Security Target for IBM z/OS Version 2 Release 1 for Compliance with the NIAP/BSI OSPP

Version:	**1.7**
Status:	**Final Version**
Last Update:	**07/17/15**
Classification:	**Public**

Trademarks

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

- Advanced Function Presentation
- AFP
- BladeCenter
- DFS
- DFSORT
- eserver
- IBM
- Infoprint
- MVS
- PR/SM
- Print Services Facility
- Processor Resource/Systems Manager
- RACF
- System z
- VTAM
- z/Architecture
- zEnterprise
- z/OS
- z/VM
- zSeries

- z10

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

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

Revision History

Revision	Author(s)	Date	Changes to Previous Revision
1.7	Helmut Kurth	07/17/15	Final version for publication

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Security Target Identification

Title:	Security Target for IBM z/OS Version 2 Release 1 for Compliance with the NIAP/BSI OSPP
Version:	1.7
Status:	Final Version
Date:	07/17/15
Sponsor:	IBM Corporation
Developer:	IBM Corporation
Certification ID:	BSI-DSZ-CC-0972
Keywords:	access control, discretionary access control, general-purpose operating system, information protection, security, UNIX®

1.2 TOE Identification

The TOE is IBM z/OS Version 2 Release 1.

1.3 TOE Overview

This Security Target (ST) documents the security characteristics of the IBM z/OS Version 2 Release 1 operating system with the additional required licensed programs (see section Software configuration of this ST) configured in a secure manner as described in *z/OS Planning for Multilevel Security and the Common Criteria* ([MLSGUIDE]) except those configuration options related to labeled security mode..

IBM z/OS, a highly-secure, robust, scalable, high-performance enterprise operating system on which to build and deploy mission-critical applications, provides a comprehensive and diverse application execution environment. IBM z/OS is the flagship operating system for IBM System z™ mainframe computers, empowering the use of their most advanced features, such as the 64-bit z/Architecture™. It delivers the highest qualities of service for enterprise transactions and data and extends these qualities to new applications using the latest software technologies. IBM z/OS serves as the heart of customers' IT infrastructures, helping to integrate their information strategy and business strategy.

IBM z/OS can be used on a single IBM System z mainframe computer, or several systems or logical partitions running the evaluated version of IBM z/OS can be connected to form a loosely-coupled complex of systems called a *sysplex*.

IBM z/OS provides such software technologies as Enterprise Java™ Beans, eXtensible Markup Language (XML), HyperText Markup Language (HTML), Unicode and distributed Internet Protocol (IP) networking. z/OS UNIX System Services allows customers to develop and run UNIX programs on z/OS and exploit the reliability and scalability of the System z processors. z/OS also incorporates cryptographic services, distributed print services, workload management, storage management, parallel sysplex availability, and automation capabilities. Not all of these functions have been analyzed in this evaluation; see section Software configuration for the software configuration of z/OS used in this evaluation. The security functions subject to this evaluation are described in chapter 8 of this document.

IBM z/OS provides identification and authentication of users using different authentication mechanisms, discretionary access control to a large number of different objects, a configurable audit functionality, protection of communication services, sophisticated security management functions, preparation of objects for reuse and functionality used internally to protect z/OS from interference and tampering by untrusted users or subjects.

1.4 TOE Description

The Target of Evaluation (TOE) is the z/OS operating system with the software components as described in section Software configuration. z/OS is a general-purpose, multi-user, multi-tasking operating system for enterprise computing systems. Multiple users can use z/OS simultaneously to perform a variety of functions that require controlled, shared access to the information stored on the system.

In this ST, the TOE is seen as one instance of z/OS running on an abstract machine as the sole operating system and exercising full control over this abstract machine. This abstract machine can be provided by one of the following:

- a logical partition provided by a certified version of PR/SM on an IBM System z™ processor (IBM System z10™ Business Class, IBM System z10™ Enterprise Class, zEnterprise 114, zEnterprise 196, or zEnterprise EC12).
- a certified version of z/VM® executing in a logical partition provided by PR/SM on one of the above-listed System z™ processors.

If the configuration includes a zEnterprise BladeCenter Extension (zBX), the operating systems running in the zBX are not part of the TOE. They are external systems, connected to z/OS only via the built-in TCP/IP networking facilities included in the zEnterprise System and zBX.

The abstract machine itself is not part of the TOE, rather, it belongs to the TOE environment. Nevertheless the correctness of separation and memory protection mechanisms implemented in the abstract machine is analyzed as part of the evaluation since those functions are crucial for the security of the TOE.

The TOE environment, as part of the System z processor, also includes specific hardware functions that provide support for the cryptographic operations involved in communications security and for the digital signature operations involved with X.509v3 digital certificates.

Multiple instances of the TOE may be connected in a basic sysplex or in a parallel sysplex with the instances sharing their RACF® database.

The platforms selected for the evaluation consist of IBM products that are available when the evaluation has been completed and will remain available for a substantial period of time afterward.

The individual TOEs can be run alone or within a network as a set of cooperating hosts, operating under and implementing the same set of security policies.

Most of the TOE security functions (TSF) are provided by the z/OS operating system Base Control Program (BCP) and the Resource Access Control Facility (RACF), a z/OS component that is used by different services as the central instance for identification and authentication and for access control decisions. z/OS comes with management functions that allow configuring of the TOE security functions to tailor them to the customer’s needs.

Some elements have been included in the TOE that do not provide security functions. These elements run in authorized mode, so they could compromise the TOE if they do not behave properly. Because these elements are essential for the operation of many customer environments, the inclusion of these elements subjects them to the process of scrutiny during the evaluation and ensures that they may be used by customers without affecting the TOE’s security status.

1.4.1 Intended Method of Use

z/OS provides a general computing environment that allows users to gain controlled access to its resources in different ways:

- online interaction with users through Time Sharing Option Extensions (TSO/E) or z/OS UNIX System Services
- batch processing (JES2)
- services provided by started procedures or tasks
- daemons and servers utilizing z/OS UNIX System Services that provide similar functions as started procedures or tasks but based on UNIX interfaces

These services can be accessed by users local to the computer systems or accessing the systems via network services supported by the evaluated configuration.

All users of the TOE are assigned a unique user identifier (user ID). This user ID, which is used as the basis for access control decisions and for accountability, associates the user with a set of security attributes. In most cases the TOE authenticates the claimed identity of a user before allowing this user to perform any further security-relevant actions. Exceptions to this authentication policy include:

i. Pre-specified identities:

   a. The authorized administrator can specify an identity to be used by server or daemon processes or system address spaces, which may be started either automatically or via system operator commands;
   b. The authorized administrator may configure a trusted HTTP server to access selected data under a specified identity, rather than the identity of the end user making the request. The HTTP server may optionally authenticate the user in this case, or may serve the data to anyone asking for it, if the administrator has determined that such anonymous access is appropriate.

ii. Users are allowed to execute programs that accept network connections on ports the user has access to. In this case the untrusted program has no knowledge about the external "user" and cannot perform authentication. The program executes with the rights of the z/OS user that started it, and any data access occurs using this user's authenticated identity.

The TOE provides mechanisms for discretionary access control.

All TOE resources are under the control of the TOE. The TOE mediates the access of subjects to TOE-protected objects. Subjects in the TOE are called *tasks*. Tasks are the active entities that can act on the user's behalf. Data is stored in named objects, which are data sets, UNIX file system objects, UNIX IPC objects or Directory Server (LDAP) objects. The TOE can associate a set of security attributes with each named object, which includes the description of the access rights to that object.

Objects are owned by users, who are assumed to be capable of assigning discretionary access rights to their objects in accordance with the organizational security policies. Ownership of named objects can be transferred under the control of the access control policy. The security attributes of users, data objects, and objects through which the information is passed are used to determine if information may flow through the system as requested by a user.

Apart from normal users, z/OS recognizes administrative users with special authorizations. These users are trusted to perform system administration and maintenance tasks, which includes configuration of the security policy enforced by the z/OS system and attributes related to it. Authorizations can be delegated to other administrative users by updating their security attributes.

The TOE also recognizes the role of an *auditor*, who uses the auditing system provided by z/OS to monitor the system usage according to the organizational security policies.

The TOE is intended to operate in a networked environment with other instantiations of the TOE as well as other well-behaved client systems operating within the same management domain. All of those systems need to be configured in accordance with a defined common security policy.

1.4.2 Summary of Security Features

The primary security features of the product are:

- identification and authentication
- discretionary access control

- auditing
- object reuse
- security management
- secure communication
- TSF protection

These primary security features are supported by domain separation and reference mediation, which ensure that the features are always invoked and cannot be bypassed.

1.4.2.1 Identification and authentication

z/OS provides identification and authentication of users by the means of

- an alphanumeric RACF user ID and a system-encrypted password or (for applications that support it) password phrase.
- an alphanumeric RACF user ID and a PassTicket, which is a cryptographically-generated password substitute encompassing the user ID, the requested application name, and the current date/time.
- an X.509v3 digital certificate presented to a server application that uses System SSL or TCP/IP Application Transparent TLS (AT-TLS) to provide TLS-based client authentication, and then “mapped” (using TOE functions) by that server application or by AT-TLS to a RACF user ID.
- a Kerberos™ v5 ticket presented to a server application that supports the Kerberos mechanism, and then mapped by that application through the TOE-provided GSS-API programming services or alternate functions that are also provided by the TOE (specifically the R_ticketServ, and R_GenSec services). These functions enable the application server to validate the Kerberos ticket, and thus the authentication of the principal. The application server then translates (or maps) the Kerberos principal (using the TOE provided function of R_userMap) to a RACF user ID.
- an LDAP LDBM bind DN (which is mapped to a RACF user ID by information in the LDAP directory) or an LDAP ICTX or SDBM bind DN (which contains a RACF user ID) together with a RACF password or password phrase. The bind processing then passes the derived RACF user ID, and the password/phrase, to RACF to complete the authentication process.
- a digital certificate presented to LDAP over TLS (LDAP SASL bind with EXTERNAL verification) which must map to a RACF USER ID.

In the evaluated configuration, all human users are assigned a unique user ID. This user ID supports individual accountability. The TOE security functions authenticate the claimed identity of the user by verifying the password/phrase (or other mechanism, as listed above) before allowing the user to perform any actions that require TSF mediation, other than actions that aid an authorized user in gaining access to the TOE.

In some cases of external access to the system, such as the HTTP server, or LDAP server, an installation may decide to define a user ID that is used for access checking of selected resources for users that have not been authenticated. This allows an installation to define resources unauthenticated users may access using that server via an appropriate client program. Users may still authenticate to the server using their user ID and password/phrase (or other authentication mechanism as above) to access additional resources they have been assigned access to.

The required password quality can be tailored to the installation's policies using various parameters. When creating users, administrators are required to choose an initial password and optionally a password phrase, that must usually be changed by the user during the initial logon that uses the password/phrase.

1.4.2.2 Discretionary access control

z/OS supports access controls that are capable of enforcing access limitations on individual users and data objects. Discretionary access control (DAC) allows individual users to specify how such resources as direct access storage devices (DASD), DASD and tape data sets, and tape volumes that are under their control are to be shared.

RACF makes access control decisions based on the user's identity, security attributes, group authorities, and the access authority specified with respect to the resource profile.

z/OS provides the following DAC mechanisms:

i. The z/OS standard DAC mechanism is used for most traditional (non-UNIX) protected objects.
ii. The z/OS UNIX DAC mechanism is used for z/OS UNIX objects (files, directories, etc.) except IPC objects
iii. The z/OS UNIX IPC DAC mechanism is used for z/OS UNIX IPC objects (shared memory segments, semaphores)
iv. The z/OS LDAP LDBM DAC mechanism is used to protect LDAP objects in the LDAP LDBM back-end data store. [Note: z/OS LDAP also supports the CDBM back-end. Statements in this Security Target about LDBM, including how DAC mechanisms work, apply to both LDBM and CDBM. However, CDBM can hold administrative policies for the LDAP server, and LDAP applies additional, CDBM-specific access control policies to the CDBM directory information tree containing those policies when accessed by a member of the LDAP administrative group, as described later in the LDAP Management section.]

z/OS standard DAC mechanism

Access types that can be granted are NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER, which form a hierarchical set of increasing access authorities.

Access authorities to resources are stored in profiles. Discrete profiles are valid for a single, named resource and generic profiles are applicable to a group of resources, typically with similar names. For access permission checks, RACF always chooses the most specific profile for a resource. Profiles can have an access control list associated with them that contains a potentially large number of entries for different groups and users, thus allowing the modeling of complex, fine-grained access controls.

Profiles are assigned to a number of resources within z/OS. This Security Target defines the resource types analyzed during the evaluation. RACF profiles are also used to manage and control privileges in z/OS and resources of subsystems that are not part of the evaluated configuration (e. g. DB2, CICS, JES3).

Access rights for subjects to resources can be set by the profile owner and by the system administrator.

The TOE allows access decisions by this mechanism for local applications or remote applications. For local applications the application, or the TOE, uses the RACROUTE programming interface to perform the access check. Remote applications can perform similar access checking via LDAP interfaces, if the z/OS ITDS LDAP server is appropriately configured, by first authenticating (binding) with an ICTX-style identity (DN), and then providing an extended-operation request indicating that the applications wants do perform an access check. LDAP will then invoke the ICTX extended operation processing routine which will check the application’s authority to make such a request, and then will process the request if authorized. The request specifies the resource to be checked and the RACF user ID or group name whose access should be checked.

z/OS UNIX DAC mechanism

z/OS implements POSIX-conformant access control for such named objects in the UNIX realm as UNIX file system objects. . Access types for UNIX file system objects are read, write, and execute/search. z/OS file system objects provide either access control based on the permission bits associated with a file, or based on access control lists, which are upward-compatible with the permission bits algorithm and implement the recommendations from Portable Operating System Interface for UNIX (POSIX) 1003.1e draft 17.

z/OS UNIX IPC DAC mechanism

z/OS implements POSIX-conformant access control for UNIX IPC objects. Access types are read and write and are based on permission bits.

z/OS LDAP DAC mechanism

The z/OS LDAP server supports several back-end data stores as well as plug-ins. Three of the data stores (LDBM, CDBM, SDBM) and three plug-ins (ICTX, Remote PKCS#11, Remote CCA) can be used in the evaluated configuration. The SDBM back-end allows RACF administration by remote administrators for systems configured in standard operation mode. The LDBM back-end allows storage of customer data and this back-end supports a standard LDAP access control mechanism to control which authenticated users can access which data. It also supports the possibility of “public” data, accessed by unauthenticated users, when the administrator has configured this kind of data and access.

The ICTX plug-in allows remote servers to issue authorization check or auditing requests to RACF.

Through the remote PKCS#11 plug-in, the z/OS LDAP server provides client applications the ability to utilize PKCS#11 cryptography provided through ICSF (Integrated Cryptographic

Security Facility). PKCS#11 is a Public Key Cryptographic Standard (PKCS) that defines a platform independent API for using cryptographic tokens. This standard defines the types of cryptographic tokens supported and how to create, delete and use (encrypt, decrypt, hash data) tokens in cryptographic operations. The remote PKCS#11 plug-in supports the RemoteCryptoPKCS#11 extended operation which allows any LDAP client application authorized (successfully bound and authenticated) to perform any PKCS#11 API by invoking the appropriate ICSF callable service. The RemoteCryptoPKCS#11 extended operation is a generic extended operation that allows an LDAP client application to specify the same data as if invoking the ICSF callable service locally.

Through the remote CCA plug-in, the z/OS LDAP server provides client applications the ability to utilize CCA cryptography provided through ICSF (Integrated Cryptographic Security Facility). The remote CCA plug-in supports the RemoteCryptoCCA extended operation which allows any LDAP client application authorized (successfully bound and authenticated) to perform any CCA callable service by invoking the appropriate ICSF callable service. The RemoteCryptoCCA extended operation is a generic extended operation that allows an LDAP client application to specify the same data as if invoking the ICSF callable service locally.

1.4.2.3 Auditing

The TOE provides an auditing capability that allows generating audit records for security-critical events. RACF provides a number of logging and reporting functions that allow resource owners and auditors to identify users who attempt to access resources. Audit records are collected by the System Management Facilities (SMF) into an audit trail, which is protected from unauthorized modification or deletion by the DAC mechanisms. This audit trail can reside directly in MVS data sets, or in an MVS log stream (which can be automatically off-loaded into MVS data sets), as configured by the administrator.

The system can be configured to halt on exhaustion of audit trail space to prevent audit data loss.

Operators are warned when audit trail space consumption reaches a predefined threshold.

RACF always generates audit records for such events as unauthorized attempts to access the system or changes to the status of the RACF database. The security administrator, auditors, and other users with appropriate authorization can configure which additional optional security events are to be logged. In addition to writing records to the audit trail, messages can be sent to the security console to immediately alert operators of detected policy violations. RACF provides SMF records for all RACF-protected resources (either “traditional” or z/OS UNIX-based) as well as for LDAP-based resources.

Remote applications can use an LDAP interface to request that RACF generate an SMF audit record, if the z/OS ITDS LDAP server is appropriately configured, by first authenticating (binding) with an ICTX-style identity (DN) and then providing an extended-operation request indicating that the applications wants do generate an audit record. LDAP will then invoke the ICTX extended operation processing routine, which will check the application’s authority to make such a request, and then will process the request if authorized. The request specifies the information to be audited.

For reporting, auditors can unload all or selected parts of the SMF data for further analysis in a human-readable formats and can then upload the data to a query or reporting package, such as DFSORT™ if desired.

1.4.2.4 Object reuse functionality

Reuse of protected objects and of storage is handled by various hardware and software controls, and by administrative practices.

All memory content of non-shared page frames is cleared before making it accessible to other address spaces or data spaces. DASD data sets can be purged during deletion with the RACF ERASE option and tape volumes can be erased on return to the scratch pool. All resources allocated to UNIX objects are cleared before reuse. Other data pools are under strict TOE control and cannot be accessed directly by normal users.

1.4.2.5 Security management

z/OS provides a set of commands and options to adequately manage the TOE’s security functions. Additionally, the TOE provides the capability of managing users, groups of users, general resource profiles, and RACF SETROPTS options via the z/OS LDAP server, which can accept LDAP-format requests from a remote administrator and transform them into RACF administrative commands via its SDBM backend processing. The TOE also provides a Java class that allows Java programs to issue commands to manage users and groups. Both the LDAP SDBM and the Java class ultimately create a RACF command and pass it to RACF using a programming interface, and then RACF runs the command using the identity associated with the SDBM session or the Java program. This behaves just the same as when a local administrator issues the command, including all the same security checking and auditing.

The TOE recognizes several authorities that are able to perform the different management tasks related to the TOE’s security:

- General security options are managed by security administrators.
- Management of users and their security attributes is performed by security administrators. Management of groups (and to some extent users) can be delegated to group security administrators.
- Users can change their own passwords or password phrases, their default groups, and their user names (but not their user IDs).
- Auditors manage the parameters of the audit system (a list of audited events, for example) and can analyze the audit trail.
- Security administrators can define what audit records are captured by the system.
- Discretionary access rights to protected resources are managed by the owners of the applicable profiles (or UNIX objects) or by security administrators.

1.4.2.6 Communications Security

z/OS provides means of secure communication between systems.

In its evaluated configuration, z/OS supports trusted communication channels for TCP/IP connections. The confidentiality and integrity of network connections are assured by Transport Layer Security (TLS) encrypted communication for TCP/IP connections ([TLSV1.1], [TLSV1.2]), which can be used explicitly by applications or applied transparently to their communications (AT-TLS) without changing the applications using it (assuming the applications that do not make use of the TLS capabilities that allow clients to authenticate to the system using a client-supplied X.509 digital certificate. If applications accept client certificates then they do need to have specific TLS-related processing within the applications.).

In addition to the TLS connection, z/OS also supports the IP Security (IPSec) protocol with Internet Key Exchange (IKE) as the key exchange method. This is an additional way to set up a trusted channel to another trusted IT product for IP-based connections. z/OS also provides centralized policy management for IPSec and AT-TLS policies across multiple z/OS systems in the network. It also provides centralized management for digital certificates, message signing, and message verification for IPSec across multiple z/OS systems in the network.

z/OS also supports Kerberos™ version 5 networking protocols, via the Integrated Security Services Network Authentication Service component, hereafter called z/OS Network Authentication Service These protocols enable both the client and the server to mutually authenticate. This authentication mechanism can be utilized with the GSS-API services provided by the z/OS Network Authentication Service to provide security services to applications. These services enable encrypted communications channels between clients and servers that may reside on the same or on different systems.

z/OS also supports, via the optional add-on product IBM Ported Tools for z/OS, the SSH v2 protocol and the ssh-daemon provided services of ssh (secure shell), scp (secure copy), and sftp (secure ftp) ([SSHV2])

TCP/IP-based communication can be further controlled by the access control function for TCP/IP connections, which allows controlling of the connection establishment based on access to the TCP/IP stack in general, specific network zones and individual ports on a per-application or per-user basis.

z/OS provides also a variety of network services, all of which use RACF for identification, authentication, and access control. In the evaluated configuration, terminal services are provided by TN3270, telnet, rlogin, rsh, and rexec. File transfer services are provided by the File Transfer Protocol (FTP), sftp and scp, Web serving functions are provided by the z/OS HTTP Server.

1.4.2.7 TSF protection

TSF protection is based on several protection mechanisms that are provided by the underlying abstract machine:

- Privileged processor instructions are only available to programs running in the processor's supervisor state

- Semi-privileged instructions are only available to programs running in an execution environment that is established and authorized by the TSF

- While in operation, all address spaces, as well as the data and tasks contained therein, are protected by the memory protection mechanisms of the underlying abstract machine

The TOE's address space management ensures that programs running in problem state cannot access protected memory or resources that belong to other address spaces.

Access to system services – through supervisor call (SVC) or program call (PC) instructions, for example – is controlled by the system, which requires that subjects who want to perform security-relevant tasks be authorized appropriately.

The hardware and firmware components that provide the abstract machine for the TOE are required to be physically protected from unauthorized access. The z/OS Base Control Program mediates all access to the TOE's hardware resources themselves, other than program-visible CPU instruction functions.

Tools are provided in the TOE environment to allow authorized administrators to check the correct operation of the underlying abstract machine.

In addition to the protection mechanism of the underlying abstract machine, the TOE also uses software mechanisms like the authorized program facility (APF) or specific privileges for programs in the UNIX system services environment to protect the TSF.

1.4.3 Configurations

1.4.3.1 Software configuration

The Target of Evaluation, z/OS Version 2 Release 1, consists of:

- z/OS Version 2 Release 1 (V2R1) Common Criteria Evaluated Base Package:
    - z/OS Version 2 Release 1 (z/OS V2R1, program number 5650-ZOS),
    - Overlay Generation Language Version 1 (OGL V1R1, program number 5688-191)
    - IBM Print Services Facility™ Version 4 Release 4 for z/OS (PSF V4.4.0, program number 5655-O15)
- IBM Ported Tools for z/OS V1.2 (FMID HOS1120, program number 5655-M23, optional) along with the IBM Ported Tools for z/OS: Supplementary Toolkit for z/OS Feature (also optional) and with the PTF UA63842 for APAR OA37278 (which allows to configure OpenSSH to use ICSF for the symmetric algorithms TDES and AES and for the SHA-1 and SHA-2 family of hash functions).
- The following APARs (or their associated PTFs):

APAR	**PTF**
OA41946	UA70056
OA38971	UA69014
OA41985	UA70053
OA43423	UA71183
OA42093	UA70345
OA41809	UA70548
PM91543	UK97738, UK97739
OA43149	UA70900
OA42679	
PM87944	UI12779
OA43712	UA71638
OA43935	UA90715
OA43539 OA43457	UA71266, UA71250
PI06650	UI90002
OA43550	UA71082
OA43536	UA71853
OA43350	UA90693
OA43794	
0A43650	UA90696
OA43812	UA90702
OA43741	UA71552
OA43398	UA71591

Note: References in this document to “HTTP Server” refer to the IBM HTTP Server Base (FMID HIMW530) and IBM HTTP Server North America Secure (JIMW531) that ship as part of z/OS, and not to the IBM Ported Tools for z/OS HTTP Server, which must not be used in the evaluated configuration.

The same software elements are used in the Labeled Security and standard modes of operation, except as otherwise noted. The mode of operation is defined by the configuration of the labeling-related options in RACF. Details are described in *z/OS Planning for Multilevel Security and the Common Criteria* ([MLSGUIDE]).

The z/OS V2R1 Common Criteria Evaluated Base package, and (if used) IBM Ported Tools for z/OS must be installed according to the directions delivered with the media and configured according to the applicable instructions in Chapter 7, “The evaluated configuration for the Common Criteria” in *z/OS Planning for Multilevel Security and the Common Criteria* ([MLSGUIDE]).

Installations may choose not to use any of the elements delivered within the ServerPac, but are required to install, configure, and use at least the RACF and ICSF components of the z/OS Security Server element.

In addition, any software outside the TOE may be added without affecting the security characteristics of the system, if it cannot run:

- in supervisor state
- as APF-authorized
- with keys 0 through 7
- with UID(0),
- with authority to FACILITY resources BPX.DAEMON, BPX.SERVER, or BPX.SUPERUSER
- with authority to UNIXPRIV resources

This explicitly excludes:

- replacement of any element in the ServerPac providing security functions relevant to this evaluation by other third-party products;
- installing system exits that run authorized (supervisor state, system key, or APF-authorized), with the exception of the sample ICHPWX11 and its associated IRRPHREX routine;
- installing IBM Tivoli Directory Server plug-ins that have not been evaluated;
- using the Authorized Caller Table (ICHAUTAB) in RACF to allow unauthorized programs to issue RACROUTE REQUEST=VERIFY (RACINIT) or RACROUTE REQUEST=LIST (RACLIST).

Note: The evaluated software configuration is not invalidated by installing and operating other appropriately-certified components that possibly run authorized. However the evaluation of those components must show that the component and the security policies implemented by the component do not undermine the security policies described in this document.

The IBM Tivoli Directory Server for z/OS component may be used as the LDAP server, but:

- For client authentication via digital certificates the administrator must configure the LDAP server to map the certificate to a RACF user ID and to fail the bind if the certificate does not map to a RACF user ID. The allowable LDAP configuration provides the following options for forming an LDBM subject:
    - LDAP may use the original DN from the certificate; or

    - LDAP may replace the original DN with an SDBM-format DN based on the RACF user ID;
    - or LDAP may add the SDBM-format DN to the LDAP subject, giving a subject with two DNs, either of which will work in LDAP ACLs.
- client authentication using the Kerberos mechanism has not been evaluated for LDAP and cannot be used in the evaluated configuration.
- authentication via passwords stored in LDAP cannot be used. Authentication must occur using RACF passwords, password phrases or Passtickets. Note that if an LDBM bind DN is specified when binding to the server, the password/phrase specified must be for the RACF user ID associated with that bind DN by the LDAP administrator.
- the LDBM, CDBM, and SDBM back-ends and the ICTX plug-in may be used. Other LDAP back-end configurations and plug-ins have not been evaluated and must not be used.

sshd (from IBM Ported Tools for z/OS), may be used, but if used:

- must be configured to use protocol version 2 and either TDES or one of the AES-based encryption suites,
- must be configured in privilege separation mode, and
- must be configured to allow only password-based (including password phrase) authentication of users or public-key based authentication of users with the public keys stored in RACF keyrings. Rhost-based and public-key based user authentication with the keys stored elsewhere may not be used in the evaluated configuration.

The Network Authentication Service component of the Integrated Security Services component, if used, and applications exploiting it, must satisfy the following constraints:

- the Network Authentication Service must use the SAF (RACF) registry. The NDBM registry is not a valid configuration for this evaluation.
- Cross Realm Trust relationships with foreign Kerberos realms is allowed, but the foreign KDC must be capable of supporting the same cipher as does the z/OS KDC.
- In order to ensure strong cryptographic protection of Kerberos tickets, Triple DES or AES should be utilized by the z/OS KDC and any KDC participating in a cross-realm trust relationship with the z/OS KDC. DES should only be used in network environments where the threat of cryptographic attacks against the tickets and Kerberos-protected sessions is deemed low enough to justify the use of these weaker encryption protocols.
- Applications supporting Kerberos may use a combination of application specific protocols and the GSS-API functions or the equivalent native platform callable services (the SAF R_TicketServ and R_GenSec callable services) to authenticate

clients, and in client-server authentication. Only the Kerberos mechanism may be used by applications that utilize GSS-API or the equivalent native platform functions. The GSS-API and R_GenSec services also enable the encryption of sensitive application messages passed via application specific protocols. These services enable the secure communication between client and server applications. The GSSAPI services include the message integrity and privacy functions that validate the authenticity and secure the communications between clients and servers.

The Network File System (NFS) Server may be used, but must be configured with the SAF or SAFEXP option, to ensure that all file and directory access (except possibly directory mounting) has appropriate RACF security checks made.

TLS (Transport Layer Security) processing, if used, must use TLS V1.1 or TLS V1.2 protocols. TLS (Transport Layer Security), if used, must use one of the cipher suites listed for TLS in the FTP_ITC.1 SFR.

IPSec (IP Security) processing, if used, must use one of the cipher suites listed for IPSec in the FTP_ITC.1 SFR.

Any application performing client authentication using client digital certificates over TLS must be configured to use RACF profiles in the RACDCERT or DIGTRING classes or PKCS#11 tokens in ICSF to store the keyrings that contain the application private key and the allowed Certificate Authority (CA) certificates that may be used to provide the client certificates that the application will support. The use of gskkyman for this purpose is not part of the evaluated configuration.

Any client that is delivered with the product that executes with the user's privileges must be used with care, since the TSF can not protect those clients from potentially hostile programs. Passwords/phrases a user enters into those client programs that those clients use to pass to the corresponding server to authenticate the user may potentially be spoofed by hostile programs running in the user's address space. This includes client programs for telnet, TN3270, ftp, r-commands, ssh, all LDAP utilities and Kerberos administration utilities that require the user to enter his password/phrase. When using those client programs the user should take care that no untrusted potentially hostile program has been called during his session.

The following elements and element components cannot be used in an evaluated system, either because they violate the security policies stated in this Security Target or because they have been removed from the evaluated configuration due to time and resource constraints of the evaluation. As they are part of the base system, either they must be not configured for use or they must be deactivated, as described in Chapter 7, “The evaluated configuration for the Common Criteria” in
z/OS Planning for Multilevel Security and the Common Criteria ([MLSGUIDE]):

- All Bulk Data Transfer (BDT) elements: BDT, BDT File-to-File , and BDT Systems Network Architecture (SNA) NJE
- Connection Manager
- The DFS™ Server Message Block (SMB) (FMID H0H23B0) components of the Distributed File Service element

- The Enterprise Identity Mapping component of the Integrated Security Services element
- Infoprint® Server
- JES3
- The Advanced Program-to-Program Communication / Multiple Virtual Storage (APPC/MVS) component of the BCP
- Process Manager component from the UNIX System Services Element

The use of TCP/IP communication for JES2 NJE has not been part of the evaluation and must not be used in the evaluated configuration.

