
IRRUT200 21, 24, 26, 29

IRRUT400 21, 24, 26, 30
ISO
 8824 87
 8825 87

J

JAVA TR - tracing in Systems Management 215

K

KDC 86, 92
kdestroy 98, 113
Keep 17
keep 17
KERBDFLT 102
Kerberos 85
 limitations 92
Kerberos authentication 10
Kerberos command examples 109
Kerberos Commands 108
 kdestroy 108, 113
 keytab 108, 114
 kinit 108–109
 klist 108, 111
 ksetup 108
Kerberos commands
 definitions 108
Kerberos Principals 104
 Foreign principals 107
 Local principals 104
Kerberos protocol overview 86
KERBLINK 99
Key Distribution Center 86
keyring file
 create 158
keys
 generating for local principals 105
keytab 114
kinit 109
klist 111
kpasswd 102
krb5.conf 97, 111

L

LDAP 3
 Native Authentication 178
 Schema discovery 178
 Schema publication 178
LDAP and WebSphere 4.0.1 178
LDAP client enhancements 3
LDAP configuration utility LDAPCNF 5
LDAP customization for Policy Director 136
LDAP server
 Load schemas 226
 Load suffix entry for TDBM backend 227
 setup 226
 starting 226
 verify configuration 227
LDAP server Kerberos configuration 10

- LDAP tracing
 - Enable and disable 216
- ldap.db2.profile
 - customizing 221
- ldap.profile
 - customizing 220
- ldap.racf.profile
 - customizing 223
- ldap.slapped.profile
 - customizing 223
- ldapcnf utility
 - starting 223
- ldif2tdbm 178
- LNOTES 20, 26
- Loading the data 210
- Local Kerberos principals 99
- local principal names
 - considerations 107
- log files 259
- Logical console security 81
- login 244
- long-term key 88

M

- macro 168
- Make certificate available to the client 161
- managed association 99
- Mapping Kerberos principals to LDAP distinguished names 11
- Mark 17
- mark 16
- Migrating existing RDBM data to a TDBM database 194
- Migrating WebSphere Application Server 4.0.1 from RDBM to TDBM 194
- Migration of RDBM to TDBM 195
- Miscellaneous changes 12

N

- NAPS 85
 - implementing 92
- native authentication
 - installation 146
- Native authentication requirements 145
- native authentication services
 - starting and stopping 146
- NDS 20
- NDLINK 20, 22, 24, 30
- netd 244
- Netscape Navigator 44, 48
- Network Authentication and Privacy Service 85
- nfs 244
- ng 4
- nmap 261
- nonce 88
- NOTELINK 20, 22, 24, 30

O

- OCEP 33

- OMVS 20
- Open Cryptographic Enhanced Plug-ins 33
- optfield 53
- os 4
- OS/390 153
- Overview 48

P

- PADS 16
- PAM
 - account argument 257
 - auth argument 257
 - configuring 256
 - password argument 257
 - session argument 257
- PARMLIB, updating 225
- PassTicket 153, 157–158, 172
- PassTicket profile
 - defining to RACF 157
- passwd 254
- password authentication 255
- peer association 99
- peer trust between Kerberos realms
 - setting up 118
- PKI 48
- PKI.CONF 51, 61
- pki.conf 50
- PKIServ 48
- Pluggable Authentication Module (PAM), using 255
- Policy Director setup 137
- pop-3 244
- port scanning 261
- ports for SSL and client authentication
 - defining 159
- Preparing to use RACDCERT commands 234
- principal 86
- principal identifier 86
- Problem determination 149
- Problems encountered and lessons learned 187
- proclib, updating 225
- Program Control enhancements
 - errors 18
- program control enhancements 16
- PTKTDATA 102
- Public Key Infrastructure 48

R

- R_PKIServ callable service 70
- RACDCERT
 - ADDRING 73
 - CERTAUTH LIST 36
 - CONNECT 73–74
 - EXPORT 35, 37, 40
 - FORMAT 35
 - FORMAT CERTB64 35
 - FORMAT CERTDER 35
 - FORMAT PKCS12B64 35
 - FORMAT PKCS12DER 35, 38
 - PASSWORD 35

- GENCERT 34, 37
 - ALTNAME 34
 - DEBUG 35
 - KEYUSAGE 34, 36
 - KEYUSAGE CERTSIGN 34
- LIST 36–37, 56–57
- RACF 152–153, 157–158, 169
 - description 16
 - under z/VM 263
- RACF Auto Registration 48
- RACF Classes
 - APPL 102
 - KERBLINK 99, 106–107
 - PTKTDATA 102
 - REALM 99, 102–103
 - RRSFDATA 99
 - SERVAUTH 104
- RACF environment for LDAP server
 - setting up 130
- RACF functions
 - keep-controlled 17
- RACF jobs
 - executing 225
- RACF Remote Sharing Facility 99
- RACF services 17
- RACF setup for Kerberos realms 101
- RACF under z/VM 263
- RACF User ID Segments
 - KERB 99
 - LNOTES 21
 - NDS 21
 - OMVS 21
- RACFICE 24, 26
- RAR 48, 53
- RCVT 16
- RDBM 178
- REALM 99
- realm 91
- Redbooks Web site 272
 - Contact us xi
- Redirector 168
- Reset 18
- Restarting WebSphere Application Server after TDBM migration 211
- RLIST 18
- RRSFDATA 99
- RRSF 99, 101
 - Local Mode 99
- RRSF nodes 99
 - managed 99
 - peer 99
- Rules for LDAP security 183
- RVARY 21, 30
 - ACTIVE 81
 - NODATASHARE 80
 - SWITCH 81
- RVARY console samples 81