The JES2 Execution Batch Monitor (XBM) facility has not been part of the evaluation and must not be used in the evaluated configuration.

The RACF Remote Sharing Facility has not been part of the evaluation and must not be used in the evaluated configuration.

The Data Facility Storage Management Subsystem (DFSMS) Object Access Method for content management type applications must not be used.

The IBM Ported Tools for z/OS HTTP Server V7.0 (FMID HHAP700) must not be used.

For the Communications Server:

- The z/OS FTP server and client, and the z/OS TN3270 server, support both manually-configured TLS, or AT-TLS. This evaluation has considered only AT-TLS configurations, and as a result manual configuration of those components to use TLS is not allowed for evaluated configurations.
- The z/OS FTP server and client can support either the protocols from the draft standard for securing FTP with TLS, or the protocols from the formal RFC 4217 level of Security FTP with TLS. This evaluation has considered only the formal RFC 4217 level of support, and as a result that option must be used in the evaluated configuration.

1.4.3.2 Hardware configuration

The following assumptions about the technical environment in which the TOE is intended to be used are made:

The TOE is running within a logical partition provided by a certified version of PR/SM, on the z/Architecture as implemented by the following hardware platforms:

- IBM System z10 Business Class with CP Assist for Cryptographic Functions (CPACF) DES/TDES Enablement Feature 3863 active, optionally with CryptoExpress3 card.
- IBM System z10 Enterprise Class with CPACF DES/TDES Enablement Feature 3863 active, optionally with CryptoExpress3 card.

- IBM zEnterprise 114 with CPACF DES/TDES Enablement Feature 3863 active, optionally with CryptoExpress3 card, and with or without the zEnterprise BladeCenter Extension (zBX).
- IBM zEnterprise 196 with CPACF DES/TDES Enablement Feature 3863 active, optionally with CryptoExpress3 card, and with or without the zEnterprise BladeCenter Extension (zBX).
- IBM zEnterprise EC12 with CPACF DES/TDES Enablement Feature 3863 active, optionally with CryptoExpress3 or CryptoExpress4S card, and with or without the zEnterprise BladeCenter Extension (zBX).

In addition, the TOE may run on a virtual machine provided by a certified version of z/VM.

The following peripherals can be used with the TOE, while still preserving the security functionality:

- All terminals that are supported by the TOE.
- Printers:
  - any printer that is supported by the TOE.
- All storage devices and backup devices supported by the TOE, such as:
  - Direct access storage devices (DASD), except RVA devices.
  - Tape drives (including encrypting tape drives, though this evaluation has not specifically examined those cryptographic functions).
- All Ethernet and token-ring network adapters that are supported by the TOE.

Note: The peripherals may be virtualized in the case of the TOE executing within a logical partition or z/VM. The logical partitioning software and z/VM software is part of the abstract machine and, therefore, part of the TOE environment. The logical partitioning software documentation as well as the z/VM documentation provides the required guidance on how to set up and configure the logical partitioning software or z/VM and how to define the logical peripheral devices so the TOE operates securely in the logical partitioning or z/VM environment.

1.4.4 Structure

The structure of this document is:

- Chapter 1 provides the ST Introduction
- Chapter 2 provides the CC Conformance Claim
- Chapter 3 provides the Security Problem Definition
- Chapter 4 provides the Security Objectives

- Chapter 5 provides the Extended Components Definition
- Chapter 6 provides the Security Requirements for the Operational Environment
- Chapter 7 provides the TOE Security Requirements
- Chapter 8 provides the TOE Summary Specification
- Chapter 9 provides Abbreviations, Terminology and References

2 CC Conformance Claim

This ST is CC Part 2 extended and CC Part 3 conformant, with the security assurance requirements as defined in [GSOSPP].

This ST claims conformance to the following Protection Profiles:

- [GPOSPP] General-Purpose Operating System Protection Profile. Version 3.9 as of 2012-12-06; strict conformance.

Common Criteria version 3.1 revision 4 has been taken as the basis for this conformance claim.

3 Security Problem Definition

3.1 Introduction

The statement of the TOE security problem definition describes the security aspects of the environment in which the TOE is intended to be used and the manner in which it is expected to be deployed.

To this end, the statement of the TOE security environment identifies the list of assumptions made on the operational environment (including physical and procedural measures) and the intended method of use of the product, defines the threats that the product is designed to counter, and the organizational security policies with which the product is designed to comply.

The threats, assumptions, organizational security policies and security objectives have been taken from [GPOSPP].

3.2 Threat Environment

Threats to be countered by the TOE are characterized by the combination of an asset being subject to a threat, a threat agent and an adverse action.

3.2.1 Assets

Assets to be protected are:

- storage objects used to store user data and/or TSF data, where this data needs to be protected from any of the following operations:
    - Unauthorized read access
    - Unauthorized modification
    - Unauthorized deletion of the object
    - Unauthorized creation of new objects
    - Unauthorized management of object attributes
- TSF functions and associated TSF data
- The resources managed by the TSF that are used to store the above-mentioned objects, including the metadata needed to manage these objects

3.2.2 Threat agents

Threat agents are external entities that potentially may attack the TOE. They satisfy one or more of the following criteria:

- External entities not authorized to access assets may attempt to access them either by masquerading as an authorized entity or by attempting to use TSF services without proper authorization.
- External entities authorized to access certain assets may attempt to access other assets they are not authorized to either by misusing services they are allowed to use or by masquerading as a different external entity.

- Untrusted subjects may attempt to access assets they are not authorized to either by misusing services they are allowed to use or by masquerading as a different subject.

Threat agents are typically characterized by a number of factors, such as expertise, available resources, and motivation, with motivation being linked directly to the value of the assets at stake. The TOE protects against intentional and unintentional breach of TOE security by attackers possessing an enhanced-basic attack potential.

The following threats are addressed by the TOE. All threats have been copied from the OSPP. As in the OSPP, there are no threats and policies to justify the assurance level.

3.2.3 Threats countered by the TOE

T.ACCESS.TSFDATA

A threat agent might read or modify TSF data without the necessary authorization.

T.ACCESS.USERDATA

A threat agent might gain access to user data stored, processed or transmitted by the TOE without being appropriately authorized according to the TOE security policy by using functions provided by the TOE.

T.ACCESS.TSFFUNC

A threat agent might use or manage functionality of the TSF bypassing protection mechanisms of the TSF.

T.ACCESS.COMM

A threat agent may access cryptographically protected data transferred via a trusted channel between the TOE and another remote trusted IT system, modify such data during transfer in a way not detectable by the receiving party or masquerade as a remote trusted IT system.

T.RESTRICT.NETTRAFFIC

A threat agent may send data packets to the recipient in the TOE via a network communication channel in violation of the information flow control policy.

T.IA.MASQUERADE

A threat agent may masquerade as an authorized entity including the TOE itself or a part of the TOE in order to gain unauthorized access to user data, TSF data, or TOE resources.

T.IA.USER

A threat agent may gain access to user data, TSF data or TOE resources with the exception of public objects without being identified and authenticated by the TSF.

T.UNATTENDED_SESSION

A threat agent may gain unauthorized access to an unattended session.

3.3 Assumptions

This section describes the security aspects of the environment in which the TOE is intended to be used. It includes information about the physical, personnel, procedural, and connectivity aspects of the environment.

The TOE is assured to provide effective security measures in a cooperative non-hostile environment only if it is installed, managed, and used correctly. The operational environment must be managed in accordance with user/administrator guidance documentation. The following specific conditions are assumed to exist in an environment where the TOE is employed.

3.3.1 Environment of use of the TOE

3.3.1.1 Physical aspects

The TOE is intended for application in user areas that have physical control and monitoring. It is assumed that the following physical conditions will exist:

A.PHYSICAL

It is assumed that the IT environment provides the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE.

3.3.1.2 Personnel aspects

A.MANAGE

The TOE security functionality is managed by one or more competent individuals. The system administrative personnel are not careless, willfully negligent, or hostile, and will follow and abide by the instructions provided by the guidance documentation.

A.AUTHUSER

Authorized users possess the necessary authorization to access at least some of the information managed by the TOE and are expected to act in a cooperating manner in a benign environment.

A.TRAINEDUSER

Users are sufficiently trained and trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their user data.

3.3.1.3 Procedural aspects

A.DETECT

Any modification or corruption of security-enforcing or security-relevant files of the TOE, user or the underlying platform caused either intentionally or accidentally will be detected by an administrative user.

A.PEER.MGT

All remote trusted IT systems trusted by the TSF to provide TSF data or services to the TOE, or to support the TSF in the enforcement of security policy decisions are assumed to be under the same management control and operate under security policy constraints compatible with those of the TOE.

A.PEER.FUNC

All remote trusted IT systems trusted by the TSF to provide TSF data or services to the TOE, or to support the TSF in the enforcement of security policy decisions are assumed to correctly implement the functionality used by the TSF consistent with the assumptions defined for this functionality.

3.3.1.4 Connectivity aspects

A.CONNECT

All connections to and from remote trusted IT systems and between physically-separate parts of the TSF not protected by the TSF itself are physically or logically protected within the TOE environment to ensure the integrity and confidentiality of the data transmitted and to ensure the authenticity of the communication end points.

3.4 Organizational Security Policies

P.ACCOUNTABILITY

The users of the TOE shall be held accountable for their security-relevant actions within the TOE.

P.USER

Authority shall only be given to users who are trusted to perform the actions correctly.

P.ROLES

Administrative authority to TSF functionality shall be given to trusted personnel and be as restricted as possible supporting only the administrative duties the person has.

4 Security Objectives

This section defines the security objectives of the TSF and its supporting environment. Security objectives, categorized as either IT security objectives or non-IT security objectives, reflect the stated intent to counter identified threats, comply with any organizational security policies identified, or both. All of the identified threats and organizational policies are addressed under one of the following categories.

4.1 Objectives for the TOE

O.AUDITING

The TSF must be able to record defined security-relevant events (which usually include security-critical actions of users of the TOE). The TSF must protect this information and present it to authorized users if the audit trail is stored on the local system. The information recorded for security-relevant events must contain the time and date the event happened and, if possible, the identification of the user that caused the event, and must be in sufficient detail to help the authorized user detect attempted security violations or potential misconfiguration of the TOE security features that would leave the IT assets open to compromise.

O.DISCRETIONARY.ACCESS

The TSF must control access of subjects and/or users to named resources based on identity of the object. The TSF must allow authorized users to specify for each access mode which users/subjects are allowed to access a specific named object in that access mode.

O.NETWORK.FLOW

The TOE shall mediate network communication between an entity outside of the TOE and a recipient within the TOE in accordance with its network information flow security policy.

O.SUBJECT.COM

The TOE shall mediate any possible sharing of objects or resources between subjects acting with different subject security attributes in accordance with its discretionary access control policy.

O.I&A

The TOE must ensure that users have been successfully authenticated before allowing any action the TOE has defined to be provided to authenticated users only.

O.MANAGE

The TSF must provide all the functions and facilities necessary to support the authorized users that are responsible for the management of TOE security mechanisms, must allow restricting such management actions to dedicated users, and must ensure that only such authorized users are able to access management functionality.

O.TRUSTED_CHANNEL

The TSF must allow authorized users to remotely access the TOE using a cryptographically-protected network protocol that ensures integrity and confidentiality of the transported data and is able to authenticate the end points of the communication. Note that the same protocols may also be used in the case where the TSF is physically separated into multiple parts that must communicate securely with each other over untrusted network connections. The protocol must also prevent masquerading of the remote trusted IT system.

O.UNATTENDED_SESSION

The TOE must allow for the temporary suspension of a user's session allowing the continuation of such a suspended session and user related input and output only after the user has resumed the session by re-authenticating himself to the TSF.

4.2 Objectives for the Operational Environment

The following objectives are to be met by the operational environment of the TOE.

OE.ADMIN

Those responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains.

OE.REMOTE

If the TOE relies on remote trusted IT systems to support the enforcement of its policy, those systems provide the functions required by the TOE and are sufficiently protected from any attack that may cause those functions to provide false results.

OE.INFO_PROTECT

Those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner. In particular:

- All network and peripheral cabling must be approved for the transmittal of the most sensitive data held by the system. Such physical links are assumed to be adequately protected against threats to the confidentiality and integrity of the data transmitted.
- DAC protections on security-relevant files (such as audit trails and authentication databases) shall always be set up correctly.
- Users are authorized to access parts of the data managed by the TOE and are trained to exercise control over their own data.

OE.INSTALL

Those responsible for the TOE must establish and implement procedures to ensure that the hardware, software and firmware components that comprise the system are distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE.

OE.MAINTENANCE

Authorized users of the TOE must ensure that the comprehensive diagnostics facilities provided by the product are invoked at every scheduled preventative maintenance period.

OE.PHYSICAL

Those responsible for the TOE must ensure that those parts of the TOE critical to enforcement of the security policy are protected from physical attack that might compromise IT security objectives. The protection must be commensurate with the value of the IT assets protected by the TOE.

OE.RECOVER

Those responsible for the TOE must ensure that procedures and/or mechanisms are provided to assure that after system failure or other discontinuity, recovery without a protection (security) compromise is achieved.

OE.TRUSTED.IT.SYSTEM

The remote trusted IT systems implement the protocols and mechanisms required by the TSF to support the enforcement of the security policy.
These remote trusted IT systems are under the same management domain as the TOE, are managed based on the same rules and policies applicable to the TOE, and are physically and logically protected equivalent to the TOE.

4.3 Security Objectives Rationale

4.3.1 Security Objectives Coverage

The following table provides a mapping of TOE objectives to threats and policies, showing that each objective counters or enforces at least one threat or policy, respectively.

Objective	Threats / OSPs
O.AUDITING	P.ACCOUNTABILITY
O.DISCRETIONARY.ACCESS	T.ACCESS.TSFDATA T.ACCESS.USERDATA
O.NETWORK.FLOW	T.RESTRICT.NETTRAFFIC
O.SUBJECT.COM	T.ACCESS.TSFDATA T.ACCESS.USERDATA
O.I&A	T.IA.MASQUERADE T.IA.USER
O.MANAGE	T.ACCESS.TSFFUNC P.ACCOUNTABILITY P.USER

Objective	Threats / OSPs
	P.ROLES
O.TRUSTED_CHANNEL	T.ACCESS.USERDATA T.ACCESS.TSFDATA T.ACCESS.TSFFUNC T.ACCESS.COMM
O.UNATTENDED_SESSION	T.UNATTENDED_SESSION

Table 1: Coverage of security objectives for the TOE

The following table provides a mapping of the objectives for the Operational Environment to assumptions, threats and policies, showing that each objective holds, counters or enforces at least one assumption, threat or policy, respectively.

Objective	Assumptions / Threats / OSPs
OE.ADMIN	A.MANAGE A.AUTHUSER A.TRAINEDUSER P.ROLES
OE.REMOTE	A.CONNECT T.ACCESS.COMM
OE.INFO_PROTECT	A.PHYSICAL A.MANAGE A.AUTHUSER A.TRAINEDUSER P.USER
OE.INSTALL	A.MANAGE A.DETECT
OE.MAINTENANCE	A.DETECT

Objective	**Assumptions / Threats / OSPs**
OE.PHYSICAL	A.PHYSICAL
OE.RECOVER	A.MANAGE A.DETECT
OE.TRUSTED.IT.SYSTEM	A.PEER.MGT A.PEER.FUNC A.CONNECT

Table 2: Coverage of security objectives for the TOE environment

4.3.2 Security Objectives Sufficiency

The following rationale provides justification that the security objectives are suitable to counter each individual threat and that each security objective tracing back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat:

Threat	**Rationale for security objectives**
T.ACCESS.TSFDATA	The threat of accessing TSF data without proper authorization is mitigated by: • O.TRUSTED_CHANNEL requiring cryptographically-protected communication channels for data including TSF data controlled by the TOE in transit between trusted IT systems, • O.DISCRETIONARY.ACCESS requiring that data, including TSF data stored with the TOE, have discretionary access control protection, • O.SUBJECT.COM requiring the TSF to mediate communication between subjects.
T.ACCESS.USERDATA	The threat of accessing user data without proper authorization is mitigated by: • O.TRUSTED_CHANNEL requiring cryptographically-protected communication channels for data including user data controlled by the TOE in transit

Threat	Rationale for security objectives
	between trusted IT systems, • O.DISCRETIONARY.ACCESS requiring that data including user data stored with the TOE, have discretionary access control protection, • O.SUBJECT.COM requiring the TSF to mediate communication between subjects.
T.ACCESS.TSFFUNC	The threat of accessing TSF functions without proper authorization is mitigated by: • O.TRUSTED_CHANNEL requiring cryptographically-protected communication channels to limit which TSF functions are accessible to external entities, • O.MANAGE requiring that only authorized users utilize management TSF functions.
T.ACCESS.COMM	The threat of accessing a communication channel that establishes a trust relationship between the TOE and another remote trusted IT system is mitigated by: • O.TRUSTED_CHANNEL requiring that the TOE implements a trusted channel between itself and a remote trusted IT system protecting the user data and TSF data transferred over this channel from disclosure and undetected modification, • OE.REMOTE requiring that those systems providing the functions required by the TOE are sufficiently protected from any attack that may cause those functions to provide false results.
T.RESTRICT.NETTRAFFIC	The threat of accessing information or transmitting information to other recipients via network communication channels without authorization for this communication attempt is mitigated by: • O.NETWORK.FLOW requiring the TOE to mediate the communication between itself and remote entities in accordance with its security policy.
T.IA.MASQUERADE	The threat of masquerading as an authorized entity in order to gain unauthorized access to user data, TSF data or

Threat	Rationale for security objectives
	TOE resources is mitigated by: • O.I_A requiring that each entity interacting with the TOE is properly identified and authenticated before allowing any action the TOE is defined to provide to authenticated users only.
T.IA.USER	The threat of accessing user data, TSF data or TOE resources without being identified and authenticated is mitigated by: • O.I_A requiring that each entity interacting with the TOE is properly identified and authenticated before allowing any action the TOE has defined to provide to authenticated users only.
T.UNATTENDED_SESSION	The threat of an attack agent using an unattended session to gain access to protected functionality of the TSF, user data, or TSF data is mitigated: • O.UNATTENDED_SESSION requiring the capability that unattended sessions can be protected from use by unauthorized persons.

Table 3: TOE threats sufficiency

Security Policies	Security Objectives
P.ACCOUNTABILITY	The policy to hold users accountable for their security-relevant actions within the TOE is implemented by: ● O.AUDITING providing the TOE with audit functionality, ● O.MANAGE allowing the management of this function.
P.USER	The policy to match the trust given to a user and the actions the user is given authority to perform is implemented by: ● O.MANAGE allowing appropriately-authorized users to manage the TSF, ● OE.INFO_PROTECT, which requires that users are trusted to use the protection mechanisms of the TOE to protect their data.
P.ROLES	The policy to only give trusted users authority is implemented by: ● O.MANAGE allowing appropriately-authorized users to manage the TSF.

Security Policies	Security Objectives
	• OE.ADMIN, which requires that users responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains.

Table 4: Security policies sufficiency

The following rationale provides justification that the security objectives for the environment are suitable to cover each individual assumption, that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption, and that if all security objectives for the environment that trace back to an assumption are achieved, the intended usage is supported:

Assumption	Rationale for security objectives
A.PHYSICAL	The assumption on the IT environment to provide the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE is covered by: • OE.INFO_PROTECT requiring the approval of network and peripheral cabling, • OE.PHYSICAL requiring physical protection.
A.MANAGE	The assumptions on the TOE security functionality being managed by one or more competent trustworthy individuals is covered by: • OE.ADMIN requiring trustworthy personnel managing the TOE, • OE.INFO_PROTECT requiring personnel to ensure that information is protected in an appropriate manner, • OE.INSTALL requiring personnel to ensure that components that comprise the system are distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE, • OE.RECOVER requiring personnel to assure that after system failure or other discontinuity, recovery without a protection (security) compromise is achieved.
A.AUTHUSER	The assumption on authorized users to possess the necessary authorization to access at least some of the information managed

Assumption	Rationale for security objectives
	by the TOE and to act in a cooperating manner in a benign environment is covered by: • OE.ADMIN ensuring that those responsible for the TOE are competent and trustworthy individuals, capable of managing the TOE and the security of the information it contains, • OE.INFO_PROTECT requiring that DAC protections on security-relevant files (such as audit trails and authentication databases) shall always be set up correctly and that users are authorized to access parts of the data maintained by the TOE.
A.TRAINEDUSER	The assumptions on users to be sufficiently trained and trusted to accomplish some task or group of tasks within a secure IT environment by exercising complete control over their user data is covered by: • OE.ADMIN requiring competent personnel managing the TOE, • OE.INFO_PROTECT requiring that those responsible for the TOE must establish and implement procedures to ensure that information is protected in an appropriate manner and that users are trained to exercise control over their own data.
A.DETECT	The assumption that modification or corruption of security-enforcing or security-relevant files will be detected by an administrative user is covered by: • OE.INSTALL requiring an administrative user to ensure that the TOE is distributed, installed and configured in a secure manner supporting the security mechanisms provided by the TOE, • OE.MAINTENANCE requiring an administrative user to ensure that the diagnostics facilities are invoked at every scheduled preventative maintenance period, verifying the correct operation of the TOE, • OE.RECOVER requiring an administrative user to ensure that procedures and/or mechanisms are provided to assure that after system failure or other discontinuity, recovery without

Assumption	Rationale for security objectives
	a protection (security) compromise is achieved.
A.PEER.MGT	The assumption on all remote trusted IT systems to be under the same management control and operate under security policy constraints compatible with those of the TOE is covered by: • OE.TRUSTED.IT.SYSTEM requiring that these remote trusted IT systems are under the same management domain as the TOE, and are managed based on the same rules and policies applicable to the TOE.
A.PEER.FUNC	The assumption on all remote trusted IT systems to correctly implement the functionality used by the TSF consistent with the assumptions defined for this functionality is covered by: • OE.TRUSTED.IT.SYSTEM requiring that the remote trusted IT systems implement the protocols and mechanisms required by the TSF to support the enforcement of the security policy.
A.CONNECT	The assumption on all connections to and from remote trusted IT systems and between physically separate parts of the TSF not protected by the TSF itself are physically or logically protected is covered by: • OE.REMOTE requiring that remote trusted IT systems provide the functions required by the TOE and are sufficiently protected from any attack that may cause those functions to provide false results, • OE.TRUSTED.IT.SYSTEM demanding the physical and logical protection equivalent to the TOE.

Table 5: Assumptions Sufficiency

5 Extended Components Definition

[GPOSPP] defines two extended components:

- FIA_PK_EXT.1: Extended: Public key based authentication
- FMT_SMF_RMT.1: Remote Management Capabilities.

6 Security Requirements for the Operational Environment

Although CC Version 3.1 does not mandate the use of security requirements for the IT environment, it allows to define the security objectives for the IT environment to the level of detail useful for the understanding and evaluation of a TOE. In the case of z/OS the security functionality defined in chapter 7 of this Security Target depends on the supporting functionality defined in this section. The authors of this Security Target decided (also for compatibility with Security Targets used for previous versions of the TOE) to define this functionality using the structure of Security Functional Requirements.

There are several components in the IT environment that are used by the TOE to implement the security functional requirements. Those are:

- The instructions provided by the underlying processor (named z/Architecture)

- The “CP Assist for Cryptographic Functions” (CPACF). Although this feature is implemented as instructions of the processor and therefore is part of the z/Architecture, it has been decided by the authors of this Security Target to treat them separate from the other instructions. One reason is that some features of CPACF are available on selected processor types only. This is expressed in the SFRs related to CPACF.

- The “Crypto Express 3” (CEX3) coprocessor board. This board can be operated in two modes, coprocessor mode (CEX3C) and an accelerator mode (CEX3A). The CEX3 is a PCI board with its own processor and cryptographic coprocessors. This board provides a set of cryptographic functions broader than the CPACF.

  In CEX3C mode the coprocessor provides a separate, physically protected environment to store cryptographic keys and perform cryptographic operations.

  In CEX3A mode the board provides functions for fast long integer arithmetic that can be used for fast implementation of asymmetric cryptographic algorithms like RSA.

  This coprocessor is optional. The ICSF component of the TOE checks for the availability of one or more of those boards.

  On the z114 and z196 processors CEX3C offers ECC (elliptic curve cryptographic) functions not available on the z10.

- the CryptoExpress4S (CEX4S) coprocessor board. This board is available for the EC12 processor and can be operated in three modes, coprocessor mode (CEX4C), accelerator mode (CEX4A) and PKCS#11 support mode (CEX4P). The CEX4 is a PCI board with its own processor and cryptographic coprocessors. In coprocessor mode and in accelerator mode the board is functionally equivalent with the CEX3 coprocessor board on a z196 processor. In CEX4P mode the board supports a modified version of the PKCS#11 interface and allows for protected keys stored in PKCS#11 Token Key Data Sets (TKDS).

The CEX3 and CEX4S coprocessors are used in a way transparent to the user when he uses the ICSF component of the TOE. ICSF scans for the available cryptographic coprocessors and uses them accordingly. The security functional requirements listed here are related to the use of those coprocessors by the functions claimed in this Security Target that rely on cryptographic operations. While the coprocessors may implement more cryptographic functions than those claimed here, those are not used to support any of the claims made in chapter 7.1 of this Security Target.

When using the PKCS#11 functions of ICSF for RSA or Elliptic Curve cryptographic functions, ICSF checks for the availability of a CryptoExpress3 coprocessor in coprocessor mode, CryptoExpress4S coprocessor in coprocessor mode or CryproExpress4S in PKCS#11 support mode and will use the coprocessor when available and when it implements the basic cryptographic algorithm. If no coprocessor is available or if the cryptographic algorithm is not implemented by the coprocessor, ICSF uses s software implementation of those algorithms. When using the CCA functions, ICSF will always attempt to use a CryptoExpress3 or CryproExpress4S coprocessor and will fail the call if no CryptoExpress3 or CryproExpress4S coprocessor is available or if the required support for the function is not provided by the CryptoExpress3 or CryproExpress4S coprocessor. The SFRs for cryptographic operations have been refined with specification that allows the reader to determine when support for cryptographic algorithms is provided by CPACF, a CryptoExpress3 or CryptoExpress4S coprocessor.

While the functions of the coprocessors can only be called using ICSF, the processor instructions implemented by the CPACF are available for all programs. The claims made in this section are only for the use of those functions by the TSF. While this checks for the correct implementation of the basic cryptographic algorithms for those instructions, no claim can be made here for applications not part of the TSF that use those instructions. They may still use those instructions incorrectly or fail to protect cryptographic keys appropriately.

The other part of the IT environment where requirements are stated is the underlying abstract machine as implemented by the z/Architecture that has to provide the mechanism to protect the TSF and TSF data from unauthorized access and tampering. This is expressed with the following security functional requirement for the processor used to execute TOE software:

6.1 General security requirements for the abstract machine

6.1.1 Subset access control (FDP_ACC.1(E))

FDP_ACC.1.1 The abstract machine shall enforce the memory access control policy on instructions as subjects and memory locations and processor registers as objects.

6.1.2 Security-attribute-based access control (FDP_ACF.1(E))

FDP_ACF.1.1 The abstract machine shall enforce the memory access control policy to objects based on the processor state (problem or supervisor).

FDP_ACF.1.2 The abstract machine shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: access to memory locations and special registers is based on the processor state and the state of the memory management unit. Access to dedicated processor registers is allowed only if the processor is in supervisor state when the instruction accessing the register is executed.

FDP_ACF.1.3 The abstract machine shall explicitly authorize access of subjects to objects based on the following additional rules: some dedicated processor registers may be read but not modified when the instruction accessing the register is in problem mode.

FDP_ACF.1.4 The abstract machine shall explicitly deny access of subjects to objects based on the following rule: none.

Application note: *The precise definition of the objects and the rules for the access control policy differ slightly depending on the processor type. Although the underlying hardware / firmware that enforces this policy is part of the IT environment, it is analyzed and tested to provide the support required for the enforcement of the TOE's self-protection. The criteria for the analysis of the high-level design require the analysis of the underlying hardware and firmware and the security functional requirements stated here are taken as the basis for this analysis.*

6.1.3 Static attribute initialization (FMT_MSA.3(E))

FMT_MSA.3.1 The abstract machine shall enforce the memory access control policy to provide permissive default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The abstract machine shall allow the no role to specify alternative initial values to override the default values when an object or information is created.

Application note: *The “default” values in this case are seen as the values the processor has after startup. They have to be “permissive”, because the initialization routine needs to set up the memory management unit and the device register. With respect to the hardware, there is no “role” model implemented, but the access control policy is purely based on a single attribute (“user” or “supervisor” state) that can not be managed or assigned to a “user”. The attribute changes under well-defined conditions (when the processor encounters an exception an interrupt, or when a call gate for a higher ring of privilege is called). The security requirement FMT_MSA.1 was therefore not applicable because the security attribute cannot be “managed”. For this reason, there is also no security requirement FMT_SMR.1 included, because there are no ”roles” that need to be managed or assigned to “users”. The dependency of FMT_MSA.3 to FMT_MSA.1 and*

FMT_SMR.1 is therefore unresolved.

6.2 Security requirements for CPACF

The CP assist for cryptographic functions (CPACF) is a feature of the z/Architecture that provides instructions to perform cryptographic operations. Those instructions are part of the general instruction set of the processor and available to programs executing in any mode and with any PSW key. The instructions provide support for the basic cryptographic operations only. No support for key management, key protection or key generation is provided. This has to be performed by the software using the instructions. The instructions are specified in [ZARCH].

6.2.1 Cryptographic operation (DES) (FCS_COP.1(1E))

FCS_COP.1.1 The CPACF shall perform encryption and decryption in accordance with a specified cryptographic algorithm TDES and cryptographic key sizes 112 and 168 bit that meet the following: NIST Special Publication 800-67.

Application note: *This function is provided by the “Cipher Message” and “Cipher Message with Chaining” instructions. Function Code 1 specifies DES, function code 2 specifies two key TDES and function code 3 specifies 3 key TDES.*

6.2.2 Cryptographic operation (AES) (FCS_COP.1(2E))

FCS_COP.1.1 The CPACF shall perform encryption and decryption in accordance with a specified cryptographic algorithm AES in CFB, OFB, GCM, and CBC-CS modes and cryptographic key sizes 128, 192 or 256 bit that meet the following: FIPS 197, November 6, 2001 (AES), NIST Special Publication 800-38A, 2001 Edition (CFB and OFB modes of operation), Addendum to NIST SP 800-38A, October 2010 (CBC-CS mode of operation), NIST Special Publication 800-38D (GCM mode of operation).

Application note: *This function is provided by the “Cipher Message” and “Cipher Message with Chaining” instructions. Function Code 18 specifies AES.*

A z114 or z196 processor is required in order to use the CPACF for AES encryption/decryption using AES modes AES-CFB (Cipher Feedback), AES-OFB (Output Feedback), AES-GCM (Galois/Counter Mode), and AES-CBC-CS (Cipher Block Chaining – CipherText Stealing)

6.2.3 Cryptographic operation (SHA-1) (FCS_COP.1(3E))

FCS_COP.1.1 The CPACF shall perform hashing in accordance with a specified cryptographic algorithm SHA-1 and cryptographic key sizes not applicable that meet the following: FIPS 180-3 (October 2008)

Application note: *This function is provided by the "Compute intermediate message digest" and "Compute last message digest" instructions. Function Code 1 specifies SHA-1.*

6.2.4 Cryptographic operation (SHA-2) (FCS_COP.1(4E))

FCS_COP.1.1 The CPACF shall perform hashing in accordance with a specified cryptographic algorithms SHA-224, SHA-256, SHA-384, and SHA-512 and cryptographic key sizes not applicable that meet the following: FIPS 180-3 (October 2008).

Application note: *This function is provided by the "Compute intermediate message digest" and "Compute last message digest" instructions. Function Code 2 specifies SHA-256. Function Code 3 specifies SHA-512. With appropriate input values and post-processing, the SHA-256 processing can produce SHA-224 results, and the SHA-512 processing can produce SHA-384 results.*

6.3 Security requirements for CEX3 or CEX4S in CEX3C/CEX4C mode

CEX3 or CEX4S in CEX3C/CEX4C mode is a cryptographic coprocessor that provides the ability to perform both symmetric and asymmetric encryption. The coprocessor can be used via ICSF which uses the CCA functions to request services from the coprocessor. In the evaluated configuration only a subset of the functions provided by the coprocessor is used providing some of the basic encryption functions required by System SSL. The following SFRs therefore reflect only those functions and not the full set of capabilities of the CEX3C. TSF functions in the evaluated configuration may use the CEX3C/CEX4C for RSA key generation as well as RSA encryption and decryption. Both the clear key option (where the private key may be exported in clear from the coprocessor to the TOE) as well as the secure key option (where the private key is never exported in clear from ICSF) are supported. The secure key option is useful in environments where the risk of leakage of the private key from the TOE is viewed as unacceptable. This allows the TOE to securely use public key cryptography, since the CEX3C/CEX4C with its physical security protection provide an additional barrier for an attacker.

Although the CEX3C/CEX4C is also capable to perform symmetric encryption operations using DES and TDES, those functions are not used by the TSF. Performing DES or TDES symmetric encryption using the CPACF is significantly more efficient than using those functions on the CEX3C.

6.3.1 Cryptographic operation (RSA) (FCS_COP.1(7E))

FCS_COP.1.1 The CEX3C/CEX4C shall perform encryption and decryption in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 to 4096 bit that meet the following: RSA encryption and decryption operation as defined in PKCS#1 V2.1 (June 14, 2002) using either non-CRT or CRT key format as defined in section 3.2 of PKCS#1, Version 2.1 (June 14, 2002).

Application note: *This function is with both the clear key and the secure key option.*

6.3.2 Cryptographic key generation (Public/Private Keys) (FCS_CKM.1(2E))

6.4 Security requirements for CEX3 or CEX4S in CEX3A/CEX4A mode

The CEX3 or CEX4S in CEX3A/CEX4A mode is used as an accelerator card for asymmetric encryption/decryption operations. It provides the ability for fast RSA encryption and decryption operations. The coprocessor performs no key generation and does not provide any key storage capability.

6.4.1 Cryptographic support operation (FCS_COP.1(6E))

FCS_COP.1.1 The CEX3A/CEX4A shall perform long integer modulo operations as support for encryption and decryption in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 to 4096 bit that meet the following: none

6.5 Additional Security requirements for CEX3C/CEX4C on z114 and z196

Crypto Express 3 or Crypto Express4S operating in CEX3C/CEX4C mode on the z114 and z196 also provides ECC functionality not provided on the z10. The following SFRs reflect those CEX3/CEX4S functions used via ICSF in z/OS V2R1. For other functions refer to the prior SFRs for CryptoExpress3/CryptoExpress4S.

6.5.1 Cryptographic operation (ECC Digital Signature Generation) (FCS_COP.1(8E))

FCS_COP.1.1 The CEX3C/CEX4C when operating on the z114 and z196 processors shall perform digital signature generation with a specified cryptographic algorithm ECDSA that meet the following: curve parameter shall be one of 192, 224, 256, 384, or 521 bit using the curve parameter defined in FIPS 186-2, or Brainpool 160, 192, 224, 256, 320, 384, or 512 bit using the curve parameter defined in RFC5639 (non-twisted curves only); the hash function shall be one of SHA-1, SHA-224, SHA-256, SHA-384, or SHA-512 as defined in FIPS 180-3; the signature shall be generated in accordance with ANSI X9.62-2005, section 7.3.

Application note: *This function is with both the clear key and the secure key option.*

6.5.2 Cryptographic operation (ECC Digital Signature Verification) (FCS_COP.1(9E))

FCS_COP.1.1 The CEX3C/CEX4C when operating on the z114 and z196 processors shall perform digital signature generation with a specified cryptographic algorithm ECDSA that meet the following: curve parameter shall be one of 192, 224, 256, 384, or521 bit using the curve parameter defined in FIPS 186-2, or Brainpool 160, 192, 224, 256, 320, 384, or 512 bit using the curve parameter defined in RFC5639 (non-twisted curves only); the hash function shall be one of SHA-1, SHA-224, SHA-256, SHA-384, or SHA-512 as defined in FIPS 180-3; the signature shall be generated in accordance with ANSI X9.62-2005, section 7.4.

Application note: *This function is with both the clear key and the secure key option.*

6.5.3 Cryptographic key generation (ECDSA Public/Private Keys) (FCS_CKM.1(3E))

FCS_CKM.1.1 The CEX3C/CEX4C when operating on the z114 and z196 processors shall generate ECDSA public/private cryptographic keys in accordance with a specified cryptographic key generation algorithm vendor specific and specified curves one of NIST 192, 224, 256, 384, or521 bit using the curve parameter defined in FIPS 186-2, or Brainpool 160, 192, 224, 256, 320, 384, or 512 bit using the curve parameter defined in RFC5639 (non-twisted curves only) that meet the following: ANSI X9.62-2005, Appendix A.4.3.

Application note: *Keys are either generated as "clear keys" where the private key can be extracted in clear by the system using the CEX3C or they are generated as "secure keys" where the private key never leaves the coprocessor except in an encrypted form.*

7 Security Requirements

7.1 TOE Security Functional Requirements

7.1.1 Security audit (FAU)

7.1.1.1 Audit data generation (FAU_GEN.1)

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the not specified level of audit; and
c) all modifications to the set of events being audited;
d) all user authentication attempts;
e) all denied accesses to objects for which the access control policy defined in the OSPP base applies;
f) explicit modifications of access rights to objects covered by the access control policies; and
g) other specifically defined auditable events as defined in the table in FAU_GEN.1.2.

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity (if applicable), and outcome (success or failure) of the event; and
b) for all management SFRs included in the Security Target: the identity of the user that performed/attempted to perform the management operation, an identification of what was managed and the indication what the administrative user has changed as part of the management operation, and
c) For each audit event type, based on the auditable event definitions of the functional components included in the following table:

SFR	Events and Event specific information
FAU_SAR.1	Event: Any attempt to access the audit records • identity of the user attempting to access the audit records • success or failure
FAU_SEL.1	Event: Any attempt to modify the events to be audited • identity of the user attempting to modify the events to be audited • success or failure • in case of success: modification to the set of events to be audited

SFR	Events and Event specific information
FDP_ACF.1	Event: Any attempt to access an object protected by the SFP • identity of the user attempting to access an object protected by the SFP. Note: if the operation is attempted by a subject not operating on behalf of a user: identity of the subject • identity of the object the user attempts to access • attempted operation • Success or failure
FDP_IFF.1	Event: Denied information flow • identification of the network interface • reason for denying information flow
FIA_AFL.1	Event: Exceeding the limit of unsuccessful consecutive authentication attempts • user identity where the limit was exceeded
FIA_UAU.1(HU)	Event: Verification that a user has been successfully authenticated • user identity • indicator that the user has been successfully authenticated In the case the authentication is performed by the TOE, also the event of a failed authentication attempt needs to be auditable: • user identity provided • indicator that the authentication failed
FTA_SSL.1 FTA_SSL.2	Event: Re-authentication attempt to unlock a session • user identity • success or failure of re-authentication Event: Re-authentication attempt to unlock a session • user identity • success or failure of re-authentication
FTP_ITC.1	Event: Initialization of a trusted channel • identity of the communication partner • protocol used to establish the channel • success or failure of setting up the channel

Application note: *Each SMF record has a standard header that includes the ID of the job that caused the event. The ID of the job is related to the user ID under which the job has been started by SMF. Users accessing the HTTP server or LDAP server without authenticating themselves are audited with the user ID the server is configured to use for unauthenticated users. Also, for the HTTP server, authenticated users running under an administrator-configured ID for data access are audited with that administrator-configured ID. Also, in some cases of client authentication via TLS, when RACF certificate mapping rules are used to assign an administrator-specified ID rather than a unique ID, the audit records will contain the administrator-specified ID and the X500-based distinguished name from the client's digital certificate for accountability purposes.*

7.1.1.2 User identity association (FAU_GEN.2)

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

7.1.1.3 Audit review (FAU_SAR.1)

FAU_SAR.1.1 The TSF shall provide **administrative users that have been given read access to the SMF data sets containing the audit records** with the capability to read **all audit information** from the audit records.

FAU_SAR.1.2 The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

Application note: ***In this case, the term "administrative users" maps to the AUDITOR role of z/OS or a user with SPECIAL.***

7.1.1.4 Restricted audit review (FAU_SAR.2)

FAU_SAR.2.1 The TSF shall prohibit all users read access to the audit records, except those users that have been granted explicit read-access.

7.1.1.5 Selective audit (FAU_SEL.1)

FAU_SEL.1.1 The TSF shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes:

a) Type of audit event;
b) Subject or user identity;
c) Outcome (success or failure) of the audit event;
d) Named object identity;
e) Time and date

Application note: *RACF allows inclusion of auditable events based on the criteria defined above.*

7.1.1.6 Protected audit trail storage (FAU_STG.1)

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2 The TSF shall be able to **prevent** unauthorised modifications to the audit records in the audit trail.

Application note: *RACF data set protection needs to be used to protect the files containing audit records from unauthorized access and modification.*

7.1.1.7 Action in case of possible audit data loss (FAU_STG.3)

FAU_STG.3.1 The TSF shall **generate an alarm to the z/OS operator** if the audit trail exceeds **the capacity of the current SMF data set** or if any of the following:

- **reaching the predefined limit for the capacity of the SMF data sets**

is detected that may result in a loss of audit records.

Application note: *The TOE switches to the next available SMF data set. Saving the SMF data set that got filled up can be done automatically or manually.*

7.1.1.8 Prevention of audit data loss (FAU_STG.4)

FAU_STG.4.1 The TSF shall **prevent audited events, except those taken by the authorized administrator** and **inform a z/OS operator** if the audit trail is full.

7.1.2 User data protection (FDP)

7.1.2.1 Subset access control: z/OS objects (FDP_ACC.1)

FDP_ACC.1.1 The TSF shall enforce the **Discretionary Object Access Control Policy** on

a) **jobs, started tasks, UNIX processes (whether initiated by rlogin, telnet, HTTP, FTP, or other method), and TSO sessions acting on behalf of users**
b) **Storage Objects of the following type**

MVS objects:

data sets, terminals, devices, volumes, consoles, operator commands, programs, System Logger objects, Communications Server Policy Agent data, TCP/IP connections;

UNIX objects:

z/OS UNIX file system objects (regular files, directories and symbolic links, character special files, UNIX domain sockets and named pipes (FIFOs);

UNIX IPC objects:

shared memory segments, message queues, semaphores;

LDAP objects:

LDAP LDBM objects;

c) **Operations: all operations among subjects and objects covered by the Discretionary Object Access Control Policy.**

7.1.2.2 Security attribute based access control: MVS (FDP_ACF.1(MVS))

FDP_ACF.1.1 The TSF shall enforce the **Discretionary Object Access Control Policy** to *MVS* objects based on the following:

a) **The user identity and group memberships associated with a subject; and**
b) **The following access control attributes associated with an object:**
   i. **an access control list capable of defining the access rights read, update, execute, alter, control, and none for individual users and groups**
   ii. **a default access right (defined by the UACC attribute in the resource profile) for users who are not addressed in the access control list**
   iii. **an entry for the resource containing the object in the global access checking table.**

Application note: *The semantics of* "*read*", "*update*", "*execute*", "*alter*", *and* "*control*" *are defined by the resource manager and follow the intuitive semantics of those terms. In the case of the Communications Server Policy Agent data, the resource manager implements only* "*read*" *access to this data.*

Any access right hierarchical to read for the profile protecting this data will therefore still result only in read access to this data. In the case of Operator Commands, the semantics of the different access rights is defined as part of the description of the command.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: **a subject has the requested type of access to a protected resource, if the resource is protected by RACF and**

a) **if access is allowed by global access checking (Note: does not apply for user with the RESTRICTED attribute; does not apply to checks performed by RACROUTE REQUEST=FASTAUTH)**

or, if a) did not grant access,

b) **if the resource is a tape or DASD data set and the high-level qualifier of the data set name is identical to the user ID**

if b) did not grant access,

c) if the requested type of access is allowed by an access control list (ACL) entry for this particular user (Note: does not apply to checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option)
if c) neither granted nor denied access then continue with d) Otherwise, if c) denied access, continue with g),

d) if the requested type of access is allowed by an ACL entry for the group the user belongs to. If list-of-groups processing is not in effect, this rule is evaluated only for the current connect group. Otherwise, this rule is evaluated for all groups to which the user is connected. (Note: does not apply to checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option)
if no entries in d) granted access, and no entries in d) denied access, then continue with e). Otherwise, if at least one entry in d) denied access, then continue with g),

e) if the user does not have the RESTRICTED attribute and the requested type of access is granted by the universal access authority (UACC) in the profile protecting the resource or granted by an ACL with ID(*)(Note: does not apply to checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option)
if e) did not grant access,

f) if the user has the OPERATIONS role or the group-OPERATIONS role (for a group to which the user is connected and the resource is within the group's scope) and OPERATIONS access is allowed for the class
if f) did not grant access,

g) if the user has an entry in the conditional access list for the profile that allows the requested type of access and the user meets the condition defined in this conditional access list entry (Note: for checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option, only conditional access list entries specifying WHEN(CRITERIA(SQLROLE…)) will apply)
or, if g) did not grant access,

h) if the user is a member of a group that has an entry in the conditional access list for the profile that allows the requested type of access and the user meets the condition defined in this conditional access list entry. If list-of-groups processing is not in effect, this rule is evaluated only for the current connect group. Otherwise, this rule is evaluated for all groups to which the user is connected. (Note: for checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option, only conditional access list entries specifying WHEN(CRITERIA(SQLROLE…)) will apply)

or, if h) did not grant access,

i) **if a conditional access list entry for ID(*) exists with requested type of access, the user does not have the RESTRICTED attribute set and the user satisfies the condition of the conditional access list entry. (Note: for checks performed by RACROUTE REQUEST=FASTAUTH with the AUTHCHKS=CRITONLY option, only conditional access list entries specifying WHEN(CRITERIA(SQLROLE…)) will apply).**

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules:

a) **the subject has the "trusted" or "privileged" attribute (which in the evaluated configuration is reserved for started tasks that are part of the TSF)**
b) **the subject is a trusted subject and has specified a nested ACEE in its call to RACF with a second user ID. In this case access is allowed if either the primary user ID specified in the first ACEE or the secondary user ID specified in the nested ACEE has the requested access right to the object and the object has been designated as eligible for nested ACEE processing and the authorization check is made using RACROUTE REQUEST=FASTAUTH.**
c) **when "program control" is activated (using the WHEN(PROGRAM) option in the SETROPTS command) and the program is protected by a profile in the PROGRAM class and the user has at least EXECUTE access to this profile, the user can execute the program in a clean z/OS environment not "contaminated" by any untrusted program. If the user has at least READ access then untrusted programs may also be used by the user.**
d) **when "program control" is activated and "PADCHK" has been defined in the profile for a program, a user may access a data set via PADS if the program that attempts the access or a higher program in the execution hierarchy is allowed to access the file in the intended mode by the conditional access list for the data set and all other active programs not from the link pack area that have been defined using the WHEN PROGRAM operand with “PADCHK” are included in the conditional access list of the data set. While a data set is open using PADS, for any new program defined with PADCHK and started in this situation in the same environment, the TOE checks that the new program is also in the conditional access list of that data set.**

Application note: *The term “trusted” in this sense means “defined to RACF via profiles in the PROGRAM class, or resident in the system link pack area.*

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **data sets that are not protected by a discrete or generic profile can not be accessed by users except for users with the SPECIAL role**.

7.1.2.3 Security attribute based access control: UNIX (FDP_ACF.1(UNIX))

FDP_ACF.1.1 The TSF shall enforce the **Discretionary Object Access Control Policy** to *UNIX* objects based on the following:

a) **The z/OS UNIX user identity and group membership(s) associated with a subject; and**
b) **The following access control attributes associated with an object: permission bits and (for file system objects) an access control list capable of defining access rights read, write, execute, or search. Default access rights are defined by a system management attribute.**

Access rights for file system objects are:

a) **read**
b) **write**
c) **execute (ordinary files)**
d) **search (directories)**

Access is defined by POSIX ACLs and permission bits. ACLs are evaluated only when the FSSEC class is active in RACF.

Users who have the AUDITOR attribute have implicit SEARCH and READ access for directories, without needing explicit permission via the permission bits or ACLs.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

If the FSACCESS class is active and SETROPTS RACLISTed, and if the subject is accessing the root directory in a mounted zFS file system (but not the system root directory), then, if the MVS data set name of the file system container is protected by a profile in the FSACCESS class the subject must have UPDATE access to that FSACCESS profile or the access request will fail.

Otherwise a subject must have search permission for every element of the path name and the requested access for the object. A subject has a specific type access to an object if:

a) **the user has the AUDITOR attribute, the requested type of access is READ or Search, and the object is a directory.**
b) **the effective user ID is 0 and the requested type of access is not execute. If this is the case, access is granted. If the**

effective user ID is 0, the requested type of access is execute, there is no permission bit, and there is no ACL that provides execute access to any user, access is denied.

c) **the effective user ID is the one of the file owner and has been granted access according to the owner permission bits, access is granted.**
d) **the FSSEC class is active in RACF and an ACL exists within the set of ACLs for the file that grants the required type of access to the requesting user, access is granted.**
e) **the effective user ID is the one of the owner of the file, the algorithm continues with step j.**
f) **the effective group ID (GID) or any of the user's supplemental GIDs matches the group of the file and has the requested type of access defined in the group permission bits, access is granted.**
g) **the effective GID or any of the user's supplemental GIDs has an ACL defined for the file that allows the requested type of access, access is granted.**
h) **the requested type of access is defined in the "other" permission bits and the user does not have the RESTRICTED attribute defined in his profile, access is granted.**
i) **the user has the RESTRICTED attribute defined and has the requested type of access defined in the RESTRICTED.FILESYS.ACCESS resource profile and the ACLs associated with this profile, access is granted.**
j) **the user has the RESTRICTED attribute defined, the RESTRICTED.FILESYS.ACCESS profile is not defined in RACF, and the requested type of access is allowed according to the "other" permission bits, access is granted.**
k) **the UNIXPRIV class is active and RACLISTed, and if the SUPERUSER.FILESYS.ACLOVERRIDE resource is protected by a profile in the UNIXPRIV class, then the user must have the correct access level as documented for the ck_access (IRRSKA00) callable service in z/OS Security Server: RACF Callable Services. If the profile exists, it determines whether file access is granted or denied.**
l) **if this step of the algorithm is reached and no decision for granting or denying access has been made, access is denied.**

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules:

a) **the UNIXPRIV class is active in RACF, the access was denied by an ACL entry but the user has the requested type of access to the file defined as access to the SUPERUSER.FILESYS.ACLOVERRIDE profile**

or

b) **the UNIXPRIV class is active in RACF, the access was denied by the permission bits, the**

SUPERUSER.FILESYS.ACLOVERRIDE profile is not defined in the UNIXPRIV class but the user has the requested type of access to the SUPERUSER.FILESYS profile, that is, if the user wants to read the file, the user must have read access to the profile, if the user wants to read and write the file, the user must have write access to the profile, if the user wants to update any directory, the user must have control access.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none**.

7.1.2.4 Security attribute based access control: LDAP (FDP_ACF.1(LDAP))

FDP_ACF.1.1 The TSF shall enforce the **Discretionary Object Access Control Policy** to *LDAP* objects based on the following:

a) **The z/OS LDAP Bind DN identity associated with a subject, together with the subject's LDAP groups derived during bind processing; and**
b) **The entryOwner attribute that applies to the object, and**
c) **LDAP ACLs or ACL Filters that determine whether the access is allowed or not.**

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

a) **The owner of the LDAP object as well as the LDAP administrator (identified by the administrator DN) are always allowed full access to the object**
b) **In the case the z/OS LDAP user identity is neither the owner nor the LDAP administrator access is determined by the LDAP ACL associated with the LDAP object. This ACL is determined as follows:**
    1. **If the LDAP object has an explicit AclEntry, the ACLs in this entry are used to determine access**
    2. **If the LDAP object has no explicit AclEntry, the next entry found when traversing up the directory tree that has an explict AclEntry and has the AclPropagate attribute set to TRUE, defines the AclEntry used to determine access**
    3. **If no LDAP object with an explicit AclEntry is found by the above two steps, the default ACL is used to determine access**
c) **ACLs in the AclEntry are evaluated as follows to determine access:**
    1. **if there is a specific value for the DN of the LDAP user, the LDAP user gets those permissions only**
    2. **else if there is a cn=this value and the DN of the**

LDAP user is the distinguished name of the entry, the LDAP user gets those permissions only

3. **else if there are one or more group values that the LDAP user is a member of, the LDAP user gets the union of the permissions for those groups**
4. **else if there is a cn=authenticated value and the LDAP user is authenticated to the directory with an LDAP bind operation, the LDAP user gets those permissions only**
5. **else if there is a cn=anybody value, the LDAP user gets those permissions only**
6. **otherwise the LDAP user gets no permissions**

d) **ACLs in the AclEntry may specify "grant" or "deny" permissions for the object as a whole, for specific named attributes within the object, or for attribute classes within the object. The LDAP server will process the ACLs in a precedence order to determine which ACL best applies to the user's request. The higher priority of the following list have preference over lower priorities (listed from highest to lowest):**
   1. **attribute-level deny permissions**
   2. **attribute-level grant permissions**
   3. **access-class deny permissions**
   4. **access-class grant permissions**

After a user's base access has been determined, it may be modified by Filter ACL entries. Filter ACLs can set permissions based on any of the following:

a) **bind DN**

b) **alternate DNs**

c) **pseudo DNs**

d) **groups that the bind or alternate DNs belong to**

e) **IP address of the client connection**

f) **time of day that directory entry was accessed**

g) **day of week that directory entry was accessed**

h) **the bind mechanism used**

i) **whether or not bind encryption was used**

Filters support wildcards. Filters have the same syntax support as LDAP search filters, where logical rules can be specified, such as "&"(and), "|"(or), and "!"(not).

To allow flexibility, the aclEntry filtering mechanism also supports three operation values that allow administrators to specify the way in which filtered ACLs will take effect:

a) **replace - the base effective ACL is replaced by the filtered ACLs.**

   If administrators want a client from a given IP address to only have a specific set of permissions, they would use replace.

b) **union - the base effective ACL is unioned with the filtered ACLs, resulting in a new effective ACL. This would be used to expand permissions.**

   If administrators want a client from a given IP address to have a specific set of permissions, at a minimum, they would use union.

c) **intersect - the base effective ACL is intersected with the filtered ACLs. This would be used to reduce permissions.**

   If administrators want a client from a given IP address to have a specific set of permissions, if and only if they already have the permissions, they would use intersect.

As with the operator precedence described above, filter ACL entries specifying "deny" take precedence over entries specifying "grant".

Application note: *The owner of an LDAP object is determined by the entryOwner attribute, or (if this does not exist for the LDAP object) by the ownerSource attribute. The ownerSource attribute is not modifiable and is managed by the TOE. It indicates the DN of the entry that holds the entryOwner attribute that applies to this object. This is the first entry encountered, while traveling up the directory tree from the object toward the root, which has an entryOwner attribute and has the ownerPropagate attribute set to TRUE.*

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **in case of an anonymous bind the ACL checking is performed against the cn=Anybody group and the user gets the access rights assigned to this group**.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none**.

7.1.2.5 Security attribute based access control: UNIX IPC (FDP_ACF.1(UNIX IPC))

FDP_ACF.1.1 The TSF shall enforce the **Discretionary Object Access Control Policy** to *UNIX IPC* objects based on the following:

a) **The z/OS UNIX user identity and group membership(s)**

associated with a subject; and

b) **The following access control attributes associated with an object: permission bits. Default access rights are defined by a system management attribute.**

Access rights for z/OS UNIX IPC objects are:

a) **read**

b) **write**

Access is defined by permission bits only.

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

Access permissions are defined by permission bits of the IPC object only. IPC objects don't have ACLs associated with them. The process creating the object defines the creator, owner, and group based on the user ID of the current process. Access of a process to an IPC object is allowed if at least one of the following conditions is true:

a) **the effective UID of the current process is equal to the UID of the IPC object creator or owner and the "owner" permission bit for the requested type of access is set or,**

b) **the user is neither the owner nor the creator of the IPC object and the effective UID of the current process is not equal to the UID of the IPC object creator or owner and the effective GID of the current process or any supplementary z/OS UNIX GIDs the user is a member of is equal to the GID of the IPC object and the "group" permission bit for the requested type of access is set or,**

c) **the "other" permission bit for the requested type of access is set for users who do not satisfy one of the first two conditions.**

FDP_ACF.1.3 The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **none**.

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none**.

7.1.2.6 Subset information flow control: network (FDP_IFC.1)

FDP_IFC.1.1 The TSF shall enforce the Network Information Flow Control Policy on

a) Originating entities:
   i. unauthenticated external IT entities that send network data

to a network interface of the TOE;
   - ii. subjects within the TOE that send network data to unauthenticated external IT entities via a network interface of the TOE;

b) Information:
   - i. Network data received by the TOE from an external entity;
   - ii. Network data provided to the TOE by a subject executing on the TOE intended to be sent to an external IT entity via a network interface controlled by the TOE;
   - iii. **none**;

c) Operations:
   - i. Receiving network data from an unauthenticated external IT entity;
   - ii. Sending network data to an unauthenticated IT entity by a subject within the TOE;

Application note: *This requirement covers IPv4 and IPv6 traffic.*

7.1.2.7 Simple security attributes (FDP_IFF.1)

FDP_IFF.1.1 The TSF shall enforce the Network Information Flow Control Policy based on the following types of subject and information security attributes:

Object security attribute: the logical or physical network interface through which the network data from an external IT entity entered the TOE or is intended to be sent out;

a) TCP/IP information security attributes:
   - **i. Source and destination IP address,**
   - **ii. Source and destination TCP port number,**
   - **iii. Source and destination UDP port number,**
   - **iv. Network protocol of IP, TCP, UDP, ICMP,**
   - **v. TCP header flags of SYN, ACK.**

FDP_IFF.1.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

for both receiving network data from an external IT entity and sending network data by a subject within the TOE to an external IT entity:

a) if the set of rules defined in accordance with the security attributes defined in FDP_IFF.1.3 define that the network data is discarded the network data shall not be delivered by the TOE to the intended recipient;

b) if the set of rules defined in accordance with the security attributes defined in FDP_IFF.1.3 define that the network data is to be delivered unaltered the network data shall be delivered unaltered by the TOE to the intended recipient;

c) if the set of rules defined in accordance with the security attributes defined in FDP_IFF.1.3 define another action to be taken than discarding the network data or delivering the data unaltered to the intended recipient, the TOE shall perform this action.

FDP_IFF.1.3 The TSF shall enforce the following rules consisting of an identification when the rule fires and an action to be taken when the rule fires:

Identification of network data using one or more of the following concepts:

a) Information security attribute matching based on the following security attributes **IP stack for which the rules apply, IP source address, IP protocol, port, ICMP type or code, direction of flow.;**

b) **matching rules as defined by the IPFilterRule statements**, **and no other matching concepts**;

Performing one or more of the following actions:

a) Discard the network data **without any further processing**;

b) Allow the network data to be delivered unaltered by the TOE to the intended recipient;

c) **Allow the network data to be processed after applying IPSec protection according to local IPSec policy.**

FDP_IFF.1.4 The TSF shall explicitly authorize an information flow based on the following rules: **no additional rules.**

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: **no additional rules.**

Application note: *This requirement covers IPv4 and IPv6 traffic.*

7.1.2.8 Full residual information protection (FDP_RIP.2)

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the **allocation of the resource to** all objects, subjects or subject/object related TSF data before the resource is assigned or made available to another subject or user.

7.1.3 Identification and authentication (FIA)

7.1.3.1 Authentication failure handling (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when an administrator-configurable an administrator-configurable positive integer within a range of acceptable values of unsuccessful authentication attempts for the authentication method*s* password based authentication, **password phrases, and RACF PassTickets** occur related to **all authentication events using these authentication methods**.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall **set the user status to REVOKE**.

7.1.3.2 User attribute definition: human users (FIA_ATD.1)

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual human users:

a) User identifier;
b) Group memberships;
c) User password *or password phrase*;
d) Security roles;
e) default access rights for objects created by the user (UACC);
f) classes in which the user can define profiles (CLAUTH);
g) indicator that global access checking, the ID(*) entry on the access list, and the UACC will not be used to allow this user access to a protected resource (RESTRICTED);
h) z/OS UNIX UID (for users also defined to UNIX System Services);
i) z/OS UNIX group memberships;
j) Kerberos principal name (for users defined to the z/OS Network Authentication Service and for foreign Kerberos principals that are defined to a Kerberos realm that has a cross realm trust relationship with the z/OS Network Authentication Service);
k) Kerberos ticket maximum lifespan for users defined to the z/OS Network Authentication Service;
l) indicator of the encryption algorithm used by the z/OS Network Authentication Service;
m) X.509v3 certificate(s)
n) z/OS LDAP user (bind) identifier (for users also defined to LDAP LDBM); and
o) **z/OS LDAP group memberships (for users also defined to LDAP LDBM)**.