S

- SAF Browser Certificate 1 Year 54

- SAF Server Certificate 61
- scanlog 262
- schema entry
 - displaying 209
- scp 250
- SDBM enhancements 7
- sealed ticket 88
- Secure copy for copying files 250
- Secure login (slogin) 251
- Secure Shell for remote access 247
- security 243
 - basics 243
 - disabling services 244
 - monitoring the Web 259
 - VM 262
- server 5
- Server authentication 234
- server certificate, creating 158
- Server front-end performance/scalability 6
- Session negotiation 151
- SET LIST 101
- SETROPTS
 - LIST 79
 - RVARYPW 80
- Setting up the Kerberos environment variable files 97
- shadow password 253
- shell 244
- SKRBKDC 92, 97
 - daemon setup 92
- slogin 251
- smtp 244
- SNAME 20, 33
- Solaris 86
- SSH 247
 - client/server sources 252
 - daemon configuration 251
 - generating keys 248
 - key authentication 247
 - sources 252
- ssh 244
- SSH protocol v1 and v2
 - differences 252
- SSL 73, 75, 151, 162, 168
- STARTED class 95
- Structure of the LDAP configuration files 181
- SUPERUSER.FILES.MOUNT 79
- SYS1.PARMLIB
 - COFVLF00 29
- system jobs
 - executing 225

T

- TARGET LIST 101
- TDBM 3, 178
- TDBM Native authentication 9
- TDBM schema
 - setting up 205
- tdbm2ldif 178
- telnet 244
- Telnet-negotiated security 151

- session negotiation 151
- Test environment 130
- TGS 116
- The insecurity of Telnet 246
- Three-tier network design 155
- Ticket Granting Service 116
- ticket-granting server 87
- Ticket-granting service exchange 89
- ticket-granting ticket 86–87, 114, 116
- time 244
- tmplname 52
- TN3270 server
 - express logon 153, 168, 170, 172
- transactionid 52
- tripwire 260
 - config-file 261
- Two-tier network design 154

U

- UID 20
- UNAME 20, 33
- UNIXMAP 20, 22, 24, 26, 30
- UNIXPRIV
 - SUPERUSER.FILESYS.MOUNT 79
- useradd 254
- userdel 254
- usermod 254
- USSMSG10 170

V

- VLF 24
- VTAM 170

W

- WebSphere Application Server 4.0.1 and LDAP 181
- WebSphere Application Server 4.0.1 overview 179
- whoami 80
- Windows 2000 86
- Windows 2000 interoperation with OS/390 NAPS 117
- Windows 2000 use of Kerberos 116
- Working with native authentication 148

X

- xinetd 244–245

Z

- z/OS LDAP as Policy Director user registry 140
- z/OS LDAP server setup 130

Archived



Redbooks

Putting the Latest z/OS Security Features to Work

(0.5" spine)

0.475" <-> 0.873"

250 <-> 459 pages



Putting the Latest z/OS Security Features to Work

Install and configure an LDAP server

Utilize the latest RACF and Kerberos features

Integrate LDAP with WebSphere 4.0.1 and other products

Many significant enhancements have been made to the security features available on the z/OS and OS/390 platforms. This IBM Redbook will help you install, customize, and configure the new functions provided with Version 1.2 of SecureWay Security Server for z/OS and OS/390 2.10. It is useful for system programmers, security administrators, and webmasters enabling e-business on these platforms.

LDAP enhancements on z/OS 1.2 that are discussed include extended operations client APIs, Kerberos authentication, the LDAP Configuration Utility (LDAPCNF), new SDBM features, TDBM native authentication. Techniques to increase server front-end performance and scalability are also described.

Security enhancements on OS/390 2.10 are achieved by new and improved RACF features, and by a new component of the Security Server for OS/390, called Network Authentication and Privacy Services (NAPS). NAPS is the IBM OS/390 implementation of Kerberos Version 5.

LDAP can provide directory services to a wide range of other applications. This redbook provides detailed information on integrating LDAP with other products to implement secure distributed computing environments and extend host data access across intranets and the Internet via simple to use and yet highly protected communications. It also gives step-by-step instructions for integrating LDAP with WebSphere Application Server, including sample code for migrating from RDBM to TDBM.

The steps necessary to configure and start an LDAP server with TDBM and SDBM are presented, along with instructions for implementing password encryption for the TDBM backend on a z/OS LDAP server. LDAP can be configured to provide RACF digital certificate support; this is discussed in detail. Finally, some basic security techniques for Linux on zSeries and S/390 are presented.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:
ibm.com/redbooks**

SG24-6540-00

ISBN 0738424501