Application note: *Attributes such as SPECIAL, GROUP-SPECIAL, AUDITOR, GROUP-AUDITOR, and OPERATIONS designate roles in the model of this Security Target and are therefore further explained in the role model in FMT_SMR.1*

7.1.3.3 Timing of authentication (FIA_UAU.1(RITE))

FIA_UAU.1.1 The TSF shall allow

a) the information flow covered by the Network Information Flow Control Policy (for remote IT entities);
b) administrator-specified anonymous access to specific data via HTTP, FTP, or LDAP

on behalf of the remote IT entity to be performed before the remote IT entity is authenticated.

FIA_UAU.1.2 The TSF shall require each remote *IT* entity to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that remote IT entity.

Application note:

z/OS allows an authorized administrator to configure the HTTP server, the FTP server, or the LDAP server to allow “anonymous” access to selected data. Such access occurs for HTTP or FTP using an administrator-specified user ID, which also is a form of pseudo-user, and the administrator controls which data that user has access to, and whether such anonymous access is enabled or not. For LDAP, the administrator can control whether a particular LDAP LDBM server allows unauthenticated access or not, and can further control which data in the LDBM database the unauthenticated user can access. For LDAP, the default is to allow anonymous access, and so the administrator who chooses to enable LDAP access must usually disable the default anonymous access.

7.1.3.4 Timing of authentication (FIA_UAU.1(HU))

FIA_UAU.1.1 The TSF shall allow

a) all functions allowed to be performed by the individual pseudo-user assigned by the authorized administrator for started procedures (started tasks);

on behalf of the *pseudo* user to be performed before the user is authenticated.

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application note: *In z/OS, predefined jobs known as started procedures (or started tasks) may be started automatically, or by an operator who has the required privileges. Those started tasks operate under a pseudo-user-ID assigned to them by the system* *administrator when the started task job was created and stored in a protected data set. z/OS allows the definition of protected user IDs for this purpose. Protected user IDs don't have a password or password phrase associated with them and cannot be used to log in under TSO or UNIX. They need to be defined in RACF and they are bound by the same RACF access control rules as a normal user. Activities performed by such a started task are accounted to the pseudo-user-ID assigned to them and not with the ID of the operator that started those tasks (because, in most cases, the operator would not know what those started tasks are doing and the operator would not be allowed to access the resources that the started tasks needs access to). No "user authentication" is performed for started tasks. Instead, they can only be started from predefined libraries. Write access to those libraries needs to be restricted to system administrators.*

This concept does not allow an unauthenticated user to execute any program or command on the TOE. Instead this concept allows an authenticated and properly authorized user to start specific tasks that have previously been defined by an authorized administrator and that operate under a pseudo–user-ID. The user that started this task usually has no influence on what the task is doing. The fact that he started the Started Procedure is auditable which ensures that the individual accountability for starting the started procedure is given. The ID of the pseudo-user listed in the JOB statement of the started procedure is not authenticated.

7.1.3.5 Multiple authentication mechanisms (FIA_UAU.5)

FIA_UAU.5.1 The TSF shall provide the following authentication mechanisms:

a) Authentication based on username and password (for human users);
b) Authentication based on username and password phrases;
c) Authentication based on software token verification data (digital certificates, Kerberos tickets);
d) Authentication based on RACF PassTickets

to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the following rules:

a) Authentication based on username and password is performed for TOE-originated requests and with credentials stored by the TSF by default unless another authentication method defined for human users in FIA_UAU.5.1 b is selected;
b) Users with expired passwords are **locked out until their password is reset by an administrator**;
c) Users with the SPECIAL attribute that have an expired password or which are locked due to too many consecutive failed authentication attempts may still logon if the system operator allows this (to avoid a denial-of-service attack for administrative tasks);
d) Authentication based on username and password phrases is performed for TOE-originated requests and with credentials

stored by the TSF if the length of the supplied credential in the field reserved for passwords exceeds the maximum size of a password and if the server within the TSF requesting the user to be authenticated supports password phrases;

e) **Authentication based on software token verification data is performed for TOE-originated requests and with credentials stored by the TSF when the server within the TSF requesting the user to be authenticated supports the specific software token verification data (digital certificates or Kerberos tickets);**

f) **Authentication based on RACF PassTickets is performed for TOE-originated requests and with credentials stored by the TSF when the server within the TSF requesting the user to be authenticated supports RACF PassTickets;**

g) **TSF servers that perform user authentication (by calling RACF) accept any of the above listed authentication mechanisms provided they are configured for this mechanism (in the case of digital certificates or Kerberos tickets) and the mechanism is also supported for the user that attempts to authenticate. Attempts to authenticate to such an application using a mechanism the application is not configured for or where the mechanism is not supported for the user will result in the rejection of the authentication attempt.**

7.1.3.6 Protected authentication feedback (FIA_UAU.7)

FIA_UAU.7.1 The TSF shall provide only obscured feedback to the user while the authentication is in progress.

Application note: *When entered during TSO LOGON the user has the option to use those TSF functions in a way that prohibits passwords/phrases from being displayed. Passwords for Operator LOGON are not displayed. Passwords a user enters via a JCL JOB statement will be suppressed in any output of the JCL statements to prohibit the password from being obtained by anybody reading the output.*

For authentication performed by servers where the userid and password/phrase is transferred over the network, the servers ensure that no feedback is provided as long as the authentication is in progress. For protocols where the server can request the client to suppress the display of characters entered by the user, such a request is sent before passwords/phrases are requested to be entered by the user. This is done for telnet, TN3270, and the r-commands. This still requires that the clients used implement those controls (e. g. switching to no-echo mode) correctly. In the case of FTP, SSH, Kerberos, and LDAP the protocols do not have any control statements that can be sent to the client to suppress the display of characters when a user enters a password/phrase. In those cases the TSF have no control how the client obtains a user's password/phrase and just ensures that no password/phrase related information is sent back to the client.

In all cases where clients operating as regular user programs are used it is outside of the control of the TSF how those clients handle the password/phrase. Where those interfaces are defined as part of the communication protocol, the TSF interfaces of the servers just *ensure that the clients get the required information to suppress displaying passwords/phrases.*

Client programs supplied by the TOE that operate as regular user programs (su, kinit, kpasswd, ssh, etc.) do not echo the password/phrase, but as they are user programs they are not part of the TSF.

Note that in the case of authentication via digital certificates, Kerberos tickets or PassTickets, no feedback is provided during the time authentication is in progress.

7.1.3.7 Timing of identification (FIA_UID.1)

FIA_UID.1.1 The TSF shall allow **access to the HTTP server, FTP server, or LDAP server (restricted to the functions and resources accessible to the pseudo user the administrator assigned for that purpose)** on behalf of the user to be performed before the user is identified.

FIA_UID.1.2 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Application note: *The pseudo-user of a started task is identified within the JOB statement of the JCL defining the started task. Users who start a started task (which will not be executed with the ID of the user that started the task) need to be identified and authenticated before they can perform this action. The FTP, LDAP, and HTTP server will assign an administrator defined ID of a pseudo user to users that connect to those servers without authenticating themselves. In this case all security related decisions are based on this ID.*

7.1.3.8 User-subject binding (FIA_USB.1)

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that human user:

a) The *RACF or LDAP* user identity *that is associated with auditable events*;
b) The *RACF or LDAP or UNIX* user security attributes that are used to enforce the Discretionary Object Access Control Policy;
c) The software token that can be used for subsequent identification and authentication with the TSF or other remote IT systems;
d) Active roles;
e) Active groups;
f) the RACF attributes/roles SPECIAL, group-SPECIAL, AUDITOR, group-AUDITOR, CLAUTH, OPERATIONS, and group-OPERATIONS.

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users:

a) **A started task executes with the user ID defined in the started class or started procedures table defining the started task.**
b) **A user that connects to the HTTP server or LDAP server without authenticating will be bound to the identity the installation has assigned for the unauthenticated user of the server, and limited to accessing data that user is allowed to access, unless and until the user is successfully identified and authenticated using his own authentication information.**

The TSF shall enforce the following rules for the assignment of subject security attributes not derived from user security attributes when a subject is created:

a) **When executing a program from a file with the set-user-ID-on-execution bit (S_ISUID) set, the subject's effective UID is set to the owner ID of the file being executed; when executing the program from a file with the set-group-ID-on-execution bit (S_ISGID) set, the subject's effective GID is set to the group ID of the file being executed;**
b) **The Port of Entry (POE) is set to TERMINAL (when the user has started his session from a terminal), CONSOLE (when the user has started his session from z console), JESINPUT (when the subject is started as a job entered via JES), or SERVAUTH (when the user has started his session via a network server application). Additional data is associated with this security attribute depending on the input class (e. g. the terminal ID in case of a terminal or the network zone in case the Port of Entry type is SERVAUTH).**

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users:

a) **A z/OS administrator may define specific z/OS applications to execute with an administrator defined user ID.**
b) **A z/OS administrator may use the SURROGAT authority mechanism to allow a user to switch his identify to another defined user (e. g. submitting jobs or changing the ID with the su command in the z/OS UNIX System Services environment) without specifying the password/phrase for this user.**

In z/OS UNIX, the following additional rules apply:

a) **The su command provides the ability to create a new session with a new set of credentials (to be inherited by subjects created within this session). The credentials are set to the UID (RUID and EUID), GID (RGID and EGID), and supplementary groups of the user requested. The user issuing the su command must have the authority to use this command, have the authority to switch to the specified UID and either authenticates properly for this UID with the**

password/phrase , has the SURROGAT authority for the new UID or has BPX.SUPERUSER authority allowing him to switch to UID 0 without supplying a password/phrase.

b) **If the BPX.DAEMON profile exists in the FACILITY class of RACF, a user with UID 0 needs to have authority other than NONE to this profile to change his UID using the setuid or seteuid system calls.**

c) **If the Ported Tools for z/OS Supplementary Toolkit Feature is installed, then if allowed by the configuration specified by the security administrator in the sudoers file, a user can utilize the sudo command to run specified UNIX commands, with specified operands, under the authority of a different user IDs.**

7.1.3.9 Extended: Public key based authentication (FIA_PK_EXT.1(IPSEC-TLS))

FIA_PK_EXT.1.1 The TSF shall use **X.509v3 certificates** as defined by **RFC5280** to support authentication for **IPSec, TLS** connections.

FIA_PK_EXT.1.2 The TSF shall store and protect certificates and/or public keys from unauthorized deletion and modification.

7.1.3.10 Extended: Public key based authentication (FIA_PK_EXT.1(SSH))

FIA_PK_EXT.1.1 The TSF shall use **public key cryptography** as defined by **RFC4252** to support authentication for **SSH** connections.

FIA_PK_EXT.1.2 The TSF shall store and protect ~~certificates and/or~~ public keys from unauthorized deletion and modification.

7.1.4 Security management (FMT)

7.1.4.1 Management of security functions behaviour (FMT_MOF.1)

FMT_MOF.1.1 The TSF shall restrict the ability to modify the behaviour of the functions password based user authentication to **users with the SPECIAL attribute** by allowing those users to specify rules for acceptable passwords that:

a) allow for uppercase characters, lowercase characters, digits, and

© Copyright IBM Corp. 2015

special characters to be used in passwords

b) define a minimum password length of 8 characters or more (at least up to 15 characters)

c) define that passwords must have at least one digit and one special character

d) reject passwords used by the same user before up to a history of at least 6 passwords

7.1.4.2 Management of object security attributes (FMT_MSA.1)

FMT_MSA.1.1 The TSF shall enforce the **Discretionary Access Control Policy** to restrict the ability to modify and **delete** the security attributes of the objects covered by the SFP to the owner of the object and

- **a) For MVS objects:**
  1. **users with the SPECIAL attribute or the appropriate group-SPECIAL attribute and**
  2. **users who have ALTER authority to the object**
- **b) For UNIX objects a user with z/OS UNIX superuser privilege**
- **c) For LDAP LDBM objects:**
  1. **The directory Administrator**
  2. **Members of the LDAP administrative group, subject to the constraints defined in the administrative roles to which they are assigned.**
  3. **Users with DAC authority to move or rename an object and**
  4. **Users with write authority to restricted attributes in the object.**

7.1.4.3 Static attribute initialization: discretionary access (FMT_MSA.3(DAC))

FMT_MSA.3.1 The TSF shall enforce the **Discretionary Access Control Policy** to provide restrictive default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **users with the SPECIAL attribute and the owner of the profile protecting the object** to specify alternative initial values to override the default values when an object or information is created.

7.1.4.4 Static attribute initialization: network (FMT_MSA.3(NI))

FMT_MSA.3.1 The TSF shall enforce the Network Information Flow Control Policy to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the **users with READ access to the appropriate profiles in the SERVAUTH class as specified in section “Communications Server ipsec Command Interface”** to specify alternative initial values to override the default values when an object or information is created.

Application note: *The ability to perform IPSec, IP filtering, and defensive filtering related network management functions can be delegated to users by providing them READ access to the profile EZB.IPSECCMD.sysname.tcpprocname.CONTROL in the SERVAUTH class of RACF.*

7.1.4.5 Security attribute value inheritance (FMT_MSA.4)

FMT_MSA.4.1 The TSF shall use the following rules to set the value of security attributes for objects covered by an access control policy **using the following rules**:

MVS objects:

Object security attributes for MVS objects are stored in RACF profiles. MVS objects inherit their object security attributes from the RACF profile that most closely matches the object's name. Users allowed to modify the profile (including the access control list) that protects the data set can modify the security attributes.

z/OS UNIX file system objects

Object security attributes for UNIX file system objects are stored with the files. UNIX file system objects obtain the values of their ACL security attributes from the model ACL of the parent directory.

z/OS UNIX IPC objects

Object security attributes for UNIX IPC objects are defined by the creator.

LDAP LDBM objects:

LDAP LDBM objects can inherit their ACLs and owner attributes from nodes above them in the directory hierarchy. Inheritance is controlled by the aclPropagate and ownerPropagate attributes. An LDBM object's ACL and owner

attribute marked with these values inherit their respective attribute to objects below in the hierarchy, unless another ACL or owner attribute with the aclPropagate or ownerPropagate value is encountered, or the ACL or owner attributes for the object have been set explicitly.

7.1.4.6 Management of TSF data: audit events (FMT_MTD.1(AE))

FMT_MTD.1.1 The TSF shall restrict the ability to query, modify the set of audited events to **users that satisfy the following rules:**

a) **users with the AUDITOR role**
b) **for events related to a profile: the profile owner.**

Application note: *This SFR applies to FAU_SEL.1.*

7.1.4.7 Management of TSF data: audit storage (FMT_MTD.1(AS))

FMT_MTD.1.1 The TSF shall restrict the ability to clear, **create, delete** the audit storage to **users that satisfy the following rules:**

a) **users with the AUDITOR role**
b) **z/OS operators.**

Application note: *This SFR applies to FAU_STG.1.*

7.1.4.8 Management of TSF data: audit trail threshold (FMT_MTD.1(AT))

FMT_MTD.1.1 The TSF shall restrict the ability to modify the

a) threshold of the audit trail when an action is performed;
b) action when the threshold is reached

to **the authorized administrator with at least WRITE access to the SYS1.PARMLIB data set**.

Application note: *This SFR applies to FAU_STG.3.*

7.1.4.9 Management of TSF data: audit storage failure (FMT_MTD.1(AF))

FMT_MTD.1.1 The TSF shall restrict the ability to modify the actions to be taken in case of audit storage failure to **the authorized administrator with at least WRITE access to the SYS1.PARMLIB data set**.

Application note: *This SFR applies to FAU_STG.4.*

7.1.4.10 Management of TSF data: certificate/public key management (FMT_MTD.1(CM))

FMT_MTD.1.1 The TSF shall restrict the ability to import, enable, disable the digital certificates or public keys used for remote entity authentication **and no other security function** to **users allowed to configure IPSec (for certificates used for IPSec) or System SSL (for certificates used for TLS)**.

Application note: *This SFR applies to FIA_CERT_EXT.1.*
Management of digital certificates is bound to the management of the key rings used by the IKE daemon and by AT-TLS. Functions for their management are restricted by RACF profiles as described in the TOE summary specification. This allows for a fine-grained access control to individual certificate management functions (both for PKCS#11 TKDS as well as in the RACF database) resulting in the possibility to assign specific certificate management functions to separate individuals. Each installation may define its own management model based on this fine-grained access control functions.

7.1.4.11 Management of TSF data: network filters (FMT_MTD.1(NI))

FMT_MTD.1.1 The TSF shall restrict the ability to define, query, modify, delete, **perform command-driven management functions for IPSec, IP filtering, and defensive filtering configuration related to** the security attributes for the rules governing the

a) identification of network data;
b) actions performed on the identified network data

to **users with READ access to the appropriate profiles in the SERVAUTH class.**

Application note: *This SFR applies to FDP_IFF.1(NI).*

Application note: *The ability to perform IPSec, IP filtering, and defensive filtering related network management functions can be delegated to users by providing them READ access to profiles of the form EZB.IPSECCMD.sysname.[one of: tcpname, clientname, sysname, DMD, GLOBAL] (potentially followed by a function name) in the SERVAUTH class of RACF. A list of profiles and the network management function they protect can be found in the TOE Summary Specification.*

7.1.4.12 Management of TSF data: authentication threshold (FMT_MTD.1(IAT))

FMT_MTD.1.1 The TSF shall restrict the ability to modify the threshold for unsuccessful authentication attempts to **users that satisfy the following rules:**

a) user has SPECIAL
b) user has group-SPECIAL in the group that owns the user, or group-SPECIAL in a higher group in the group tree if group

ownership is setup appropriately.

Application note: *This SFR applies to FIA_AFL.1.*

7.1.4.13 Management of TSF data: account re-enablement (FMT_MTD.1(IAF))

FMT_MTD.1.1 The TSF shall restrict the ability to re-enable the authentication to the account subject to authentication failure to **users that satisfy the following rules:**

- **a) user has SPECIAL**
- **b) user has group-SPECIAL in the group that owns the user, or group-SPECIAL in a higher group in the group tree if group ownership is setup appropriately**

In addition, the following users may change the REVOKE status and the user's password or password phrase:

- **c) user has READ access to FACILITY resource IRR.PASSWORD.RESET assuming the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**
- **d) user has READ access to FACILITY resource IRR.PWRESET.OWNER.owner-of-profile where "owner-of-profile" specifies the user or group that owns the revoked user, and the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**
- e) **user has READ access to FACILITY resource IRR.PWRESET.TREE.owner-of-tree where "owner-of-tree" specifies a user or group that would have group-SPECIAL over the revoked user, and the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**

Application note: *This SFR applies to FIA_AFL.1.*

7.1.4.14 Management of TSF data: user security attributes (FMT_MTD.1(IAU))

FMT_MTD.1.1 The TSF shall restrict the ability to initialize, modify, delete the user security attributes *other than authentication data,* to **users that satisfy the following rules:**

- **a) user has SPECIAL**
- **b) users with CLAUTH attribute for the USER class**
- **c) user has group-SPECIAL in the group that owns the user, or group-SPECIAL in a higher group in the group tree if group ownership is setup appropriately**
- **d) LDAP administrators with appropriate roles for administration of LDAP-based users, groups, and roles.**

Application note: *This SFR applies to FIA_ATD.1, FIA_UAU.1, FIA_UID.1.*

7.1.4.15 Management of TSF data: authentication data (FMT_MTD.1(IAU-AUTH))

FMT_MTD.1.1 The TSF shall restrict the ability to modify the user *authentication data* to **users that satisfy the following rules:**

a) **users authorized to modify their own authentication data (passwords and passphrases)**
b) **Users with the SPECIAL or appropriate group-SPECIAL attribute can modify a user's password/phrase;**
c) **Users with access to FACILITY resource IRR.PASSWORD.RESET are allowed to reset passwords/phrases for any user that does not have the PROTECTED, SPECIAL, AUDITOR, or OPERATIONS attributes;**
d) **Users with access to FACILITY resource IRR.PWRESET.OWNER.owner-value are allowed to reset passwords/phrases for users owned by "owner-value" if those users do not have the PROTECTED SPECIAL, AUDITOR, or OPERATIONS attributes and are not exempted from reset by the IRR.PWRESET.EXCLUDE.userID resource in the FACILITY class;**
e) **Users with access to FACILITY resource IRR.PWRESET.TREE.owner-value are allowed to reset passwords/phrases for users in the scope of the group specified by "owner-value" if those users do not have the PROTECTED SPECIAL, AUDITOR, or OPERATIONS attributes and are not exempted from reset by the IRR.PWRESET.EXCLUDE.userID resource in the FACILITY class. (Note: this "tree" function applies to the same target users that group-SPECIAL would affect.);**
f) **Users may be allowed to renew or revoke their own digital certificates via the z/OS PKI Services component.**
g) **Users with the required authorities to issue the RACDCERT subcommands for the management of digital certificates as defined in the section "Authority checking for RACDCERT Processing" in the TOE Summary Specification.**

Application note: *This SFR applies to FIA_ATD.1, FIA_UAU.1, FIA_UID.1.*

7.1.4.16 Management of TSF data: EIA (FMT_MTD.1(EIA))

FMT_MTD.1.1 The TSF shall restrict the ability to **initialize, modify, delete** the **user security attributes used for the remote identification and authentication policy** to **the authorized administrator, or users that satisfy the following rules:**

a) **user has SPECIAL**

b) **user has group-SPECIAL in the group that owns the user, or group-SPECIAL in a higher group in the group tree if group ownership is setup appropriately**

In addition, the following users may reset the REVOKE status of a revoked user account (re-enable the account) and the user's password or password phrase:

c) **user has READ access to FACILITY resource IRR.PASSWORD.RESET assuming the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**
d) **user has READ access to FACILITY resource IRR.PWRESET.OWNER.owner-of-profile where "owner-of-profile" specifies the user or group that owns the revoked user, and the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**
e) **user has READ access to FACILITY resource IRR.PWRESET.TREE.owner-of-tree where "owner-of-tree" specifies a user or group that would have group-SPECIAL over the revoked user, and the revoked user does not have the SPECIAL, OPERATIONS, or AUDITOR attributes.**

7.1.4.17 Management of TSF data: key import (FMT_MTD.1(CRYPTO1))

FMT_MTD.1.1 The TSF shall restrict the ability to **import or modify** the **cryptographic keys** to **the authorized administrator**.

Application note: *The process of a user requesting a certificate from PKI Services involves the user sending a public key to the PKI server. Similarly, authentication of a client via TLS involves the client sending a public key to the server. For the purposes of this ST, neither of those operations, nor other operations similar to them, are considered to be importation of a cryptographic key.*

7.1.4.18 Management of TSF data: digital certificates (FMT_MTD.1(CRYPTO2))

FMT_MTD.1.1 The TSF shall restrict the ability to **perform management functions for** the **digital certificates** to **users with the SPECIAL attribute and users assigned authority to specific management functions as defined in the tables in the section on managing digital certificates in the TOE Summary Specification**.

Application note: *To perform a specific management function for digital certificates, a user that does not have the SPECIAL attribute must have RACF authority to a profile of the type IRR.DIGTCERT.function in the FACILITY class where function is the name of the management function. The list of management functions and the semantics of READ, UPDATE and CONTROL authority for each function is defined in the tables in Authority checking for RACDCERT Processing, Authority Checking for R_datalib Processing and*

Authority Checking for PKCS#11 Cryptographic Tokens in the ICSF TKDS. That chapter also discusses use of resources in the CRYPTOZ resource class to control access to PKCS#11 tokens. To determine the authority a user has to those profiles, RACF uses the algorithm defined in FDP_ACF.1(1).

7.1.4.19 Revocation: objects (FMT_REV.1(OBJ))

FMT_REV.1.1 The TSF shall restrict the ability to revoke object security attributes defined by SFPs associated with the corresponding object under the control of the TSF to **users that satisfy the following rules:**

a) users authorized to modify the security attributes covered by the Discretionary Access Control Policy.

FMT_REV.1.2 The TSF shall enforce the following rules:

a) The access rights associated with an object shall be enforced when an access check is made;
b) no other rule

Application note: *For the access rights to data sets, z/OS UNIX file system objects, volumes, terminals, and TCP/IP connections, the access checks are performed once when the user starts to use the resource and are not checked again until the user releases the resource and attempts to use it again. Immediate revocation for these attributes can be achieved by terminating all active jobs of the user, his TSO sessions and all the z/OS UNIX processes acting on behalf of this user.*

7.1.4.20 Revocation: users (FMT_REV.1(USR))

FMT_REV.1.1 The TSF shall restrict the ability to revoke user security attributes defined by the SFP associated with the corresponding user under the control of the TSF to **the authorized roles as defined by FMT_SMR.1 bullets a), c), f), g), h), n), p), and q)**.

Application note: *z/OS has several kinds of authorized administrators, including users with SPECIAL and group-SPECIAL attributes, as well as owners and users with authority to change another user's password/phrase. All of these can, in some sense, revoke some or all of a user's security attributes. Additionally, via PKI Services, users who own a digital certificate may request revocation of their certificate, and posting of that certificate to the certificate revocation list (CRL) maintained by PKI Services.*

FMT_REV.1.2 The TSF shall enforce the following rules:

a) The enforcement of the revocation of security-relevant authorizations with the next user-subject binding process during the next authentication of the user;

b) **no other rules**.

7.1.4.21 Remote Management Capabilities (FMT_SMF_RMT.1)

FMT_SMF_RMT.1.1 The TSF shall allow management functions also to be performed from a remote IT entity using a trusted channel established in accordance with the requirements stated in FTP_ITC.1.

7.1.4.22 Security roles (FMT_SMR.1)

FMT_SMR.1.1 The TSF shall maintain the roles:

a) authorized administrator *(user with the SPECIAL attribute)*;
b) regular user
c) User role with the following rights:
  i. Users are authorized to modify their own user password;
  ii. Users are authorized to modify the access control permissions for the named objects they own;
d) users authorized by the discretionary access control policy to modify object security attributes;
e) users authorized to modify their own authentication data;
f) users authorized to perform administrative actions within a defined group (group-SPECIAL attribute)
g) users authorized to perform administrative actions for user or group security attributes via ownership
h) RACF auditors (users who have the RACF AUDITOR attribute in their profiles)
i) RACF group auditors (users who have the RACF group-AUDITOR attribute in their profiles)
j) Operations roles (users with the OPERATIONS attribute)
k) z/OS operators (users who are allowed to issue operator commands)
l) z/OS pseudo-user (protected user IDs used for executing defined started tasks, and for "anonymous" access to administrator-specified data via HTTP)
m) z/OS UNIX superuser
n) LDAP Administrator (as specified in the LDAP configuration file or added to the administrative group as defined in o) below)
o) Additional LDAP administration roles if the LDAP configuration file specifies that the LDAP administration group is enabled: Directory Data Administrator, No Administrator, Replication Administrator, Root Administrator, Schema Administrator, Server Configuration Group Member, Password Administrator, and Operational Administrator.
p) PKI Services Administrator (as specified by authorization

to FACILITY class resource IRR.RPKISERV.PKIADMIN and for some functions specific IRR.DIGTCERT.function-name resources). PKI Services administration may also be governed by authorization to PKISERV class resources that are based on the instance of PKI, the administration action to be performed and the type of certificate or request.

q) **Users authorized to perform management operations for digital certificates based on access rights to RACF profiles protecting the individual management operations**
r) **Users authorized to perform IPSec network management functions based on access rights to RACF profiles protecting the individual management operations**
s) **Users authorized to perform other management functions based on access rights to RACF profiles protecting the individual management operations.**

FMT_SMR.1.2 The TSF shall be able to associate users with roles.

7.1.5 Protection of the TSF (FPT)

7.1.5.1 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps.

7.1.6 TOE access (FTA)

7.1.6.1 TSF-initiated session locking (FTA_SSL.1)

FTA_SSL.1.1 The TSF shall lock an interactive session to a human user maintained by the TSF after **an administrator-defined time interval of user inactivity** by:

a) clearing or overwriting TSF controlled display devices, making the current contents unreadable;
b) disabling any activity of the user's data access/TSF controlled display devices other than unlocking the session.

FTA_SSL.1.2 The TSF shall require the following events to occur prior to unlocking the session:

a) Successful re-authentication with the credentials of the user owning the session using **one of the authentication methods out of the list of allowed methods specified in FIA_UAU.5**;
b) **no other events**.

Application note: *This SFR applies to directly attached terminals, not networked sessions.*

Application note: *It is possible that the TSF establishes a connection to a session on a remote trusted IT system, for example when using SSH. This remote trusted IT system maintains the session established with the communication channel. The locking requirement however applies to the session maintained by the TSF only as the TSF can only exercise control of the sessions it maintains.*

7.1.6.2 User-initiated locking (FTA_SSL.2)

FTA_SSL.2.1 The TSF shall allow user-initiated locking of the user's own interactive session maintained by the TSF, by:

a) clearing or overwriting TSF controlled display devices, making the current contents unreadable;
b) disabling any activity of the user's data access/TSF controlled display devices other than unlocking the session.

FTA_SSL.2.2 The TSF shall require the following events to occur prior to unlocking the session:

a) Successful re-authentication with the credentials of the user owning the session using **one of the authentication methods from the list of allowed methods specified in FIA_UAU.5**;
b) no other events.

Application note: *This SFR applies to directly attached terminals, not networked sessions.*
Application note: *See also application note for FTA_SSL.1*

7.1.7 Trusted path/channels (FTP)

7.1.7.1 Inter-TSF trusted channel (FTP_ITC.1)

FTP_ITC.1.1 The TSF shall provide a communication channel between itself and another trusted IT product that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification and disclosure using the following mechanisms:

Cryptographically-protected communication channel using:

i. **SSH as defined in RFCs 4251, 4252, 4253, and 4254 with a combination of the following cipher suites defined there:**
   - **3DES-CBC, AES256-CBC, AES128-CBC, AES192-CBC for encryption;**
   - **HMAC_SHA1, HMAC-SHA1-96, HMAC-MD5, HMAC-MD5-96 for integrity**
   - **DIFFIE-HELLMAN-GROUP14-SHA1, DIFFIE-HELLMAN-GROUP1-SHA1 for key exchange**
   - **SSH-DSS, SSH-RSA for public key encryption;**

ii. **TLS as defined in RFC 5246 using X.509 certificates and supporting the following cipher suites defined there:**

- **TLS_RSA_WITH_AES_128_CBC_SHA**
- **TLS_RSA_WITH_AES_256_CBC_SHA**
- **TLS_RSA_WITH_AES_128_CBC_SHA256**
- **TLS_RSA_WITH_AES_256_CBC_SHA256**
- **TLS_DH_DSS_WITH_AES_128_CBC_SHA**
- **TLS_DH_RSA_WITH_AES_128_CBC_SHA**
- **TLS_DHE_DSS_WITH_AES_128_CBC_SHA**
- **TLS_DHE_RSA_WITH_AES_128_CBC_SHA**
- **TLS_DH_DSS_WITH_AES_256_CBC_SHA**
- **TLS_DH_RSA_WITH_AES_256_CBC_SHA**
- **TLS_DHE_DSS_WITH_AES_256_CBC_SHA**
- **TLS_DHE_RSA_WITH_AES_256_CBC_SHA**
- **TLS_DH_DSS_WITH_AES_128_CBC_SHA256**
- **TLS_DH_RSA_WITH_AES_128_CBC_SHA256**
- **TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256**
- **TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384**
- **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256**
- **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384**

iii. **IPSEC protocol ESP as defined in RFC 4303 using the cryptographic algorithms:**

- **AES-CBC-128, AES-CBC-256 (both specified by RFC 3602), AES-GCM-128 as specified in RFC 4106, AES-GCM-256 as specified in RFC 4106 for ESP encryption;**
- **HMAC-SHA1-96, AES-XCBC-MAC-96 for ESP authentication and authentication header protection;**
- **IKEv1 as defined in RFCs 2407, 2408, 2409, RFC 4109, and no other RFCs for hash functions, IKEv2 as defined in RFCs 5996, 4307 and no other RFCs for hash functions, for key negotiation and SA establishment;**
- **DH Groups 14 (2048-bit MODP), and 24 (2048-bit MODP with 256-bit POS), 19 (256-bit Random ECP), 20 (384-bit Random ECP), DH Group 21 (521-bit), DH Group 5 (1536-bit) (not usable when configured for FIPS mode) for use in IKE key establishment;** ***(Note: also DH Groups 1 and 2 are supported, but not in FIPS mode);***
- **ECDSA** ***and RSA*** **algorithm for Peer Authentication.**

FTP_ITC.1.2 The TSF shall permit **the TSF, another trusted IT product** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for all security functions specified in the ST that interact with remote trusted IT systems and **all conditions defined in the configuration where the establishment of a trusted channel is required**.

7.2 Security Functional Requirements Rationale

7.2.1 Security Requirements Coverage

The following table provides a mapping of SFR to the security objectives, showing that each security functional requirement addresses at least one security objective.

Security Functional Requirements	Objectives
FAU_GEN.1	O.AUDITING
FAU_GEN.2	O.AUDITING
FAU_SAR.1	O.AUDITING
FAU_SAR.2	O.AUDITING
FAU_SEL.1	O.AUDITING
FAU_STG.1	O.AUDITING
FAU_STG.3	O.AUDITING
FAU_STG.4	O.AUDITING
FDP_ACC.1(z/OS objects)	O.DISCRETIONARY.ACCESS, O.SUBJECT.COM
FDP_ACF.1(MVS)	O.DISCRETIONARY.ACCESS
FDP_ACF.1(UNIX)	O.DISCRETIONARY.ACCESS
FDP_ACF.1(LDAP)	O.DISCRETIONARY.ACCESS

Security Functional Requirements	Objectives
FDP_ACF.1(UNIX IPC)	O.SUBJECT.COM
FDP_IFC.1	O.NETWORK.FLOW
FDP_IFF.1	O.NETWORK.FLOW
FDP_RIP.2	O.AUDITING, O.DISCRETIONARY.ACCESS, O.I&A, O.NETWORK.FLOW, O.SUBJECT.COM
FIA_AFL.1	O.I&A
FIA_ATD.1	O.I&A
FIA_UAU.1(RITE)	O.I&A
FIA_UAU.1(HU)	O.I&A
FIA_UAU.5	O.I&A,
FIA_UAU.7	O.I&A
FIA_UID.1	O.I&A
FIA_USB.1	O.I&A, O.DISCRETIONARY.ACCESS[1]
FIA_PK_EXT.1(IPSEC-TLS)	O.TRUSTED_CHANNEL
FIA_PK_EXT.1(SSH)	O.TRUSTED_CHANNEL
FMT_MOF.1	O.I&A, O.MANAGE
FMT_MSA.1	O.DISCRETIONARY.ACCESS[1],

[1] The author of this Security Target thinks that in addition to what GP-OSPP states this SFR also addresses O.DISCRETIONARY.ACCESS

Security Functional Requirements	**Objectives**
	O.MANAGE
FMT_MSA.3(DAC)	O.DISCRETIONARY.ACCESS[1], O.MANAGE
FMT_MSA.3(NI)	O.NETWORK.FLOW[2], O.MANAGE
FMT_MSA.4	O.DISCRETIONARY.ACCESS[1], O.MANAGE
FMT_MTD.1(AE)	O.AUDITING[3], O.MANAGE
FMT_MTD.1(AS)	O.AUDITING[3], O.MANAGE
FMT_MTD.1(AT)	O.AUDITING[3], O.MANAGE
FMT_MTD.1(AF)	O.AUDITING[3], O.MANAGE
FMT_MTD.1(CM)	O.TRUSTED.CHANNEL[4] O.MANAGE
FMT_MTD.1(NI)	O.NETWORK.FLOW[2], O.MANAGE
FMT_MTD.1(IAT)	O.I&A[5], O.MANAGE
FMT_MTD.1(IAF)	O.I&A[5], O.MANAGE
FMT_MTD.1(IAU)	O.MANAGE

[2] The author of this Security Target thinks that in addition to what GP-OSPP states this SFR also addresses O.NETWORK.FLOW

[3] The author of this Security Target thinks that in addition to what GP-OSPP states this SFR also addresses O.AUDITING

[4] The author of this Security Target thinks that in addition to what GP-OSPP states this SFR also addresses O.TRUSTED_CHANNEL

[5] The author of this Security Target thinks that in addition to what GP-OSPP states this SFR also addresses O.I&A

Security Functional Requirements	Objectives
FMT_MTD.1(IAU-AUTH)	O.I&A, O.MANAGE
FMT_MTD.1(EIA)	O.I&A, O.MANAGE
FMT_MTD.1(CRYPTO1)	O.TRUSTED_CHANNEL, O.MANAGE
FMT_MTD.1(CRYPTO2)	O.TRUSTED_CHANNEL, O.MANAGE
FMT_REV.1(OBJ)	O.MANAGE
FMT_REV.1(USR)	O.MANAGE
FMT_SMF_RMT.1	O.MANAGE
FMT_SMR.1	O.MANAGE
FPT_STM.1	O.AUDITING
FTA_SSL.1	O.UNATTENDED_SESSION[6]
FTA_SSL.2	O.UNATTENDED_SESSION[6]
FTP_ITC.1	O.TRUSTED_CHANNEL

Table 6: Mapping of Security Functional Requirements to Security Objectives

7.2.2 Security Requirements Sufficiency

The following rationale provides justification for each security objective for the TOE, showing that the security functional requirements are suitable to meet and achieve the security objectives:

[6] The author of this Security Target thinks that the mapping in GP-OSPP is wrong and needs to be modified to a mapping to O.UNATTENDED_SESSION

Security objectives	Rationale
O.AUDITING	The events to be audited are defined in FAU_GEN.1 and are associated with the identity of the user that caused the event (FAU_GEN.2). Authorized users are provided the capability to read the audit records (FAU_SAR.1), while all other users are denied access to the audit records (FAU_SAR.2). The authorized user must have the capability to specify which audit records are generated (FAU_SEL.1). The TOE prevents the audit log from being modified or deleted (FAU_STG.1) and ensures that the audit log is not lost due to resource shortage (FAU_STG.3, FAU_STG.4). To support auditing, the TOE is able to maintain proper time stamps (FPT_STM.1). The protection of reused resources ensures that no data leaks from other protected sources (FDP_RIP.2). The management of the audit functionality is expressed by the ability to manage the events to be audited (FMT_MTD.1(AE)), manage the audit trail storage (FMT_MTD.1(AS)), manage the audit threshold and the action to be performed when this threshold is reached (FMT_MTD.1(AT)), and manage the actions to be taken in the event of an audit storage failure (FMT(MTD.1(AF)).
O.DISCRETIONARY.ACCESS	The TSF must control access to resources based on the identity of users that are allowed to specify which resources they want to access for storing their data. The access control policy must have a defined scope of control (FDP_ACC.1(z/OS objects)). The rules for the access control policy are defined by FDP_ACF.1(MVS), FDP_ACF.1(UNIX), FDP_ACF.1(LDAP), and FDP_ACF.1(UNIX IPC). The protection of reused resources ensures that no data leaks from other protected sources (FDP_RIP.2). FIA_USB.1 contributes to the objective by defining the subject security attributes used to enforce the discretionary access control policy. In addition FMT_MSA.1 and FMT_MSA.4 contribute to the objective by controlling the setting, modification and deletion of security attributes of objects and FMT_MSA3(DAC) which controls the setting of default values for those.
O.NETWORK.FLOW	The packet filter mechanism controls the information flowing between different entities (FDP_IFC.1). The TOE

Security objectives	Rationale
	implements a rule-set governing the information flow (FDP_IFF.1). The protection of reused resources ensures that no data leaks from other protected sources (FDP_RIP.2). In addition FMT_MSA.3(NI) and FMT_MTD.1(NI) contribute to the objective by restricting the ability to manage aspects related to networking and the network policy.
O.SUBJECT.COM	The TSF must control the exchange of data using transient storage objects between subjects based on the identity of users. The access control policy must have a defined scope of control (FDP_ACF.1(z/OS objects)). The rules for the access control policy are defined in FDP_ACF.1(UNIX IPC). The protection of reused resources ensures that no data leaks from other protected sources (FDP_RIP.2).
O.I&A	The TSF must ensure that only authorized users gain access to the TOE and its resources. Users authorized to access the TOE must use an identification and authentication process (FIA_UID1, FIA_UAU.1(HU), FIA_UAU.1(RITE)). Multiple I&A mechanisms are allowed as specified in FIA_UAU.5. To ensure authorized access to the TOE, authentication data is protected (FIA_ATD.1, FIA_UAU.7). Proper authorization for subjects acting on behalf of users is also ensured (FIA_USB.1). To support the strength of authentication methods, the TOE is capable of identifying and reacting to unsuccessful authentication attempts (FIA_AFL.1). The protection of reused resources ensures that no data leaks from other protected sources (FDP_RIP.2). FMT_MOF.1 contributes to the objective by managing the behavior of the password based user authentication, FMT_MTD.1(IAT) contributes to the objective by defining the management of the number of unsuccessful authentication attempts, FMT_MTD.1(IAF) contributes to the objective by defining the conditions an administrative user has to satisfy to re-enable users that have been blocked, FMT_MTD.1(IAU-AUTH) contributes to the objective by defining how authentication data can be managed, and FMT_MTD.1(EIA) contributes to the objective by defining management of security attributes

Security objectives	Rationale
	used for remote user authentication.
O.MANAGE	The TOE provides management interfaces for: • the access control policies (FMT_MSA.1, FMT_MSA.3(DAC), FMT_MSA.4; • the information flow control policy (FMT_MSA.3(NI), FMT_MTD.1(NI); • the auditing aspects (FMT_MTD.1(AE), FMT_MTD.1(AS), FMT_MTD.1(AT), FMT_MTD.1(AF)); • the identification and authentication aspects (FMT_MOF.1, FMT_MTD.1(IAT), FMT_MTD.1(IAF), FMT_MTD.1(IAU), FMT_MTD.1(IAU-AUTH), FMT_MTD.1(EIA)); • the management of key material used for setting up a trusted channel (FMT_MTD.1(CM), FMT_MTD.1(CRYPTO1) and FMT_MTD.1(CRYPTO2)); The rights management for the different management aspects is defined with FMT_SMR.1. The management interfaces for the revocation of user and object attributes is provided with (FMT_REV.1_OBJ, FMT_REV.1_USR). The possibility to perform management from a remote system is addressed by FMT_SMF_RMT.1.
O.TRUSTED_CHANNEL	The TOE provides a trusted channel protecting communication between a remote trusted IT system and itself (FTP_ITC.1). The management of the key material and certificates used for setting up a trusted channel is defined in FMT_MTD.1(CM, FMT_MTD.1(CRYPTO1) and FMT_MTD.1(CRYPTO2). The authentication of the end point of the trusted channel is based on public key cryptography, which in the case of IPSec and TLS uses digital certificates (FIA_PK_EXT.1(IPSEC-TLS)) and in the case of SSH uses administrator entered public key material (FIA_PK_EXT.1(SSH).
O.UNATTENDED_SESSION	The TOE provides functions that require a user re-

Security objectives	Rationale
	authentication for an unattended sessions that may either be initiated by the user (FTA_SSL.2) or by the system after a defined period of inactivity (FTA_SSL.1).

Table 7: Mapping Security Objectives to Security Functional Requirements

7.2.3 Security Requirements Dependency Analysis

The following table demonstrates the dependencies of SFRs modeled in CC Part 2 as well as the ones defined by extended components and how the SFRs for the TOE resolve those dependencies:

Security Functional Requirements	Dependency	Resolved by
FAU_GEN.1	FPT_STM.1	FPT_STM.1
FAU_GEN.2	FAU_GEN.1	FAU_GEN.1
	FIA_UID.1	FIA_UID.1
FAU_SAR.1	FAU_GEN.1	FAU_GEN.1
FAU_SAR.2	FAU_SAR.1	FAU_SAR.1
FAU_SEL.1	FAU_GEN.1	FAU_GEN.1
	FMT_MTD.1	FMT_MTD.1(AE)
FAU_STG.1	FAU_GEN.1	FAU_GEN.1
FAU_STG.3	FAU_STG.1	FAU_STG.1
FAU_STG.4	FAU_STG.1	FAU_STG.1
FDP_ACC.1(z/OS objects)	FDP_ACF.1	FDP_ACF.1(MVS), FDP_ACF.1(UNIX), FDP_ACF.1(LDAP), FDP_ACF.1(UNIX IPC)

Security Functional Requirements	**Dependency**	**Resolved by**
FDP_ACF.1(MVS)	FDP_ACC.1	FDP_ACC.1(z/OS objects)
	FMT_MSA.3	FMT_MSA.3(DAC)
FDP_ACF.1(UNIX)	FDP_ACC.1	FDP_ACC.1(z/OS objects)
	FMT_MSA.3	FMT_MSA.3(DAC)
FDP_ACF.1(LDAP)	FDP_ACC.1	FDP_ACC.1(z/OS objects)
	FMT_MSA.3	FMT_MSA.3(DAC)
FDP_ACF.1(UNIX IPC)	FDP_ACC.1	FDP_ACC.1(z/OS objects)
	FMT_MSA.3	FMT_MSA.3(DAC)
FDP_IFC.1	FDP_IFF.1	FDP_IFF.1
FDP_IFF.1	FDP_IFC.1	FDP_IFC.1
	FMT_MSA.3	FMT_MSA.3(NI)
FDP_RIP.2	No dependencies	-
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_ATD.1	No dependencies	-
FIA_UAU.1(RITE)	FIA_UID.1	FIA_UID.1
FIA_UAU.1(HU)	FIA_UID.1	FIA_UID.1
FIA_UAU.5	No dependencies	-
FIA_UAU.7	FIA_UAU.1	FIA_UAU.1(HU)
FIA_UID.1	No dependencies	-
FIA_USB.1	FIA_ATD.1	FIA_ATD.1

Security Functional Requirements	Dependency	Resolved by
FIA_PK_EXT.1(IPSEC-TLS)	FMT_MTD.1	FMT_MTD.1(CM)
FIA_PK_EXT.1(SSH)	FMT_MTD.1	FMT_MTD.1(CM)
FMT_MOF.1	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MSA.1	FDP_ACC.1	FDP_ACC.1(z/OS objects)
	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MSA.3(DAC)	FMT_MSA.1	FMT_MSA.1
	FMT_SMR.1	FMT_SMR.1
FMT_MSA.3(NI)	FMT_MSA.1	see note in the GP-OSPP
	FMT_SMR.1	FMT_SMR.1
FMT_MSA.4	FDP_ACC.1	FDP_ACC.1(z/OS objects)
FMT_MTD.1(AE)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(AS)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(AT)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(AF)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below

Security Functional Requirements	**Dependency**	**Resolved by**
FMT_MTD.1(CM)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(NI)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(IAT)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(IAF)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(IAU)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(IAU-AUTH)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(EIA)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(CRYPTO1)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_MTD.1(CRYPTO2)	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	see note below
FMT_REV.1(OBJ)	FMT_SMR.1	FMT_SMR.1
FMT_REV.1(USR)	FMT_SMR.1	FMT_SMR.1

Security Functional Requirements	**Dependency**	**Resolved by**
FMT_SMF_RMT.1	FTP_ITC.1	FTP_ITC.1
FMT_SMR.1	FIA_UID.1	FIA_UID.1
FPT_STM.1	No dependencies	-
FTA_SSL.1	FIA_UID.1	FIA_UID.1
FTA_SSL.2	FIA_UID.1	FIA_UID.1
FTP_ITC.1	No dependencies	-

Table 8: Security Functional Requirements Dependency Analysis

Note: the authors of [GPOSPP] did not include FMT_SMF.1 in the Protection Profile, since they believe that the specific instances of FMT_MTD.1 make this dependency obsolete. Please see the rationale for this in the dependency analysis in [GPOSPP]. This rationale also applies for the additional instantiation of FMT_MTD.1 defined in this Security Target.

7.2.4 TSF Rationale

The following table maps the security functional requirements to the security functions as defined in the TOE summary specification to show that all security functional requirements are addressed by the security functions.

SFR	**Security Functions**
FAU_GEN.1	Section 8.4.3.1 "Generation of audit records" explains how audit records are generated. This section also explains the structure of the audit records.
FAU_GEN.2	Section 8.4.3.1 "Generation of audit records" explains the information contained in the audit records. Tools to export audit records in human-readable format are mentioned in this section, too.

SFR	Security Functions
FAU_SAR.1	Section 8.4.5.1, subsection "User roles and attributes" and subsection "RACF Roles", part titled "AUDITOR and group-AUDITOR" explain the auditor role. Section 8.4.3.1 describes the purpose of the audit dump programs that read audit records from the audit trail and store them in a data set where they can be assessed.
FAU_SAR.2	Section 8.4.3.2 "Protection of the audit trail" explains how to protect the audit trail from unauthorized access.
FAU_SEL.1	Sections 8.4.3.1 "Generation of audit records" and 8.4.5.1, subsection "RACF roles", paragraph titled "AUDITOR and group-AUDITOR" explain how the auditor role can configure the events that are audited. These chapters also explain that the owner of a profile can define which events related to the profile are audited.
FAU_STG.1	Section 8.4.3.2 "Protection of the audit trail" explains how to protect the audit trail from unauthorized access.
FAU_STG.3	Section 8.4.3.2 "Protection of the audit trail explains how the operator is informed about the fact that a SMF data set is full and the TOE has switched to the next non-full SMF data set.
FAU_STG.4	Section 8.4.3.2 "Protection of the audit trail explains how the TOE prevents the loss of audit data by halting the system on audit trail exhaustion.
FDP_ACC.1(z/OS objects)	The general operation of access control is explained in section 8.4.2 "Access Control". The possible access rights for discretionary access control are explained in section 8.4.2.1 "Discretionary Access Control". Since access control to TCP/IP stacks, addresses and ports is managed with RACF profiles, this explanation is also applicable to these transient objects. The protected resources are explained in section 8.4.2.1, subsection "Protected resources". Unix objects and their access control attributes are explained in section 8.4.2.1, subsections "UNIX file system objects" and "z/OS UNIX IPC objects". LDAP objects and their access control attributes are explained in 8.4.2.1, subsection LDAP LDBM objects.

SFR	Security Functions
FDP_ACF.1(MVS)	Discretionary access control for z/OS MVS objects is explained in section 8.4.2.1, subsection "DAC for MVS resources" and subsection "Algorithm to check for DAC access to MVS resources" , listing all the different types of objects and the specifics of their access control mechanisms.
FDP_ACF.1(UNIX)	Section 8.4.2.1, subsection "DAC for UNIX objects" explains access control for z/OS UNIX objects. This subsection has a part that defines the access control algorithm for UNIX file system objects titled "Algorithm to check DAC access to UNIX file system objects".
FDP_ACF.1(LDAP)	Section 8.4.2.1, subsection "DAC for LDAP LDBM objects" explains access control to objects in the LDAP LDBM database.
FDP_ACF.1(UNIX IPC)	Section 8.4.2.1, subsection "DAC for UNIX objects" explains access control for z/OS UNIX objects. This subsection has a part that defines the access control algorithm for UNIX IPC objects titled "Algorithm to check DAC access to UNIX IPC objects".
FDP_IFC.1 FDP_IFF.1	The packet filtering functions of the Communications Server are explained in section 8.4.9.1 "Communications Server" which has a bullet point in its list of functions related to packet filtering.
FDP_RIP.2	Object reuse is described in section 8.4.7 "Object Re-Use".
FIA_AFL.1	The system-wide attribute REVOKE for the number of failed consecutive authentication attempts is explained in the subsection "RACF Attributes" of section 8.4.5.1 and in table 23 "User Profile Structure and Content". The effect of a user ID being revoked is also described there.
FIA_ATD.1	User attributes for human users are defined in section 8.4.5.1 in the subsection "User profiles". The individual user security attributes are also explained in section 8.4.5.1.
FIA_UAU.1(RITE)	The authentication of remote IT entities is performed as part of the establishment of a trusted channel. This can be done using the TLS protocol, IPSec or SSH. Those are described in chapter 8.4.9 in separate section for each of those protocols.

SFR	Security Functions
FIA_UAU.1(HU)	User authentication is explained in section 8.4.1.
FIA_UAU.5	Authentication using passwords and password phrases is explained in section 8.4.1.1. Authentication using digital certificates is explained in section 8.4.1.1 in the subsection titled "Authentication via Client Digital Certificates". Authentication using Kerberos tickets is explained in section section 8.4.1.1 in the subsection titled "Authentication via Kerberos" Authentication using RACF PassTickets is explained in section 8.4.1.1 in the subsection titled "RACF PassTickets.
FIA_UAU.7	Section 8.4.1.1 describes that passwords are not displayed when entered during authentication (in the subsection on password phrases).
FIA_UID.1	User identification is explained in section 8.4.1.1.
FIA_USB.1	User subject binding for z/OS is explained in section 8.4.1.1. The user security attributes are taken from the user and group profiles (which are defined in section 8.4.5.1) and copied into the ACEE created for the user during the authentication process.
FIA_PK_EXT.1(IPSEC-TLS)	Certificate use for TLS and IPSec is described in section 8.4.5.7.
FIA_PK_EXT.1(SSH)	SSH key management is described in section 8.4.9.3.
FMT_MOF.1	The password policy enforced by RACF is described in section 8.4.1.1 in the subsection "Password Quality".
FMT_MSA.1	Management of object security attributes is explained in section 8.4.5.1 (and subsections) where the different RACF profiles and their management is described. For z/OS UNIX objects this is described in section 8.4.5.5 and for LDAP LDBM objects this is described in section 8.4.5.6.

SFR	Security Functions
FMT_MSA.3(DAC)	Default values for the access control are defined in the UACC attribute in the resource profiles as explained in section 8.4.2.1 in the subsection titled "DAC for MVS resources". Defaults for z/OS UNIX objects are described in the subsection titled "DAC for UNIX objects" and LDAP LDBM objects are discussed in the subsection titled "DAC for LDAP LDBM objects".
FMT_MSA.3(NI)	In section 8.4.6.1 the configuration files for the Communications Server are described.
FMT_MSA.4	The inheritance of MVS object attributes from the corresponding RACF profile is described in section 8.4.2.1 in the subsection titled "DAC for MVS resources". The inheritance mechanism for LDAP LDBM objects is described in he subsection titled "DAC for LDAP LDBM objects".
FMT_MTD.1(AE)	Management of audited events is explained in section 8.4.3.1 and in the section on the AUDITOR role (section 8.4.5.1, subsection titled "RACF roles", paragraph titled "AUDITOR and group-AUDITOR").

FMT_MTD.1(AS)
FMT_MTD.1(AT)
FMT_MTD.1(AF)
	Audit trail management is explained in section 8.4.3.2. Halting the system in case of audit trail exhaustion is described in the same section as is the threshold to warn about audit trail exhaustion is implemented by the process of switching audit data sets and dumping them to external storage. The management of Log Streams is described in the subsection titled "Using a System Log Stream for SMF" and in section 8.4.6.3.

FMT_MTD.1(CM)	Certificate and key management in RACF is described in section 8.4.5.7.
FMT_MTD.1(NI)	Configuration and management of IPSec configuration data via the command line is explained in section 8.4.9.4.

FMT_MTD.1(IAF)
FMT_MTD.1(IAT)
	The management of the system-wide attribute REVOKE for the number of failed consecutive authentication attempts is explained in section 8.4.1.1 and. The effect of a user ID being revoked is described in the description of the user profile (REVOKE attribute) in section 8.4.5.1.

SFR	Security Functions

FMT_MTD.1(IAU)
FMT_MTD.1(IAU_AUTH)
FMT_MTD.1(EIA)
	Management of user attributes is explained in section 8.4.5.1. The management authorities to resume revoked users and/or to reset their passwords is described also in section 8.4.5.1 in the text describing the authorization required to reset a user's password or passphrase. .

FMT_MTD.1(CRYPTO1)	Management of import and export of cryptographic keys is explained in section 8.4.5.7.
FMT_MTD.1(CRYPTO2)	Configuration and management of digital certificates is explained in section 8.4.5.7.
FMT_REV.1(OBJ)	Revocation of object attributes is explained as part of the management of access control to objects in section 8.4.2.1 and as part of the RACF management commands described in section 8.4.5.4.
FMT_REV.1(USR)	Revocation of user attributes is explained as part of the management of user attributes in section 8.4.5.1, subsection "User revocation" and in the description of the RACF management commands in section 8.4.5.4.
FMT_SMF_RMT.1	Remote management can be performed by a user that connects via a trusted channel with a protocol (SSH or telnet/TN3270) that allows for the use of regular commands. Some remote management actions are also possible via the LDAP SDBM backend, which is described in section 8.4.5.1, which provides an administrative interface as a wrapper to the R_admin callable service (which itself is a wrapper for RACF commands).
FMT_SMR.1	The roles are explained in sections 8.4.5.1.
FPT_STM.1	The time mechanism is explained in section 8.4.8.2.
FTA_SSL.1 FTA_SSL.2	Trivially satisfied: As stated, the TOE does not have direct interactive human connections, since the hardware does not allow direct connections of 3270 terminals any more, so session locking is irrelevant to this TOE.

SFR	Security Functions
FTP_ITC.1	The different ways of establishing a trusted channel (TLS, IPSec, SSH, Kerberos are explained in section 8.4.9 with subsection 8.4.9.2 explaining TLS, 8.4.9.3 explaining SSH, and section 8.4.9.4 explaining IPSec.

Table 9: Security Functional Requirements to Security Functions mapping

7.3 Security Assurance Requirements Rationale

The security assurance requirements for this evaluation are defined in [GPOSPP] and will be taken for this evaluation as defined there. Since this is a trial evaluation using [GPOSPP], refinements and changes to those security assurance requirements and their evaluation methodology may be derived from the experience gained within the evaluation. Updates to this Security Target during the conduct of the evaluation may be required as a result from the refinements and modifications agreed upon by the GPOSPP technical community.

8 TOE Summary Specification

8.1 TOE Architecture

8.1.1 zSeries processor hardware support

The following section provides a short overview of the supporting protection mechanisms of the abstract machine on which z/OS is running. The purpose of this section is to better understand how z/OS uses those mechanisms to protect itself against tampering and bypassing of the security functions of z/OS.

8.1.1.1 PSW, interrupt and exception handling

The System z processors have two distinctive states: problem and supervisor. A bit in a processor internal special register, the program status word (PSW) indicates if the processor is in problem or supervisor state. When in problem state the processor will not execute so called “privileged instructions”. Those include instructions to perform I/O operations, modify the content of processor control registers, set storage keys for pages within real memory, modify the hardware support tables for virtual memory management or modify critical parts of the PSW like the problem/supervisor bit or the storage key mask bits. When a program in problem state tries to execute one of those instructions, the processor generates a program check interrupt {SP.1::SP.1.1}.

8.1.1.2 Memory Management and Protection

The processor also contains support for virtual memory management. This support allows z/OS to define separate virtual address spaces and define the protection within those address spaces on a per page basis.

Pages within real storage can be protected using a so-called “storage key” that can be associated with each page of real storage. Programs can modify data within a page only if the storage key in the current PSW matches the storage key of the page or if the storage key in the current PSW is zero {SP.1::SP.1.2}. In addition pages can have an indicator, stating if the page is fetch protected. If this is the case, a program can read data from the page only if the storage key of the page and the storage of the program in the PSW match or if the storage key in the PSW is zero {SP.1::SP.1.3}. Storage protection is in effect whether the processor is in problem or supervisor state. There is one exemption from the rules stated above: If the “Storage Protection Override Control” bit is set in control register 0 of the processor, programs executing with storage key 8 are allowed to store and fetch into storage and from storage with a key of 9.

All processors within a machine share the real storage except for the first 8 KB, which are individual for each processor. The first 8 KB contain the PSWs loaded upon an interrupt.

8.1.1.3 Abstract machine modes of operation

z/OS may execute in one of these modes:

- logical partition mode

- VM guest mode

In all of those cases, z/OS operates on an abstract machine that implements the z/Architecture.

In logical partition mode, z/OS has full control of all of the resources allocated to the partition when it has been set up on the hardware management console. The logical partitioning software (PR/SM) starts the processors allocated to a partition in the “interpretative execution” mode using the SIE instruction. Each processor is then “confined” into the boundaries specified for the logical partition with respect to the physical memory and the channels it can access. Whenever a resource “virtualized” by PR/SM is accessed by an instruction on a processor, the processor breaks out of the interpretative environment into the PR/SM code which then services the request in accordance with its own policy. For z/OS this operation is transparent. PR/SM is part of the TOE environment that provides the abstract machine for the operation. PR/SM has been evaluated separately.

In VM guest mode, z/OS is operating within the boundaries defined by the z/VM operating system. z/VM is similar to PR/SM but provides more virtualization functions and more services a guest operating system may request from the virtual machine monitor. Like PR/SM z/VM also uses the SIE instruction to run a guest operating system within the boundaries of the virtual machine. z/VM itself may operate within a logical partition. When z/OS is operating in VM guest mode, the virtual machine monitor system z/VM is part of the TOE environment. z/VM itself is subject to a separate evaluation.

8.1.1.4 System Calls: SVC and PC instructions

System services offered by z/OS can be invoked from programs running in problem state using the supervisor call (SVC) and Program Call (PC) instructions of the processor. When the SVC instruction is executed, the executing processor generates an interrupt, stores the current PSW at a fixed location in absolute memory, loads a new PSW from another fixed location in absolute storage and proceeds execution at the address and with the privilege settings defined in this new PSW. During system startup z/OS has defined the new PSW to be loaded into the absolute storage in case of an interrupt or exception for all interrupts and exceptions that may occur. The new PSW contains the address of the SVC interrupt handler and z/OS checks if the caller has the required privileges to obtain the requested service before providing it.

When a program issues a supervisor call instruction the processor stores the current PSW of the calling program (which contains the instruction pointer pointing to the instruction following the supervisor call instruction) into a fixed location in the processor individual real storage in the first 8KB and loads a dedicated PSW from another location within the first 8 KB. The same procedure applies for interrupts, where each type of interrupt has dedicated locations for the “old” PSW to store and the “new” PSW to fetch. All those locations are within the first 8 KB. Program Call instructions save the current PSW (plus some other information on the caller’s context) in the linkage-stack program-call state entry. Control Register 15 serves as a stack pointer to the linkage-stack.

When a Program Call instruction is executed, the hardware checks the authorization of the caller to call the requested PC routine. A program-call number specified by the second operand address is used in a multi-level lookup to locate an entry-table entry (ETE). The program is authorized to use the ETE when the AND of the PSW-key mask in control register 3 and the authorization key mask in the ETE is nonzero or when the CPU is in the supervisor state. The ETE also defines the entry point address of the PC routine and if the PC routine will run in supervisor or problem state.

A number of SVC and PC system services as well as specific parameters of system services are restricted to authorized programs and the service will be rejected if the caller is not authorized.

8.1.1.5 Timer and Time of Day Clock

The hardware also provides a single time reference within a machine that can be used by all processors. Different time references within different processors in a parallel sysplex may also be synchronized by the hardware for platforms other than the EC12 (which only supports the Sysplex Timer Protocol for time synchronization within a Parallel Sysplex). Only users with the privileges to use the operator command to set and change the time may modify the time and date in the TOE {SP.1::SP.1.4}.

8.1.2 Hardware Management

The channel subsystem (CSS) and the IBM z/OS operating system need to know what hardware resources are available in the computer system and how these resources are connected. This information is called hardware configuration. Hardware Configuration Definition (HCD) provides an interactive interface that allows to define the hardware configuration for both the channel subsystem and the operating system.

The configuration defined with HCD may consist of multiple processors, each containing multiple partitions. HCD stores the entire configuration data in a central repository, the input/output definition file (IODF). The IODF acts as single source for all hardware and software definitions for a multi-processor or multi-partition system and eliminates the need to maintain several independent MVSCP or IOCP data sets. That means that the information is entered only once using an interactive dialog.

Hardware Configuration Management (HCM) presents an interactive configuration diagram which allows to maintain not only the logical connectivity data in the IODF, but also the physical information about a configuration. The logical information in the IODF represents the operating system and the channel subsystem definitions; the physical information - cabinets, patchports, crossbar switches, cables, locations and so on - adds the infrastructure to the logical data.

Furthermore, the logical and physical information for each object in the configuration match because they are created by the same process. When creating an object, its logical and physical information is added at the same time. When connecting, for example, a controller to a processor, the selected control units are logically defined to the selected CHPID through a controller channel interface.

8.1.3 Hardware Support for Cryptographic Functions

8.1.3.1 CPACF

CP Assist Cryptographic Function (CPACF) is a microcode assist function plus hardware component implemented in a coprocessor (shared with a data compress function) accessed by a pair of PUs within the PU chip in an MCM. CPACF provides high performance encryption and decryption support. It is a hardware-synchronous implementation; that is, holding processor processing of the instruction flow until the operation completes. Several instructions are able to invoke the CPACF, when executed by the PU:

- KMAC Compute Message Authentic Code
- KM Cipher Message
- KMC Cipher Message with Chaining
- KMF Cipher Message with CFB
- KMCTR Cipher Message with Counter
- KMO Cipher Message with OFB
- KIMD Compute Intermediate Message Digest
- KLMD Compute Last Message Digest
- PCKMO Provide Cryptographic Key Management Operation

CPACF offers a set of symmetric (meaning that encryption and decryption processes use the same key) cryptographic functions that enhance the encryption and decryption performance of clear key operations of TLS, VPN, and data storing applications that do not require FIPS 140-2 level 4 security. Clear key means that the key used is located in central storage.

The following algorithms are available:

- Data encryption and decryption algorithms
    - Data Encryption Standard (DES)
        - Double-length key DES
        - Triple-length key DES (TDES)
    - Advanced Encryption Standard (AES) for 128-bit, 192-bit, and 256-bit keys
- Hashing algorithms: SHA-1, SHA-256, SHA-384, and SHA-512
- Message authentication code (MAC):
    - Single-key MAC
    - Double-key MAC
- Pseudo Random Number Generation (PRNG)

These functions are directly available to application programs, thereby diminishing programming overhead of going through ICSF. The CPACF complements, but does not execute, public key (PKA) functions. Note that keys, when needed, are to be provided in clear form only.

Traditionally CPACF was only able to execute clear-key cryptographic algorithms. This has changed with "Protected keys in CPACF", which is available for z196, z114 and newer processor models. The use of this feature requires support provided by ICSF, since the key management instructions for this feature can only be executed in supervisor state.

An enhancement to CPACF facilitates the continued privacy of cryptographic key material when used for data encryption. In CryptoExpress3, a secure key is encrypted under a master key, whereas a protected key is encrypted under a wrapping key that is unique to each LPAR.

When the wrapping key is unique to each LPAR, a protected key cannot be shared with another LPAR. CPACF, using key wrapping, ensures that key material is not visible to applications or operating systems during encryption operations.

CPACF code generates the wrapping key and stores it in the protected area of hardware system area (HSA). The wrapping key is accessible only by firmware. It cannot be accessed by operating systems or applications. DES/TDES and AES algorithms were implemented in CPACF code with support of hardware assist functions. Two variations of wrapping key are generated, one for DES/TDES keys and another for AES keys.

Wrapping keys are generated during the clear reset each time an LPAR is activated or reset. There is no customizable option available at SE or HMC that permits or avoids the wrapping key generation.

If a CEX3 coprocessor is available, a protected key can begin its life as a secure key. Otherwise, an application is responsible for creating or loading a clear key value and then using the new PCKMO instruction to wrap the key.

In the following figure, the source key is already stored in CKDS as a secure key (encrypted under the master key). This secure key is sent to CEX3C to be deciphered and sent to CPACF in clear text. At CPACF, the key is wrapped under the LPAR wrapping key and then it is returned to ICSF. After the key is wrapped, ICSF can keep the protected value in memory, passing it to the CPACF, where the key will be unwrapped for each encryption/decryption operation. The protected key is designed to provide substantial throughput improvements for a large volume of data encryption as well as low latency for encryption of small blocks of data. A high performance secure key solution, also known as a protected key solution, requires ICSF HCR7770, and it is highly desirable to use a CryptoExpress3 card.

Classification: Public
© Copyright IBM Corp. 2015

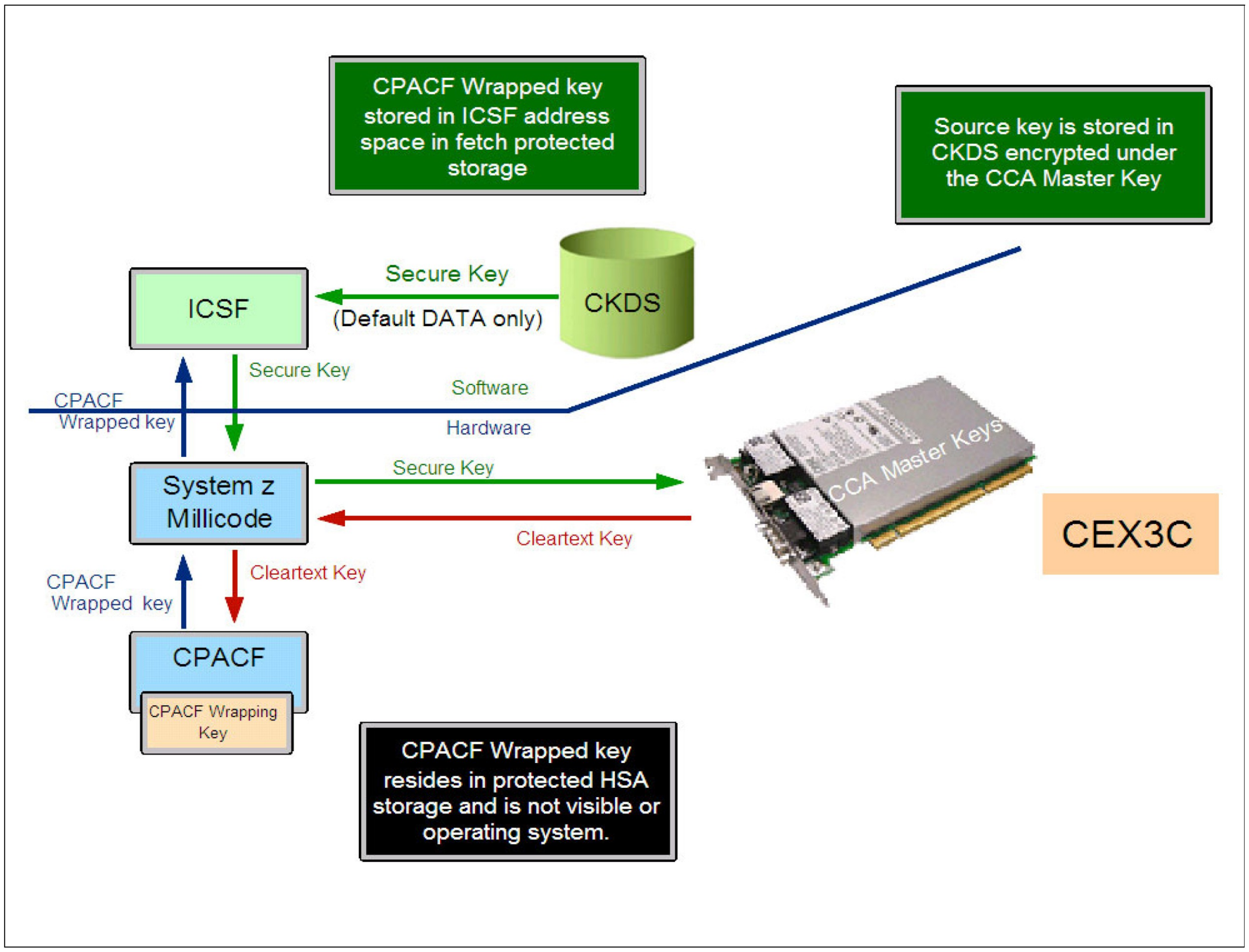

CPACF Wrapped key stored in ICSF address space in fetch protected storage
Source key is stored in CKDS encrypted under the CCA Master Key
Secure Key
(Default DATA only)
ICSF
CKDS
Secure Key
Software
Hardware
CPACF Wrapped key
System z Millicode
Secure Key
Cleartext Key
CCA Master Keys
CEX3C
CPACF Wrapped key
Cleartext Key
CPACF
CPACF Wrapping Key
CPACF Wrapped key resides in protected HSA storage and is not visible or operating system.

Figure 1 : Secure Key CPACF

8.1.3.2 Crypto Express3

The CryptoExpress3 feature is an outboard coprocessor. It is peripheral, being located in the I/O cage in one I/O card. To make it simple, it looks like a channel coupled to a cryptographic coprocessor. All its processing is asynchronous because the processor does not wait for the cryptographic request completion. The end of a request is indicated by turning on a bit in the HSA vector area. These bits are inspected by the z/OS dispatcher. CryptoExpress3 provides a high-performance cryptographic environment with added functions.

CryptoExpress3 represents the newest-generation cryptographic feature designed to complement the cryptographic functions of CPACF. The CryptoExpress3 resides in the I/O cage of the z196 (or z10 EC) or in the I/O drawer of z196 (or z10 BC), and continues to support all of the cryptographic functions available on the older CryptoExpress2.

CryptoExpress3 is a state-of-the-art, tamper-sensing and tamper-responding, programmable cryptographic feature. The cryptographic electronics and microprocessor provide a secure cryptographic environment using two PCI-Express (PCI-E) adapters. Each PCIe adapter contains dual processors that operate in parallel to support the IBM Common Cryptographic Architecture (CCA) with high reliability.

Each feature has two Peripheral Component Interconnect eXtended (PCI-X) cryptographic adapters. Each PCI-X cryptographic adapter can be configured as a cryptographic coprocessor or a cryptographic accelerator.

Each CryptoExpress3 feature may be configured as:

- Two PCI-E cryptographic coprocessors (default mode)
- One PCI-E cryptographic coprocessor and one PCI-E cryptographic accelerator
- Two PCI-E cryptographic accelerators

The z196 supports up to eight CryptoExpress3 features (up to sixteen PCI-X cryptographic adapters) to be installed.

The security-relevant portion of the cryptographic functions is performed inside the secure physical boundary of a tamper-resistant card. Master keys and other security-relevant information is also maintained inside this secure boundary.

It does not use CHPID, but requires one slot in the I/O cage and one PCHID for each PCI-X cryptographic adapter. The feature is attached to an IFB-MP and has no other external interfaces. Each PCI-X cryptographic adapter can be shared by any logical partition defined in the system, up to a maximum of 16 logical partitions per PCI-X cryptographic adapter.

The CryptoExpress3 coprocessor enables the user to:

- Encrypt and decrypt data utilizing secret-key (non-clear key) algorithms. Triple-length key DES and double-length key DES algorithms are supported.
- Generate, install, and distribute cryptographic keys securely, using both public and secret key cryptographic methods.
- Generate, verify, and translate personal identification numbers (PINs).
- Ensure the integrity of data by using message authentication codes (MACs), hashing algorithms, and Rivest-Shamir-Adleman (RSA) public key algorithm (PKA) digital signatures.

8.1.3.3 CryptoExpress4S

The new cryptographic coprocessor Crypto Express 4S can also be installed using a new firmware that implements PKCS#11 like functions. This firmware needs to be used for secure key functions that PKI Services and RACDCERT can use to operate with secure keys stored in the ICSF TKDS. Those keys are never exposed in clear even in ICSF. Their use adds an additional level of protection against users that have or may gain access to the data set storing the ICSF Token Key Data Set (TKDS).

8.1.4 I/O Subsystem

In addition to the main processor there is a dedicated I/O hardware subsystem, the “Channel” subsystem that allows I/O operations to be performed in parallel to the normal processor operation. Configuring and programming the I/O subsystem is restricted to programs operating in supervisor state.

The zSeries channel subsystem contains channels are able to communicate with I/O control units (CU) and manage the movement of data between central storage and the control units. They are located in I/O cards in the zSeries I/O cage. Being more specific, the channels can:

- Send channel commands from the memory to a CU
- Transfer data during read and write operations
- Receive status at the end of operations
- Receive sense information from control units, such as detailed error information

In z/Architecture, the I/O operation is created when a privileged START SUBCHANNEL (SSCH) instruction is executed by the input/output supervisor (IOS), a z/OS component, which issues the instruction on behalf of a task. It finishes when an I/O interrupt is received by the CPU, forcing the execution of the IOS component again.

8.1.4.1 Subchannels

Within the channel subsystem are subchannels. One subchannel is provided for and dedicated to each I/O device accessible to the channel subsystem. Each subchannel that has an I/O device assigned to it also contains a system-unique parameter called the device number. The device number is a 16-bit value that is assigned as one of the parameters of the subchannel at the time the device is assigned to the subchannel.

The device number provides a means to identify a device, independent of any limitations imposed by the system model, the configuration, or channel-path protocols. The device number is used in communications concerning the device that take place between the system and the system operator.

Each subchannel provides information concerning the associated I/O device and its attachment to the channel subsystem. The subchannel also provides information concerning I/O operations and other functions involving the associated I/O device.

The subchannel is the means by which the channel subsystem provides information about associated I/O devices to CPs, which obtain this information by executing I/O instructions.

The actual number of subchannels provided depends on the model and the configuration; the maximum addressability is 65,536.

The device number is stored in the UCW (in the Hardware System Area - HSA) at Power-on Reset (POR) and comes from the IOCDS. HSA is a piece of central storage not addressable by z/OS. It is allocated at Power-on Reset (POR) and contains LPAR LIC, microcode work areas, and the I/O configuration as UCWs used by the channel subsystem. For each pair device/logical partition, there is one UCW representing the device in HSA.

8.1.4.2 UCB

The device number is also stored in the UCB at IPL, to be used by IOS. A unit control block (UCB) holds information about an I/O device, such as:

- State information for the device

- Features of the device

The UCW represents the device from the channel subsystem point of view; an UCB represents the device from the IOS point of view.

8.1.4.3 Control units

A control unit provides the logical capabilities necessary to operate and control an I/O device and adapts the characteristics of each device so that it can respond to the standard form of control provided by the CSS. A control unit can be housed separately, or it can be physically and logically integrated with the I/O device, the channel subsystem, or within the server itself.

8.1.4.4 I/O devices

An I/O device provides external storage, a means of communication between data-processing systems, or a means of communication between a system and its environment. In the simplest case, an I/O device is attached to one control unit and is accessible through one channel path.

For more information on the I/O operation in a zSeries architecture see [ZARCH], chapters 13 to 17 and [ABC-VOL10], chapter 1.

8.2 z/OS Basic Principles

8.2.1 Address Spaces and Data Spaces

8.2.1.1 Addresses

z/Architecture has evolved from System /360 with 24-bit addresses (16 MB) over System / 370, 370-XA, ESA/370 with 31-bit addresses (2 GB) to z/Architecture with 64-bit addresses (16 Exabyte). Programs written for 24-bit or 31-bit addresses can still be executed when switching the processor into those old addressing modes.

A 64-bit address is interpreted as follows:

- the lower 12 bit represent the byte index into a page of 4K
- the next higher 8 bit represent a page index of up to 256 pages
- the next higher 11 bit represent a segment index of up to 2048 page indexes
- the next higher 11 bit represent the R3 index of up to 2048 segments
- the next higher 11 bit represent the R2 index of up to 2048 R3 indexes
- the next higher 11 bit represent the R1 index of up to 2048 R2 indexes

8.2.1.2 Dynamic Address Translation (DAT)

Address translation on z/Architecture can involve two steps. The first one, if dynamic address translation is turned on, involves the use of hierarchical page tables to convert a virtual address to a real address. The second one involves conversion of a real address to an absolute address using prefixing. If dynamic address translation is turned off, the address translation consists of just one step, that of converting a real address to an absolute address.

Bit 5 of the current PSW (DAT mode) indicates whether or not an effective address is to be translated using paging tables. If is equals ‘0’, the effective address provided is considered to be a real address rather than a virtual address. If DAT mode is set (i.e. Bit 5 of the PSW equals ‘1’), bits 16 and 17 of the PSW define the address translation mode used for the translation as follows:

Bit 16	**Bit 17**	**Address Translation Mode**
‘0’	‘0’	primary space
‘1’	‘0’	secondary space
‘0’	‘1’	access-register space
‘1’	‘1’	home space

Table 10 - z/Architecture Address Translation Modes

The following diagram illustrates the logic used to determine the translation mode.

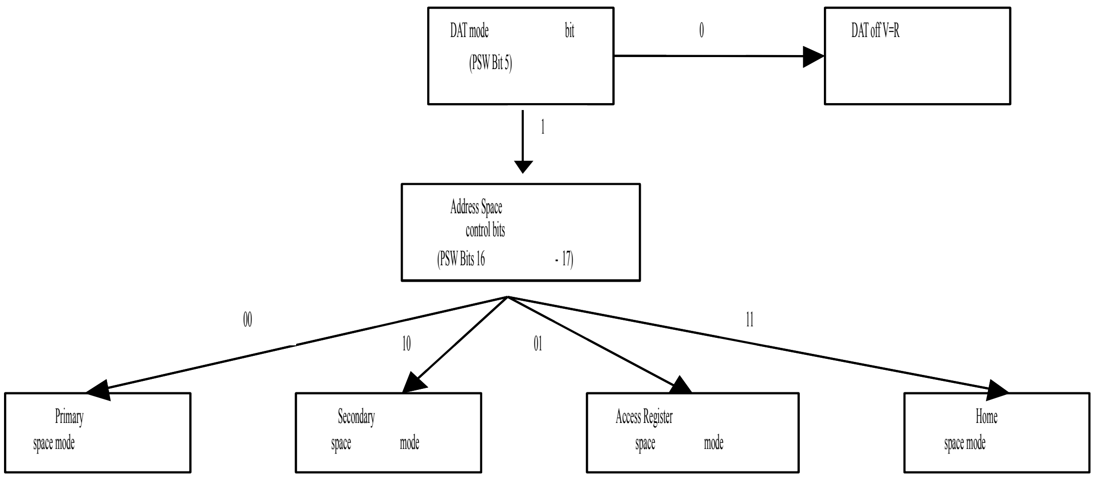


Figure 2 : z/Architecture Determination of Address Translation Mode

Page table protection

The page table protection mechanism is applied to virtual addresses during their translation to real addresses. The page table protection mechanism controls access to virtual storage by using the page-protection bit in each page-table entry and segment-table entry. Protection can be applied to a single page or an entire segment (a collection of contiguous pages). Once the ASCE is located, the following dynamic address translation is used to translate virtual address to a real address. Page table protection (for a page or a segment) is applied at this stage. The diagram illustrates the DAT process for 31-bit addresses (in black) and for the 64-bit addresses (in blue).

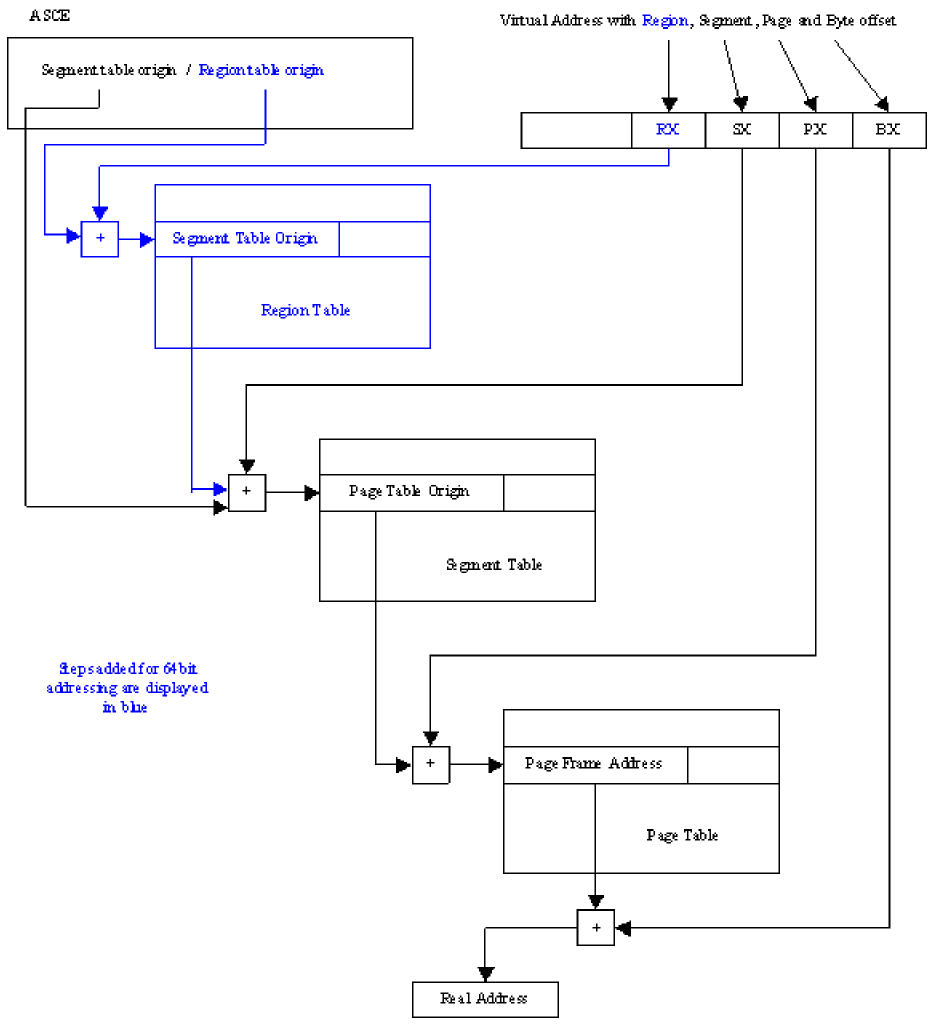

ASCE
Segment table origin / Region table origin
Virtual Address with Region, Segment, Page and Byte offset
RX
SX
PX
BX
+
Segment Table Origin
Region Table
Page Table Origin
Segment Table
Steps added for 64 bit addressing are displayed in blue
Page Frame Address
Page Table
Real Address

Figure 3 : z/Architecture Dynamic Address Translation

With z10 machines and higher are capable of supporting 1M pages. DAT was enhanced (EDAT1) as illustrated below:

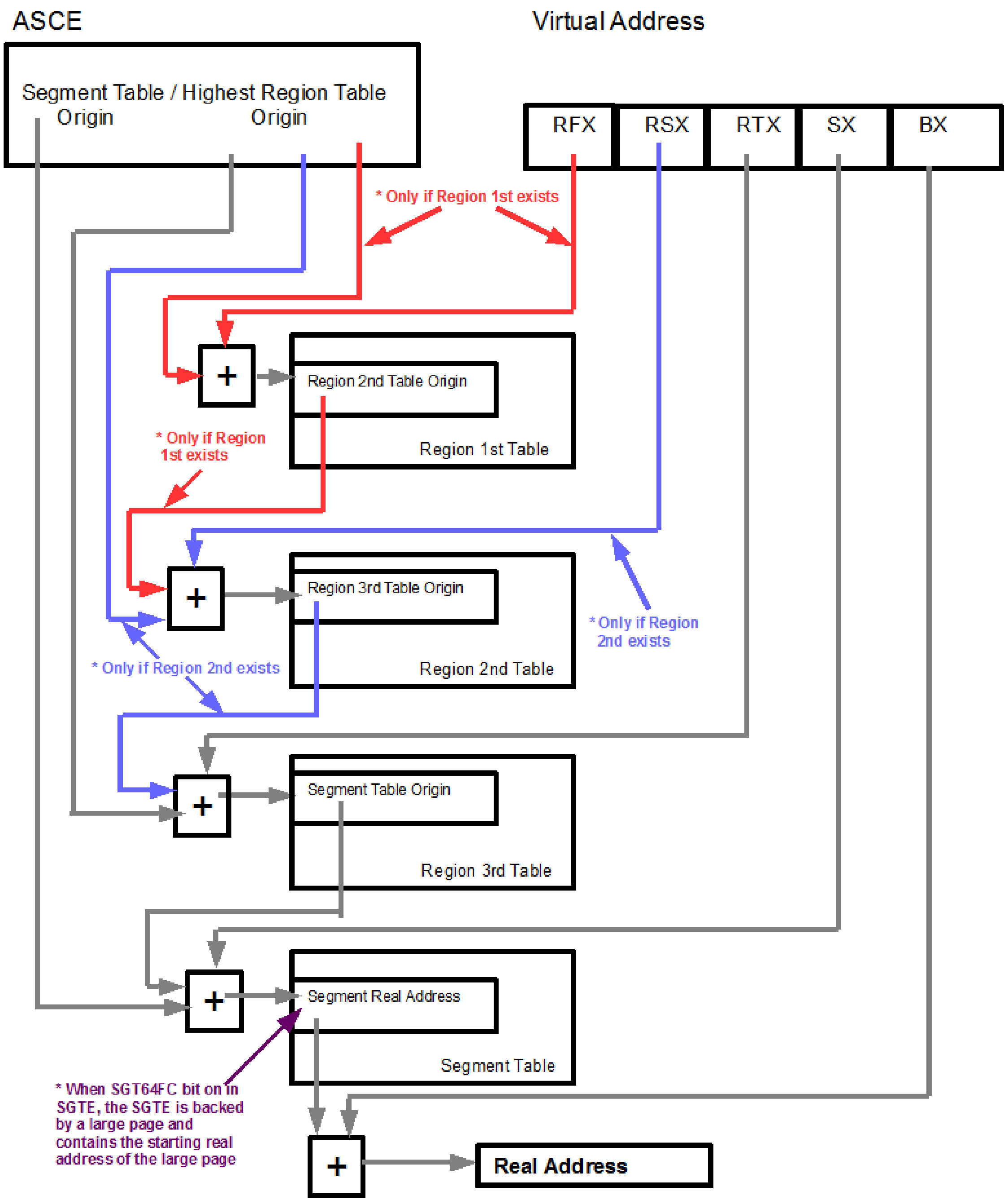

ASCE
Segment Table / Highest Region Table
Origin
Origin
Virtual Address
RFX
RSX
RTX
SX
BX
* Only if Region 1st exists
+
Region 2nd Table Origin
Region 1st Table
* Only if Region 1st exists
+
Region 3rd Table Origin
Region 2nd Table
* Only if Region 2nd exists
* Only if Region 2nd exists
+
Segment Table Origin
Region 3rd Table
+
Segment Real Address
Segment Table
* When SGT64FC bit on in SGTE, the SGTE is backed by a large page and contains the starting real address of the large page
+
Real Address

Figure 4 : z/Architecture EDAT1 process

With zEC12 and higher are capable of supporting 2G pages. EDAT1 was further enhanced (EDAT2) as illustrated below:

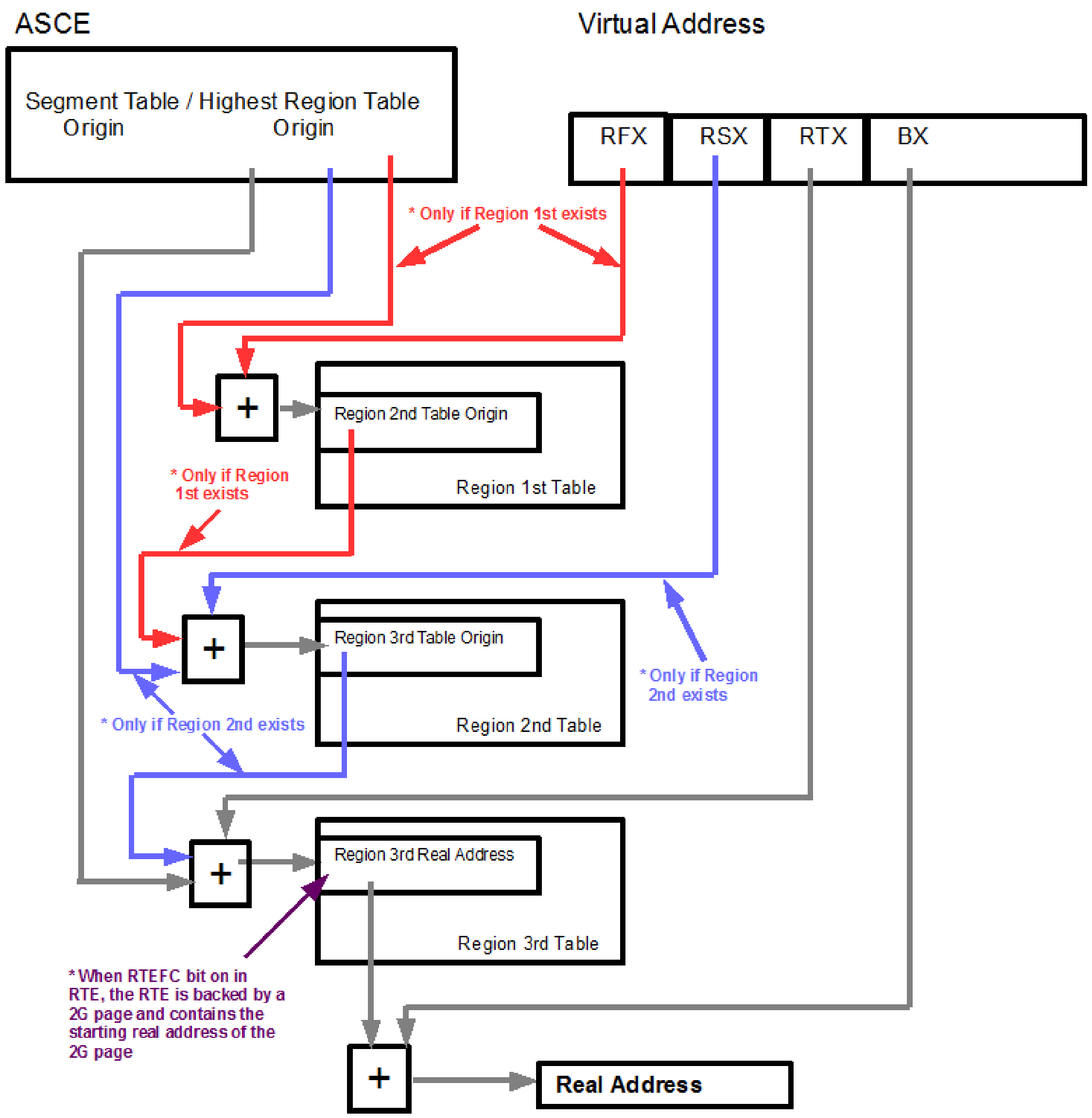

ASCE
Virtual Address
Segment Table / Highest Region Table
Origin Origin
RFX
RSX
RTX
BX
* Only if Region 1st exists
+
Region 2nd Table Origin
Region 1st Table
* Only if Region 1st exists
Region 3rd Table Origin
Region 2nd Table
* Only if Region 2nd exists
* Only if Region 2nd exists
Region 3rd Real Address
Region 3rd Table
* When RTEFC bit on in RTE, the RTE is backed by a 2G page and contains the starting real address of the 2G page
Real Address

Figure 5 :   z/Architecture EDAT2 process

Prefixing

Prefixing provides the ability to assign a range of real addresses to a different block in absolute memory for each CPU, thus permitting more than one CPU sharing main memory to operate concurrently with a minimum of interference. Prefixing is performed using a prefix register. No access control is performed while translating a real address to an absolute address.

Prefixing is typically used to allow each CPU to have a unique view of the first 8K of memory, where (for example) the PSWs are stored when interrupts happen, and where the new PSWs are obtained to process those interrupts.

8.2.2 Memory protection mechanisms

In addition to the separation of address spaces, the z/Architecture provides mechanisms to generally protect memory from unauthorized access. Memory protections are implemented using a combination of the Program Status Word (PSW) register, a set of sixteen control registers (CRs), and a set of sixteen access registers (ARs). The remainder of this section describes memory protection mechanisms and how they are implemented using the PSW, CRs, and ARs.

z/Architecture provides three mechanisms for protecting the contents of main memory from destruction or misuse by programs that contain errors or are unauthorized: low-address protection, page table protection, and key-controlled protection. The protection mechanisms are applied independently at different stages of address translation; access to main memory is only permitted when none of the mechanisms prohibit access.

Low-address protection is applied to effective addresses, page table protection is applied to virtual addresses while they are being translated into real addresses, and key-controlled protection is applied to absolute addresses.

Low-address protection

The low address protection mechanism provides protection against the destruction of main memory information used by the CPU during interrupt processing. Each CPU present in the system has its dedicated low-address range.

Protection of the address range is implemented by preventing instructions from writing to addresses in the ranges 0 through 511 and 4096 through 4607 (the first 512 bytes of each of the first and second 4K-byte address blocks).

Low-address protection is applied to effective addresses only if the following bit positions are set in control register 0 and the Address Space Control Element (ASCE) of the address space to which the effective address belongs.

Bit 35 of CR 0	Bit 55 of ASCE	Low-Address Protection
‘0’	‘0’	disabled
‘0’	‘1’	disabled
‘1’	‘0’	enabled
‘1’	‘1’	disabled

Table 11: z/Architecture Low-Address Protection Condition

Key-controlled protection

Key-controlled protection is applied when an access attempt is made for an absolute address, which refers to a memory location. Each 4K page, real memory location has a 7-bit storage key associated with it. These storage keys for pages can only be set when the processor is in the supervisor state. The Program Status Word contains an access key corresponding to the current running program. Key-controlled protection is based on using the access key and the storage key to evaluate whether access to a specific memory location is granted.

The 7-bit storage key consists of access control bits (0, 1,2, 3), fetch protection bit (4), reference bit (5), and change bit (6).

The following two diagrams describe the key-controlled protection.

Bit 39 of control register 0 is the storage protection override control. When this bit is one, storage-protection override is active. When this bit is zero, storage-protection override is inactive.

When storage-protection override is active, key-controlled storage protection is ignored for storage locations having an associated storage-key value of 9. When storage-protection override is inactive, no special action is taken for a storage-key value of 9.

Storage-protection override applies to instruction fetch and to the fetch and store accesses of instructions whose operand addresses are logical, virtual, or real. It does not apply to accesses made for the purpose of channel-program execution or for the purpose of channel-subsystem monitoring.

Storage-protection override has no effect on accesses, which are not subject to key-controlled protection.

Figure 6 **– z/Architecture Matching Access Key and Storage Key**

The z/Architecture allows for fetch protection override for key-controlled protection. The following diagram describes how fetch protection override can be used.

Figure 7 : **z/Architecture Fetch Protection Override**

For more information about dynamic address translation and memory protection, see [ZARCH].

8.2.2.1 Address Space in z/OS (section needs to be updated with the HLD)

An address space is a dedicated logical block of virtual storage. Each address space in z/OS has a similar structure. The structural elements consist of:

- The common area below 16 MB
- The private area below 16 MB
- The extended common area above 16 MB
- The extended private area above 16 MB
- Low user private above 2GB
- Local system area (32GB to 288GB)
- High common area above 2GB
- Shared Area above 2GB
- High virtual user private for USE2Gto32G requests 2GB-32GB
- High virtual user private (local system area 32GB-288GB)
- High virtual user private (low area)
- High virtual common area
- High virtual shared area
- High virtual user private area (high area)

Note: The 16MB boundary is called “The Line”, whereas the 2GB boundary is called “The Bar”.

The common area contains system control programs and control blocks. The following storage areas are located in the common area:

- Prefixed storage area (PSA)
- Common service area (CSA)
- Pageable link pack area (PLPA)
- Fixed link pack area (FLPA)
- Modified link pack area (MLPA)
- System queue area (SQA)
- Nucleus, which is fixed and nonswappable

Each storage area in the common area (below 16 MB) has a counterpart in the extended common area (above 16 MB) with the exception of the PSA. Each address space uses the same common area. Portions of the common area are paged in and out as the demands of the system change and as new user jobs (batch or time-shared) start and old ones terminate.

The private area contains:

- A local system queue area (LSQA)
- A scheduler work area (SWA)
- Subpools 229, 230, and 249 (the authorized user key area)
- A 16 KB system region area
- Either a V=V (virtual = virtual) or V=R (virtual = real) private user region for running programs and storing data

Except for the 16 KB system region area and V=R user regions, each storage area in the private area below 16 MB has a counterpart in the extended private area above 16 MB.

Each address space has its own unique private area allocation. The private area (except LSQA) is pageable unless a user specified a V=R region or the user explicitly asks to fix specific storage ranges. If assigned as V=R, the actual V=R region area (excluding SWA, the 16 KB system region area, and high private subpools) is fixed and nonswappable.

System Queue Area (SQA/Extended SQA)

This area contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage and number of private virtual storage address spaces are two of the factors that affect the system's use of SQA.

The SQA is allocated directly below the nucleus; the extended SQA is allocated directly above the extended nucleus.

Common Areas (CSA/Extended CSA/High virtual common)

This contain pageable and fixed data areas that are addressable by all active virtual storage address spaces. Common storage normally contains data referenced by a number of system address spaces, enabling address spaces to communicate by referencing the same piece of common storage data. CSA is allocated directly below the MLPA; extended CSA is allocated directly above the extended MLPA; high virtual common is allocated above high private and below high virtual shared. If the virtual SQA space is depleted, the system will allocate additional SQA space from the CSA.

High Virtual Shared Area

This area contains storage that can be obtained and shared with programs in other address spaces. By default memory objects in this area is not addressable by other address spaces but programs can request access to them by providing the same unique token that was used when the memory object was created.

Pageable Link Pack Area (PLPA/Extended LPA)

This area contains SVC routines, access methods, and other read-only system programs along with any read-only reenterable user programs selected by an installation that can be shared among users of the system. Any module in the pageable link pack area will be treated by the system as though it came from an APF-authorized library.

Modified Link Pack Area (MLPA/Extended MLPA)

This area may be used to contain reenterable routines from either APF-authorized or non-APF-authorized libraries that are to be part of the pageable extension to the link pack area during the current IPL. Any module in the modified link pack area will be treated by the system as though it came from an APF-authorized library.

Fixed Link Pack Area (FLPA/Extended FLPA)

An installation can elect to have some modules that are normally loaded in the pageable link pack area (PLPA) loaded into the fixed link pack area (FLPA). This area should be used only for modules that significantly increase performance when they are fixed rather than pageable. Modules placed in the FLPA must be reentrant and refreshable. Modules in FLPA are also treated as if they came from an APF library, like the other forms of LPA.

Local System Queue Area (LSQA/Extended LSQA)

Each virtual address space has an LSQA. The area contains tables and queues associated with the user’s ad-dress space. LSQA is intermixed in the high end of the private area with SWA and subpools 229, 230, and 249 downward from the bottom of the CSA into the unallocated portion of the private area, as needed. Extended LSQA is intermixed with SWA and sub-pools 229, 230, and 249 downward from 2 gigabytes into the unallocated portion of the extended private area, as needed.

Scheduler Work Area (SWA/Extended SWA)

This area contains control blocks that exist from task initiation to task termination. It includes control blocks and tables created during JCL interpretation and used by the initiator (see Volume 2, JES2 Overview) during job step scheduling. Each initiator has its own SWA within the user’s private area.

Nucleus (Extended Nucleus)

The nucleus area contains the nucleus load module and extensions to the nucleus that are initialized during IPL processing. The modules to be added to the nucleus region, or deleted from it, must reside in members of SYS1.NUCLEUS. The nucleus is always fixed storage.

Regions

The portion of the user’s private area within each virtual address space that is available to the user’s problem programs is called the user region. In z/OS, the storage available for a program is virtual storage or central storage (also called real storage):

Virtual storage is addressable space that appears to the user as central (real) storage. Instructions and data are mapped from virtual storage into central storage locations, where they are executed.

Central (real) storage is the storage from which the processor can directly obtain instructions and data and to which it can directly return results.

The virtual storage address space is 16 Exabytes. The address space contains the commonly addressable system storage, the nucleus, and the private address space, which includes the user’s region. When a program is selected, the system brings it into virtual storage and divides it into pages of 4K bytes. The system transfers the pages of a program into central (real) storage for execution and out to auxiliary storage when not needed. Paging is done automatically; to the programmer the entire program appears to occupy contiguous space in central storage at all times. Actually, not all pages of a program are necessarily in central storage at one time. Also, the pages that are in central storage do not necessarily occupy contiguous space.

Subpools 229, 230, 249

Subpools 229, 230, 249 - Extended 229, 230, 249 This area allows local storage within a virtual address space to be obtained in the requestor’s storage protect key. The area is used for control blocks that can be obtained only by authorized programs having appropriate storage protect keys. These control blocks were placed in storage by system components on behalf of the user. These subpools are intermixed with LSQA and SWA. Subpool 229 is fetch protected; subpools 230 and 249 are not. All three subpools are pageable. Subpools 229 and 230 are freed auto-matically at task termination; subpool 249 is freed automatically at jobstep task termination.

Storage Keys

The following tables summarizes how storage keys are used within z/OS.

Key	Used by
0	MVS system control program; control blocks and code
1	JES2 and JES3
2	IBM’s Virtual Storage Personal Computing (VSPC) subsystem
3	(reserved)
4	(reserved)
5	Data management for input/output
6	TCAM and VTAM services routines
7	Database/data communications subsystem; e.g. IMS
8-F	All batch jobs, TSO sessions, and virtual storage user programs
9	Used by CICS and specific hardware modifications
A-F	Application programs

Table 12 – Storage Key Summary

8.2.2.2 Data Spaces and Hiperspaces

Data spaces and hiperspaces are data-only spaces that can hold up to 2 gigabytes of data. They provide integrity and isolation for the data they contain in much the same way as address spaces provide integrity and isolation for the code and data they contain. They are an extremely flexible solution to problems related to accessing large amounts of data. There are two basic ways to place data in a data space or a hiperspace. One way is through buffers in the program’s address space. Another way avoids using address space virtual storage as an intermediate buffer area: through data-in-virtual services, a program can move data into a data space or hiperspace directly. For hiperspaces, this second way reduces the amount of I/O.

Programs that use data spaces run in AR ASC mode. They use MVS macros to create, control, and delete data spaces. Assembler instructions executing in the address space directly manipulate data that resides in data spaces.

Programs that use hiperspaces run in primary or AR ASC mode. They use MVS macros to create, control, and delete hiperspaces. Programs cannot directly manipulate data in a hiperspace, but use MVS macros to transfer data to and from the hiperspace for data manipulation. Hiperspaces provide high-speed access to large amounts of data.

Data Spaces

A data space is a range of up to two gigabytes of contiguous virtual storage addresses that a program can directly manipulate through assembler instructions. Unlike an address space, a data space contains only data; it does not contain common areas or system data or programs. Program code does not execute in a data space, although a program can reside in a data space as non-executable code.

A program’s ability to create, delete, and access data spaces depends on whether it is a problem state program with PSW key 8 - F, a supervisor state program, or a PSW key 0-7 program. All programs can create, access, and delete the data spaces they own or created, and can share their data spaces with other programs running in the same address space. In addition, supervisor state or PSW key 0-7 programs can share their data spaces with programs in other address spaces. Unless otherwise stated, this chapter describes what the supervisor state or PSW key 0-7 programs can do.

Hiperspaces

A hiperspace is a range of up to two gigabytes of contiguous virtual storage addresses that a program can use as a buffer. Like a data space, a hiperspace holds only data, not common areas or system data; code does not execute in a hiperspace. Unlike a data space, data is not directly addressable.

To manipulate data in a hiperspace, the accessing program brings the data, in blocks of 4K bytes, into a buffer area in its address space. The program can use the data only while it is in the address space. This buffer area can be though of as a “view” into the hiperspace.

The data in the hiperspace and the buffer area in the address space must both start on a 4K byte boundary. A program would use a hiperspace rather than a data space if the program needs an area outside the address space primarily for storage purposes, and not for data manipulation.

Memory Subpools

Conceptually, a subpool is an area of virtual storage with a certain set of attributes, created by z/OS to satisfy a re-quest for virtual storage. A two-gigabyte virtual storage address space is divided into multiple areas, each intended to support specific system and user needs. z/OS provides subpools within the following areas of the address space:

- Low private

- Extended private
- Private local system queue area (LSQA) and extended LSQA
- Common system queue area (SQA) and extended SQA
- Common service area (CSA) and extended CSA.

Besides being in a particular area in an address space, subpools are distinguished by other attributes, such as which programs can access them, whether the storage can be paged out, and how long the storage persists.

More details about memory subpools can be found in [AUTHASSEM], section 10.

Real Storage Management

The task of the Real Storage Manager (RSM) is to control the usage of real storage frames. RSM acts together with the auxiliary storage manager to support the virtual storage concept, and with VSM to ensure that a page obtained with the GETMAIN SVC will be backed up with a real storage frame. Furthermore, RSM establishes many services to other components and application programs to manipulate the status of pages and frames.

RSM controls the allocation of central storage during initialization and pages in user or system functions for execution. Some RSM functions are to:

- Allocate central storage to satisfy GETMAIN requests for SQA and LSQA
- Allocate central storage for page fixing
- Allocate central storage for an address space that is to be swapped in

If there is storage above 16 megabytes, RSM allocates central storage locations above 16 megabytes for SQA, LSQA, and the pageable requirement of the system. When non-fixed pages are fixed for the first time, RSM:

- Ensures that the pages occupy the appropriate type of frame
- Fixes the pages and records the type of frame used

Virtual Storage Management

Virtual storage is managed by the virtual storage manager (VSM). Its main function is to distribute the virtual storage among all requests.

Virtual storage is requested with the GETMAIN or STORAGE OBTAIN macro and returned to the virtual storage manager with the FREEMAIN or STORAGE RELEASE macro.

Virtual storage is normally larger than main storage (called real storage in z/OS). The size of real storage depends on the CPU type.

A z/OS program resides in virtual storage and only parts of the program currently active need to be in real storage at processing time. The inactive parts are held in auxiliary storage. An active virtual storage page resides in a real storage frame. An inactive virtual storage page resides in auxiliary storage. Moving pages between frames and slots is called paging.

Auxiliary Storage Management

Auxiliary storage manager (ASM) is the component of the z/OS system that is responsible for transferring virtual pages between real and auxiliary storage. ASM manages the transfer by initiating the I/O and by maintaining tables to reflect the current status of auxiliary storage.

Enough auxiliary storage must be available for the programs and data that comprise the system. Auxiliary storage may be comprised of the following storage mediums:

- Paging data sets on Direct Access Storage Devices (DASD)
- Storage-Class Memory (SCM)

8.2.2.3 Paging

The paging controllers of the auxiliary storage manager (ASM) attempt to maximize I/O efficiency by incorporating a set of algorithms to distribute the I/O load as evenly as is practical. In addition, every effort is made to keep the system operable in situations where a shortage of a specific type of page space exists.

To page efficiently and expediently, ASM divides the pages of the system into classes, namely PLPA, common, and local. Contention is reduced when these classes of pages are placed on different physical devices. Although the system requires only one local page data set, performance improvement can be obtained when local page data sets are distributed on more than one device, even though some devices may be large enough to hold the entire amount of necessary page space.

The PLPA and common page data sets are both optional, and there can be only one of each. Spillage back and forth between the PLPA and common page data sets is permissible. It is also possible for either page data set to spill to a local page data set. The page data sets are:

- PLPA page data set: this contains pageable link pack area. (There is only one.)
- Common page data set: this contains the non-PLPA virtual pages of the system common area. (There is only one.)
- Local page data set: this contains the private area (address space) pages, and space for any VIO data sets. Local page data sets can be dynamically added to and deleted from the paging configuration without re-IPLing the system. (There are one or more.)
- These page data sets may contain PLPA and/or common area pages as well if there is no room on the PLPA and Common page data sets.   High Virtual Common storage and Shared storage also goes here.

8.2.3 Executable Entities

8.2.3.1 Tasks and Programs

For a program to execute in z/OS it must be associated with a task. A task is a separately dispatchable unit of work, to which the dispatcher can assign the control of a processor. Tasks are represented by a control block called the TCB. In a z/OS user address space there will be multiple tasks; there are three “system” tasks established by z/OS, as well as one or more tasks representing the application. Each task has a chain of one or more request blocks that describe services and programs that have to run. The z/OS tasks in an address space are there for housekeeping reasons and are usually inactive most of the time. They are:

- Region Control Task (RCT) – handles swap-out and swap-in of this user address space
- System Dump Task (DUMP) – takes system dumps of this user address space
- Started Task Control (STC) – prepares the address space to support initiation of a user program

Below the three system tasks lie the tasks representing the user applications. The structure depends on the type of address space.

- For a batch job, a batch initiator task is followed by the job step task, which represents the currently executing application program for the step
- For a started task (i.e. VTAM) – the initiator task followed by the application program
- For a TSO user – the terminal monitor program, followed by a task for the command being executed (see Volume 2, TSO/E Overview for more information about TSO/E)
- For a MOUNT command – a task for the mount processor
- For an APPC started address space – the tasks are similar to those for batch (initiator and transaction program)

The following picture illustrates this structure.

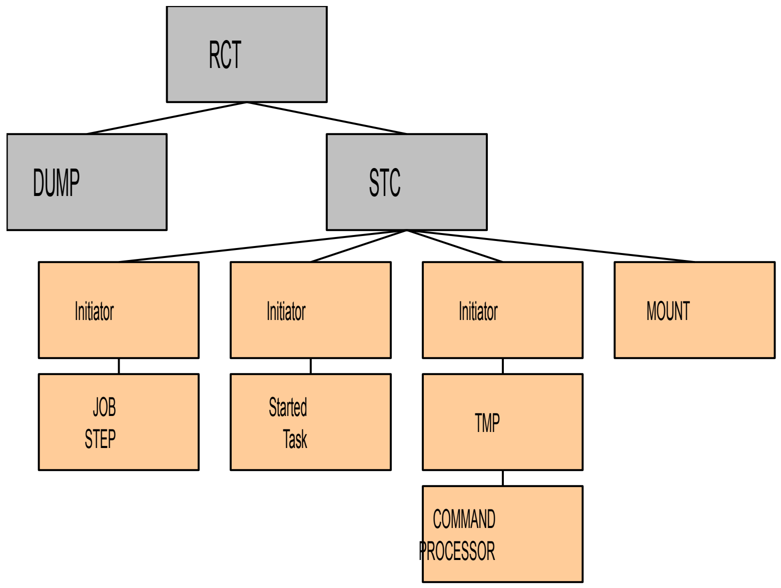


Figure 8 : Address Space Task Structure

The following control blocks are relevant to the z/OS task handling:

- TCB - Task Control Block; represents a task; contains pointer to current request blocks, pointers to related tasks' TCBs, pointers to related I/O structures, task status flags, task register save area, pointer to ECB (Event Control Block) used to post when the task is completed (optional), pointer to LLE (see previous page), pointer to related STCB
- STCB - Secondary TCB; save area for high words of 64-bit registers; pointers for UNIX System Services control blocks, etc.
- PRB - Program Request Block; contains current status information for a program currently being run, a register save area, address of CDE for program
- SRB – Service Request Block; represent "requests to execute a service routine" and are usually initiated by system code executing from one address space to perform actions affecting another address space. SRBs are created using the SCHEDULE macro by units of work running in supervisor state and system key. SRBs are non-preemptive, which means that when they are interrupted control must be returned immediately after the interruption has been processed.
- Contents Management control blocks – to keep track of programs in memory: where they are, their attributes, and how many users are sharing the program. z/OS uses this information to see if a program needs to be loaded into memory and relocated, or if an existing copy in memory is available to use Application programs can request contents management services, too, for example to dynamically load subroutines or to find out information about an executable:
    - XTLST - extent list; contains starting address and length of an executable in memory

- CDE – Contents Directory Entry; contains member name, entry point address, module attributes, address of the related XTLST, count of number of current users, and address of next element on the CDE chain
- LLE - Load List Element; contains information about executable programs LOADed by the main program (number of current LOAD requests for the program, pointer to the CDE for the program, address of next element on the list)

The following picture gives an overview of the relation of address spaces, tasks, programs and content management with regard to control blocks.

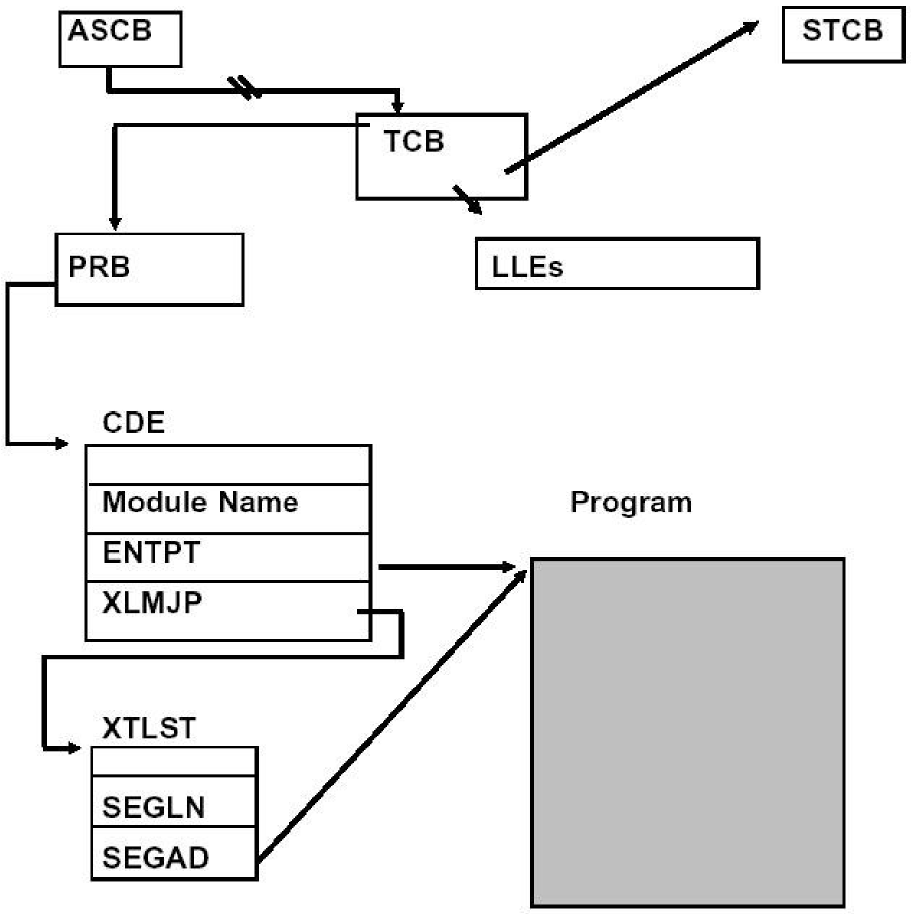


Figure 9 : Address Spaces, Tasks and Programs

8.2.3.2 Dispatching

Dispatchable entities are organized in so-called queues (chains of dispatchable elements). The elements are rep-resented by control blocks that in in turn represent address spaces and units of work.

The address space control block (ASCB) ready queue is a chain of those ASCBs, that are swapped in and contain at least one TCB or SRB that is ready to execute. The ASCB ready queue is ordered by priority, that means that the ASCB with the highest priority comes first in the queue.

Each address space has its own queues of SRBs and TCBs, these queues are linked by the ASCB. Whenever an event that changes the status of an address space completes, the relevant BCP function updates the dispatching queue to reflect that.

Whenever the dispatcher receives control after a unit of work has been terminated or is being interrupted, it selects the highest priority unit of work that is ready to execute. The dispatcher applies the following general order:

1. Select any special exit routines (e.g. hardware recovery)
2. Select SRBs that have global priority
3. Select the highest priority ASCB
4. Select ready SRBs within the selected address space
5. Select ready TCBs within the selected address space, if there are no ready SRBs
6. If there is no work ready to execute, the dispatcher loads an “enabled wait PSW”, otherwise it builds a PSW for the selected unit of work and loads it.

8.2.4 Authorized Programs

In addition to supervisor and PC routines, z/OS has a number of “authorized programs” that need to be trusted because they are not restricted by the security policy defined in this Security Target. An authorized program may call a number of program calls or supervisor calls or use supervisor call parameters that are reserved for authorized programs. In particular, it is authorized to call the MODESET SVC used to switch into supervisor state. With this function, authorized programs can execute any privileged instruction.

A program is authorized if at least one of the following conditions is true:

The program is executing in supervisor state {SP.3::SP.3.1}

The program is executing with a PSW key of 0 to 7 or a PSW key mask value that supports at least one key in the range of 0 to 7 in control register 3 {SP.3::SP.3.2}.

The authorization bit is set in the Job Step Control Block (JSCB) under which the program is executing {SP.3::SP.3.3}.

Whenever a supervisor routine reserved for authorized programs is called or when a parameter reserved for authorized programs is used, the routine invoked to service the request checks if one of the above listed conditions is satisfied. Only if this is true, the request is honored {SP.3::SP.3.4}. Note that the hardware performs some checks when a supervisor routine is called with a Program Call (PC) instruction. In this case the routine implementing the service only needs to perform its own checks if additional restrictions to those implied by the hardware checks apply. Note also that some supervisor routine may be more restrictive, i. e. only a subset of the three conditions mentioned above is checked and the request is rejected if not one of the conditions in the subset apply. For example the hardware can not check if a program running in problem state with a PSW key of 8 is authorized by the authorization bit in the JSCB.

An authorized program can be started in one of the following ways:

- By starting a program from a dedicated program library (defined in the system configuration data set SYS1.PARMLIB) that has the authorization bit set in the directory entry of the member of the partitioned data set (library) containing the program. This program has to be the one started with the EXEC JCL statement of the job step, as a TSO command, as a UNIX process using exec(), or started as a dedicated task by an authorized program using the ATTACH supervisor call with parameters reserved for authorized programs {SP.3::SP.3.5}

  Note: TSO commands might be entered directly by the user at a terminal, executed in a batch job that runs TSO TMP, or entered programmatically using the TSO IKJEFTSR service. They may also be executed by any service that uses either the TMP or IKJEFTSR service such as the REXX 'address tso' function or the Unix shell 'tsocmd' function.
- By starting a started task from an authorized library using the operator START command {SP.3::SP.3.6}
- By starting an authorized program from a zFS file system {SP.3::SP.3.V1R7.1}. A program in a zFS file system is authorized when the authorization bit has been set using the extattr –a command for the file containing the program {SP.3::SP.3.V1R7.2}. A user needs to have been authorized to the BPX.FILEATTR.APF profile in the FACILITY class to set the authorization bit {SP.3::SP.3.V1R7.3}. If a program running in an APF-authorized address space attempts to load a program from zFS that does not have the APF-extended attribute set, the load is rejected {SP.3::SP.3.V1R7.4}. Sanction lists can be defined that restrict access of authorized programs in the z/OS Unix System Services environment to files and directories defined in those sanction lists {SP.3::SP.3.V1R7.5}. Note that the APF-authorized extended attribute of a file is not honored if the file system containing the file has been mounted as NOSECURITY {SP.3::SP.3.V2R1.1}.

For the invocation of MVS programs linkedited AC=1 found in an APF-authorized library invoked via the z/OS UNIX spawn, exec and attach_exec service and for MVS load library programs that are to run as a z/OS UNIX set-user-id or set-group-id program the following rules apply:

- If the z/OS UNIX path name supplied to spawn, exec or attach_exec represents an external link that resolves to a MVS program found in an APF-authorized library and linkedited with the AC=1 attribute, the external link must have a owning UID of 0 and not be found in a file system mounted as NOSECURITY to allow this type of invocation {SP.3::SP.3.V2R1.1}.
- If the z/OS UNIX path name supplied to spawn, exec, or attach_exec represents a regular file with the sticky bit attribute that resolves to a MVS program found in an APF-authorized library and linkedited with the AC=1 attribute, the sticky bit file must have an owning UID of 0 or have the APF extended attribute turned on to allow this type of invocation. Additionally, the sticky bit file must not be found in a file system mounted as NOSECURITY to allow this type of invocation {SP.3::SP.3.V2R1.2}.

- If the z/OS UNIX path name supplied to spawn, exec or attach_exec represents a symbolic link to a regular file with the sticky bit attribute and the sticky bit file has the set-user-id attribute, the symbolic link must have an owning uid of 0 or an owning uid equal to that of the sticky bit file. If the sticky bit file has the set-group-id attribute, the symbolic link must have an owning uid of 0 or an owning gid equal to that of the sticky bit file. Additionally, the symbolic link must not be found in a file system mounted as NOSECURITY to allow this type of invocation {SP.3::SP.3.V2R1.3}.

Libraries that can contain authorized programs need to be protected from unauthorized modifications including the possibility to add new programs to the library. zFS files containing authorized programs also need to be protected from unauthorized modifications. The discretionary access control features of z/OS have to be used to protect those libraries.

The IKJTSOxx member of SYS1.PARMLIB can be used to define the authorized programs and commands that can be executed in the TSO environment {SP.3::SP.3.V1R7.6}.

Some trusted subsystems of z/OS are started as part of the standard startup procedure or may be later started by explicit request of a properly authorized user.

8.2.4.1 Protection of authorized programs

Authorized programs need to be trusted because they are allowed to increase their privileges up to running in supervisor mode with a storage key of zero. Authorized programs therefore must be carefully protected from unauthorized modification and the system must be protected from adding authorized programs other than those allowed in the evaluated configuration.

A program executes with authorization when:

- the program was linked with an authorization code into an authorized library or assigned the authorization attribute in the zFS file system and
- the program is the first program started within a job step or is started as an authorized TSO command. All programs started within the same job step by this program also run authorized {SP.3::SP.3.7}.

To protect the integrity of the TOE the following security measures must be in place:

- all program libraries that are authorized libraries must be protected from update or alter access by other than the system administrators using the discretionary access control functions and
- the system configuration library needs to be protected from any modification by other than the system administrators using the discretionary access control functions

No program other than the programs allowed in the evaluated configuration should be linked with an authorization code in the authorized libraries or specified in the PPT as having a system key or supervisor state.

Note that once a job step is authorized all programs called as part of the execution of the job step run with authorization and need to be trusted. The TOE protects trusted programs from accidentally executing any program from an untrusted library {SP.3::SP.3.8}. Trusted programs can take deliberate actions to bypass this protection.

Note that when within a non-authorized (untrusted) job step a program linked with authorization code into an authorized library is called, the program executes without authorization and will fail if it attempts to use privileges allowed only for programs executing with authorization {SP.3::SP.3.9}.

8.2.5 System Initialization (IPL)

The system initialization process prepares the system control program and its environment to do work for the installation. The process essentially consists of:

- System and storage initialization, including the creation of system component address spaces.
- Master scheduler initialization and subsystem initialization.

When the system is initialized and the job entry subsystem is active, the installation can submit jobs for processing with the START or MOUNT command.

The initialization process begins when the system operator selects the LOAD function at the system console. MVS locates all of the usable central storage that is online and available to the system, and creates a virtual environment for the building of various system areas. IPL includes the following major initialization functions:

- Loads the DAT-off nucleus into central storage.
- Loads the DAT-on nucleus into virtual storage so that it spans above and below 16 megabytes (except the prefixed storage area (PSA), which IPL loads at virtual zero).
- Builds the nucleus map, NUCMAP, of the DAT-on nucleus. NUCMAP resides in virtual storage above the nucleus.
- Allocates the system’s minimum virtual storage for the system queue area (SQA) and the extended SQA.
- Allocates virtual storage for the extended local system queue area (extended LSQA) for the master scheduler address space.

The system continues the initialization process, interpreting and acting on the system parameters that were specified. NIP (Nucleus Initialization Program) carries out the following major initialization functions:

- Expands the SQA and the extended SQA by the amounts specified on the SQA system parameter.
- Creates the pageable link pack area (PLPA) and the extended PLPA for a cold start IPL; resets tables to match an existing PLPA and extended PLPA for a quick start or a warm start IPL.
- Loads modules into the fixed link pack area (FLPA) or the extended FLPA. Note that NIP carries out this function only if the FIX system parameter is specified.
- Loads modules into the modified link pack area (MLPA) and the extended MLPA. Note that NIP carries out this function only if the MLPA system parameter is specified.

- Allocates virtual storage for the common service area (CSA) and the extended CSA. The amount of storage allocated depends on the values specified on the CSA system parameter at IPL.
- Page protects the: NUCMAP, PLPA and extended PLPA, MLPA and extended MLPA, FLPA and extended FLPA, and portions of the nucleus.

  Note: An installation can override page protection of the MLPA and FLPA by specifying NOPROT on the MLPA and FIX system parameters.

8.2.5.1 Initial System Address Space Creation

In addition to initializing system areas, z/OS establishes system component address spaces. z/OS establishes an address space for the master scheduler (the master scheduler address space (MSTR)) and other system address spaces for various subsystems and system components. Details about the initial address space creation can be found in [MVSTUNE.G], chapter 1.

8.2.5.2 Master Scheduler Initialization (MSTR)

Master scheduler initialization routines initialize system services such as the system log and communications task, and start the master scheduler itself. They also cause creation of the system address space for the job entry subsystem (JES2), and then start the job entry subsystem.

8.2.5.3 z/OS Subsystem Initialization

z/OS subsystem initialization is the process of readying a subsystem for use in the system. IEFSSNxx members of SYS1.PARMLIB contain the definitions for the primary subsystems, such as JES2, and possibly the secondary subsystems, such as SMS and DB2. For detailed information about the data contained in IEFSSNxx members for secondary systems, please refer to the installation manual for the specific system.

During system initialization, the defined subsystems are initialized. The system administrator should define the primary subsystem (JES2) first, because other subsystems require the services of the primary subsystem in their initialization routines – problems can occur if subsystems that use the primary subsystem’s services in their initialization routines are initialized before the primary subsystem.

After the primary subsystem JES2 is initialized, then the subsystems are initialized in the order in which the IEFSSNxx parmlib members are specified by the SSN parameter. For example, for SSN=(aa,bb) parmlib member IEFSSNaa would be processed before IEFSSNbb.

Note: The storage management subsystem (SMS) is the only subsystem that can be defined before the primary subsystem.

Using IEFSSNxx to initialize the subsystems, the system administrator can specify the name of a subsystem initialization routine to be given control during master scheduler initialization, and the system administrator can specify the input parameter to be passed to the subsystem initialization routine.

8.2.5.4 START/LOGON/MOUNT Processing

After the system is initialized and the job entry subsystem is active, jobs may be submitted for processing. When a job is activated through START (for batch jobs), LOGON (for time-sharing jobs) or MOUNT, a new address space must be allocated. Note that before LOGON, the operator must have started TCAM or VTAM/TCAS, which have their own address spaces.

The system resources manager decides, based on resource availability, whether a new address space can be created. If not, the new address space will not be created until the system resources manager finds conditions suitable.

8.2.5.5 System Configuration

z/OS systems can be configured in nearly every aspect. Therefore, it provides a number of configuration parameters. However, the mechanisms to implement this configuration are actually quite limited: The one central mechanism to manifest the configuration of a z/OS system is the SYS1.PARMLIB dataset. [MVSTUNE.G] provides information about system configuration and SYS1.PARMLIB to the extent of 700 pages. Please use this book if you are interested in details.

The following list gives a brief overview of the configuration parameters provided by SYS1.PARMLIB together with a very short description of its purpose.

Name	Purpose
ADYSETxx	Parameters that control dump analysis and elimination (DAE) processing.
ALLOCxx	Parameters that control installation defaults for allocation values.
APPCPMxx	Parameters that define or modify the APPC/MVS configuration.
ASCHPMxx	Parameters that define scheduling information for the ASCH transaction
BLSCECT	Parameters that control dump formatting performed by IPCS, and by the system (through the ABEND and SNAP macros).
BLSCUSER	Parameters that specify IPCS customization.
BPXPRMxx	Parameters that control the OS/390 UNIX System Services environment and the hierarchical file system (HFS). The system uses these values when initializing the OS/390 UNIX System Services kernel.
CLOCKxx	Parameters that control operator prompting to set the TOD clock, specifying the difference between the local time and GMT, and ETR usage.

Name	Purpose
CNGRPxx	Parameters that define console groups as candidates for switch selection if a console fails.
CNLcccxx	Defines how translated messages are to be displayed at the installation.
COFDLFxx	Allows a program to store DLF objects that can be shared by many jobs in virtual storage managed by Hiperbatch™.
COFVLFxx	Allows an authorized program to store named objects in virtual storage managed by VLF.
COMMNDxx	Commands to be issued by the control program immediately after initialization. JES commands may not be included.
CONFIGxx	Allows the installation to define a standard configuration that is compared with current configuration and to reconfigure processors, storage, and channel paths.
CONSOLxx	Parameters to define an installation’s console configuration, initialization values for communications tasks, the default routing codes for all WTO/WTOR messages that have none assigned, and the characteristics of the hardcopy message set. CONSOLxx also contains parameters that define the hardcopy medium and designate the alternate console group for hardcopy recovery.
COUPLExx	Describes the systems complex (sysplex) environment for the system.
CSVLLAxx	Allows an installation to list the entry point name or LNKLST libraries that can be refreshed by the ‘MODIFY LLA, UPDATE=xx’ command.
CSVRTLxx	Defines the run-time library services (RTLS) configuration. CSVRTLxx can be used to specify names of libraries to be managed, as well as storage limits for caching modules from the libraries.
CTncccxx	Specifies component trace options.
CUNUNIxx	Specifies parameters for Unicode conversion services.
DEVSUPxx	Allows an installation to specify whether data will be stored in a compacted format on a 3480 or 3490 tape subsystem with the Improved Data Recording

Name	Purpose
	Capability feature.
DIAGxx	Contains diagnostic commands that control the common storage tracking and GETMAIN/FREEMAIN/STORAGE (GFS) trace functions.
EPHWP00	Allows z/OS UNIX to use man pages.
EXITxx	Allows an installation to specify installation exits for allocation decisions.
EXSPATxx	Allows an installation to specify actions taken to recover from excessive spin conditions without operator involvement.
GRSCNFxx	Configuration parameters for systems that are to share resources in a global resource serialization complex.
GRSRNLxx	Resource name lists (RNLs) that the system uses when a global resource serialization complex is active.
GTFPARM	Parameters to control GTF.
GTZPRMxx	Parameters to control Generic Tracker.
HZSPRMxx	Parameters to control Health Checker.
IEAABD00	Default parameters for an ABEND dump when a SYSABEND DD statement has been specified.
IEAAPFxx	Names of authorized program libraries.
IEAAPP00	Names of authorized installation-written I/O appendage routines.
IEACMD00	IBM-supplied commands that are processed during system initialization.
IEADMCxx	Parameters for DUMP command.
IEADMP00	Default parameters for an ABEND dump when SYSUDUMP DD statement has
IEADMR00	Default parameters for an ABEND dump when SYSMDUMP DD statement has

Name	Purpose
	been specified.
IEAFIXxx	Names of modules to be fixed in central storage for the duration of the IPL.
IEAICSxx	Parameters of an installation control specification that associates units of work (transactions) with performance groups. Performance groups are used by the system resources manager to control and report on transactions.
IEAIPSxx	Parameters of an installation performance specification that control workload manager of system resources manager.
IEALPAxx	Names of reenterable modules that are to be loaded as a temporary extension to the PLPA.
IEAOPTxx	Parameters that control resource and workload management algorithms in the system resources manager.
IEAPAKxx	“Pack List” names of groups of modules in the LPALST concatenation that the system will load between page boundaries to minimize page faults.
IEASLPxx	Contains valid SLIP commands.
IEASVCxx	Allows the installation to define its own SVCs in the SVC table.
IEASYMxx	Specifies, for one or more systems in a multisystem environment, the static system symbols and suffixes of IEASYSxx members that the system is to use. One or more IEASYMxx members are selected using the IEASYM parameter in the LOADxx parmlib member.
IEASYSxx	System parameters that are valid responses to the SPECIFY SYSTEM PARAMETERS message. Multiple system parameter lists are valid. The list is chosen by the operator SYSP parameter or through the SYSPARM statement of the LOADxx parmlib member
IECIOSxx	Parameters that control missing interrupt handler (MIH) intervals and update hot I/O detection table (HIDT) values.
IEFSSNxx	Parameters that identify what subsystems are to be initialized.
IFAPRDxx	Parameters that define a product enablement policy.

Name	Purpose
IGDDFPKG	One or more statements that specify which DFSMS/MVS® functional components and separately orderable features are licensed for use on a particular MVS system or across a sysplex.
IGDSMSxx	Initialize the Storage Management Subsystem (SMS) and specify the names of the active control data set (ACDS) and the communications data set (COMMDS).
IKJPRMxx	TIOC parameters that control TSO/TCAM time sharing buffers.
IKJTSOxx	For TSO/E specifies authorized commands and authorized program, programs that are authorized when called through the TSO service facility, commands that may not be issued in the background, and defaults for SEND and LISTBC processing.
IPCSPRxx	Parameters that are used during an IPCS session.
IVTPRM00	Sets Communications Storage Manager ICSM) parameters.
LNKLSTxx	List of data sets to be concatenated to form the LNKLST concatenation.
LOADxx	Specifies data sets MVS uses to configure the system.
LPALSTxx	List of data sets to be concatenated to SYS1.LPALIB from which the system builds the pageable LPA (PLPA).
MMSLSTxx	Specifies information that the MVS message service (MMS) uses to control the languages that are available in the installation.
MPFLSTxx	Parameters that the message processing facility uses to control message processing and display.
MSTJCLxx	Contains the master scheduler job control language (JCL) that controls system initialization and processing.
NUCLSTxx	Specifies members of SYS1.NUCLEUS to be included in, or excluded from,

Name	Purpose
	the nucleus region at IPL-time.
PFKTABxx	Parameters contain the definitions for program function key tables (PFK tables).
PROGxx	Contains statements that define the format and contents of the APF-authorized program library list, control the use of installation exits and exit routines, define the LNKLST concatenation, and specify alternate data sets for SYS1.LINKLIB, SYS1.MIGLIB, and SYS1.CSSLIB to appear at the beginning of the LNKLST concatenation and SYS1.LPALIB to appear at the beginning of the LPALST concatenation.
SCHEDxx	Provides centralized control over the size of the master trace table, the completion codes to be eligible for automatic restart and programs to be included in the PPT.
SMFPRMxx	Parameters that define SMF options.
TSOKEYxx	VTIOC parameters that are used by TSO/VTAM time sharing.
VATLSTxx	Volume attribute list that defines the “mount” and “use” attributes of direct access volumes.
XCFPOLxx	Specifies the actions that a system in a sysplex on PR/SM is to take when another system in the sysplex becomes inactive.

Table 13 - SYS1.PARMLIB Overview

8.3 z/OS Main Subsystems

8.3.1 Base Control Program (BCP)

BCP is the core subsystem of z/OS responsible for (real and virtual) storage management, management of address spaces, tasks and SRBs, scheduling, handling of interrupts and exceptions, synchronization etc. The main components of BCP are:

- System Resources Manager (SRM) responsible for managing the assignment of system resources like CPU time, real memory, and I/O capabilities to address spaces

- Real Storage Management (RSM) responsible for managing the use of real storage frames
- Virtual Storage Management (VSM) responsible for managing virtual storage
- Auxiliary Storage Management (ASM) responsible for transferring virtual storage pages between real and auxiliary storage
- Program Management (or Content Supervision) responsible for searching programs in program libraries and loading programs into virtual memory.
- Virtual Lookaside Facility (VLF) responsible for managing data spaces that can be used to cache data
- Workload Manager (WLM) responsible for optimizing the use of resources in order to achieve administrator defined performance goals
- Serialization and Locking responsible to manage the synchronization of the use of serially reusable resources
- Hardware Management Support (BCPii) responsible for managing hardware resources via communication with the HMC/SE.

The main functions provided by BCP to unauthorized programs are described in [ASSEM] with the interface descriptions in [ASSEM.REF-1] and [ASSEM.REF-2]. Note that not all interfaces described there are actually TSFI. Some interfaces are just library functions which may call some TSF interfaces.

Functions available to TSF-internals functions are described in [AUTHASSEM] with the definition of the interfaces in {AUTHASSEM.REF-1] to [AUTHASSEM.REF-4]. Data structures used within the TSF are described in [MVS.DA-1] to MVS.DA-6]. System Messages that may be generated by BCP (and some other components of z/OS) are described in [MVS.MSG-1] to [MVS.MSG-10]. System Commands are described in [MVS.CMDS].

8.3.2 System Management Facilities (SMF)

System management facilities (SMF) collects and records system and job-related information that the installation can use for:

- Billing users.
- Reporting reliability.
- Analyzing the configuration.
- Scheduling jobs.
- Summarizing direct access volume activity.
- Evaluating data set activity.
- Profiling system resource use.
- Maintaining system security.

SMF produces and knows about several record types. More information can be found in [SMF], chapter 1.

Audit records can either be collected in SMF data sets or log streams or both. If SMF data sets are used, SMF will write records to the SMF data sets the system administrator has allocated.

If SMF logging is used, SMF will write records to the system logger managed log streams set up by the system administrator. SMF can write much larger chunks of data to the log stream than it can to SMF data sets. This has the potential to make writing SMF records faster, which could mean collecting more data. SMF logging also allows to group different SMF record types into different log streams defined for different purposes. It is also possible to write a single record to multiple log streams.

An overview of SMF is presented in the following figure, when SMF data sets are used. The description of the process is as follows:

1. Routines (SMF, system, program-product, installation-written) collect and format data into records and then pass the records to the SMF writer.
2. In addition to collecting data for SMF, some routines interface with the SMF exits, passing control to them at several points during job (and job step) processing.
3. The SMF exits get control when specific events occur, such as when a data set exceeds the output limit or at designated points during job processing.
4. SMF routines copy records to SMF buffers, transfer records from the SMF buffers to the SMF data sets, and issue messages to the operator indicating the successful or unsuccessful completion of specific SMF-related events.
5. The SMF data sets are filled one at a time; while SMF writes records on one data set, other SMF data sets can be dumped or cleared.
6. The SMF dump program copies data from SMF system data sets to SMF dump data sets for permanent storage.
7. Analysis and report routines, either user-written or those such as the Tivoli Performance Reporter for OS/390 program product, process information records. Analysis routines read the SMF data set, list the dumped SMF data set, use a sort/merge program to order the SMF-recorded information, or perform a detailed investigation of one particular SMF data item, such as job CPU time under TCBs. Report routines usually format and print the statistics and/or results of the analysis routines.

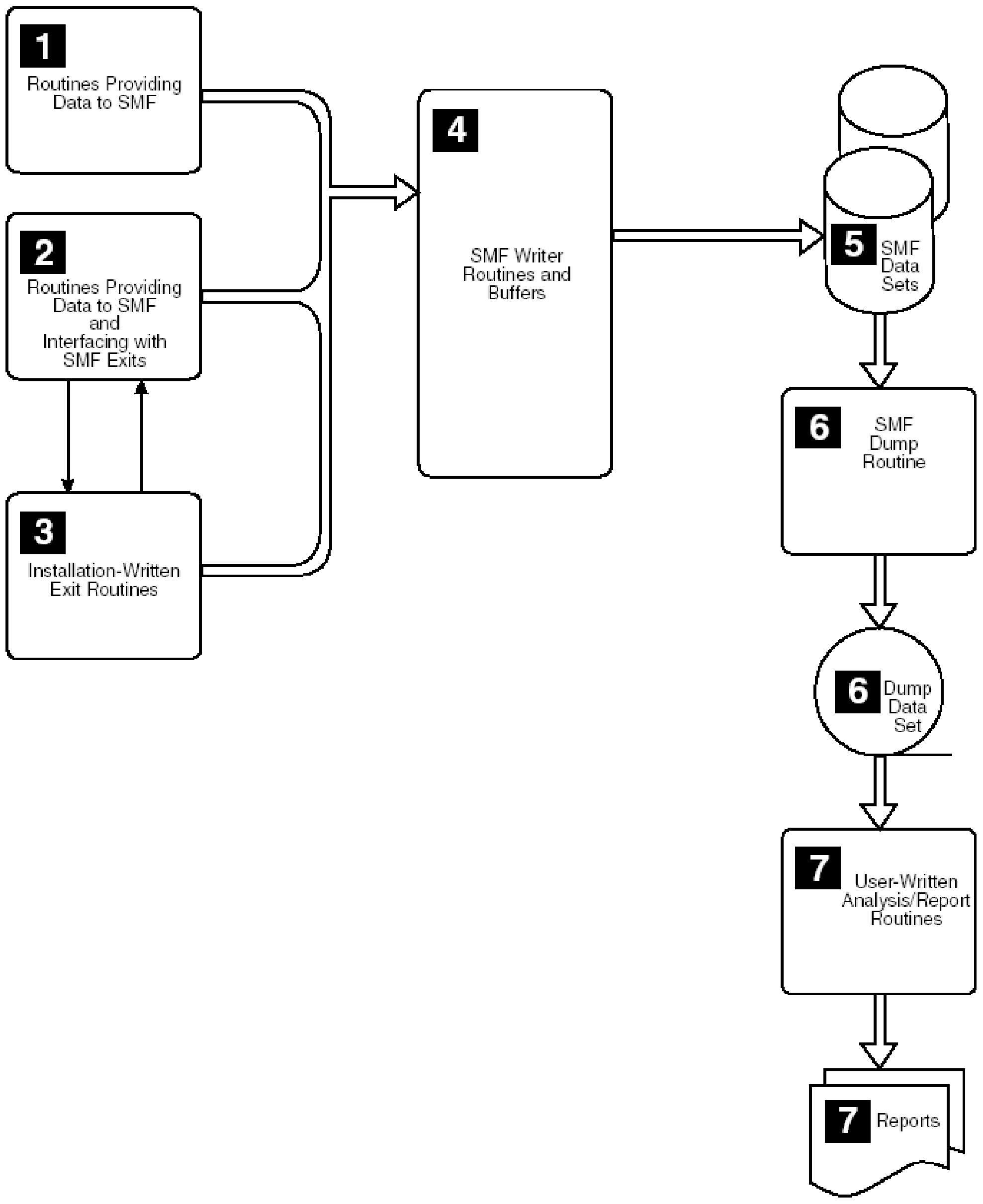

1
Routines Providing Data to SMF
2
Routines Providing Data to SMF and Interfacing with SMF Exits
3
Installation-Written Exit Routines
4
SMF Writer Routines and Buffers
5
SMF Data Sets
6
SMF Dump Routine
6
Dump Data Set
7
User-Written Analysis/Report Routines
7
Reports

Figure 10 : SMF Overview when using SMF data sets

When SMF is recording to log streams, subsystems or user programs still invoke the SMFEWTM macro to write their SMF record. SMFEWTM, to take the user record and invoke SMF code. As shown in Picture 17.4, instead of locating a buffer in SMF private storage, SMF locates a dataspace corresponding to the user's record type and log stream where the record will be written. A buffer with space to hold the record is located and the record is copied there. When the record is full, a task is posted.   Unlike the scheduled approach in SMF data set recording, this task is already started and ready to write. In addition, writes to system logger are done at near memory-to-memory speeds, resulting in an economy of scale not currently possible with SMF data set recording.

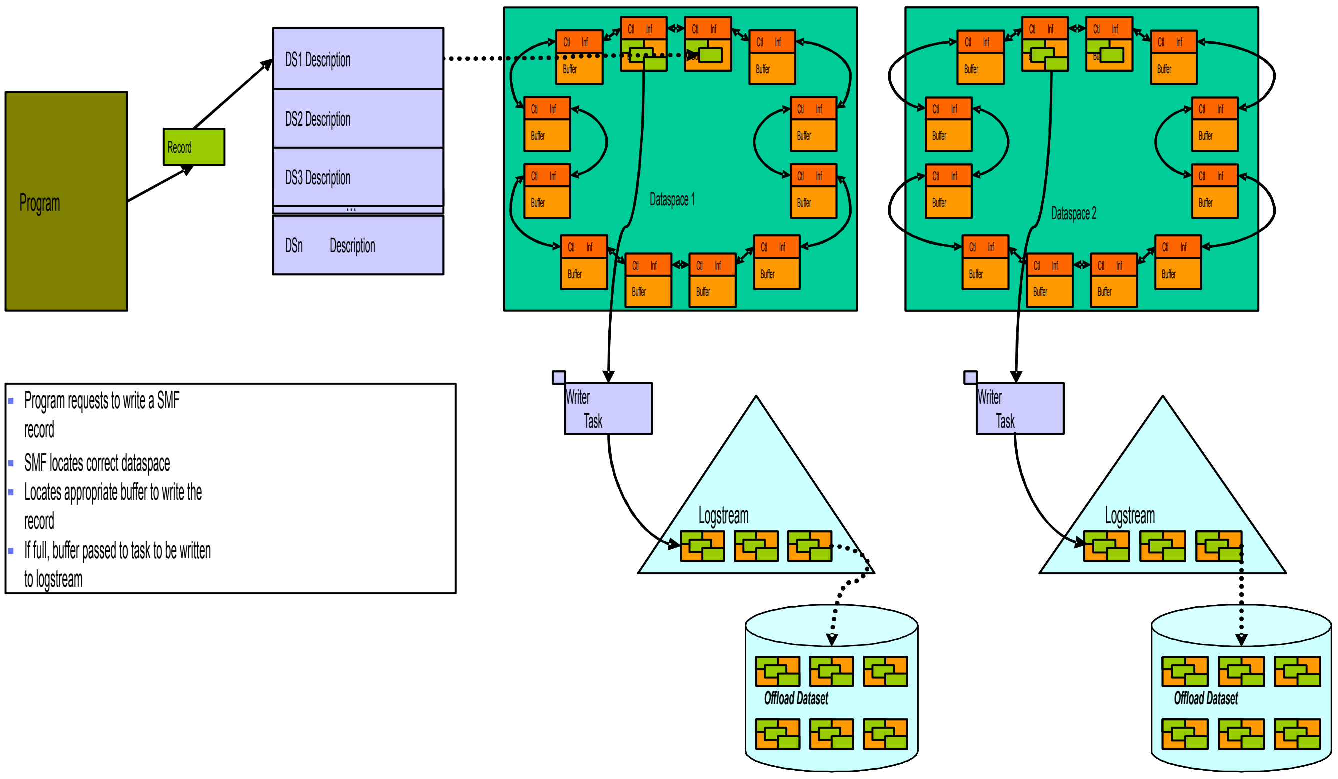

Program
Record
DS1 Description
DS2 Description
DS3 Description
DSn Description
Dataspace 1
Dataspace 2
Ctl Inf
Buffer
Writer Task
Logstream
Offload Dataset
Program requests to write a SMF record
SMF locates correct dataspace
Locates appropriate buffer to write the record
If full, buffer passed to task to be written to logstream

Figure 11 : SMF Overview when using log streams

SMF configuration and operation and the structure of BCP-related SMF records is described in [SMF]. Information on System Logger can be found in [System_Logger].

8.3.3 Data Facility Storage Management Subsystem (DFSMS)

System-managed storage is the IBM automated approach to managing storage resources. It uses software pro-grams to manage data security, placement, migration, backup, recall, recovery, and deletion so that current data is available when needed, space is made available for creating new data and for extending current data, and obsolete data is removed from storage.

The system administrator can tailor system-managed storage to his needs. The requirements for performance, security, and availability are defined, along with storage management policies used to automatically manage the direct access, tape, and optical devices used by the operating systems.

DFSMS functional components and related program products automate and centralize storage management, based on policies the installation defines for availability, performance, space, and security. DFSMS consists of the following functional components:

- DFSMSdfp
- DFSMSdss
- DFSMShsm

- DFSMSrmm
- DFSORT

The DFSMSdfp functional component of DFSMS provides the storage, program, data, and device management functions of z/OS. The Storage Management Subsystem (SMS) component of DFSMSdfp is fundamental to providing these functions. DFSMSdfp provides the foundation for distributed data access, using the Distributed File-Manager to support remote access of z/OS data and storage resources from workstations or personal computers.

The DFSMSdss functional component of DFSMS copies and moves data for z/OS.

The DFSMShsm functional component of DFSMS provides automation for backup, recovery, migration, recall, disaster recovery, and space management functions in the DFSMS environment.

The DFSMSrmm functional component of DFSMS provides the management functions for removable media, including tape cartridges and reels.

DFSORT sorts, merges, and copies data sets. It also helps to analyze data and produce detailed reports using the ICETOOL utility or the OUTFIL function.

8.3.3.1 Data Sets

A data set is a collection of logically related data and can be a source program, a library of macros, or a file of data records used by a processing program. Data records are the basic unit of information used by a processing pro-gram. By placing data into volumes of organized data sets, the data can be saved and processed.

Exception: z/OS UNIX files are different from the typical data set because they are byte oriented rather than record oriented.

The following section will provide a brief overview of data set concepts. For detailed information consult [DFSMS.DS].

Data Storage and Management

Users can store data on secondary storage devices, such as a direct access storage device (DASD) or magnetic tape volume. The term DASD applies to disks or to a mass storage medium on which a computer stores data. A volume is a standard unit of secondary storage. Users can store all types of data sets on DASD but only sequential data sets on magnetic tape. Mountable tape volumes can reside in an automated tape library. Users can also direct a sequential data set to or from spool, a UNIX file, a TSO/E terminal, a unit record device, virtual I/O (VIO), or a dummy data set.

Each block of data on a DASD volume has a distinct location and a unique address, making it possible to find any record without extensive searching. Users can store and retrieve records either directly or sequentially. In addition DASD volumes can be used for storing data and executable programs, including the operating system itself, and for temporary working storage. Users can use one DASD volume for many different data sets, and reallocate or reuse space on the volume.

Data management is the part of the operating system that organizes, identifies, stores, catalogs, and retrieves all the information (including programs) that the installation uses.

Access Methods

An access method defines the technique that is used to store and retrieve data. Access methods have their own data set structures to organize data, macros to define and process data sets, and utility programs to process data sets. Access methods are identified primarily by the data set organization. For example, the programmer can use the basic sequential access method (BSAM) or queued sequential access method (QSAM) with sequential data sets. However, there are times when an access method identified with one organization can be used to process a data set organized in a different manner. For example, a sequential data set (not extended-format data set) created using BSAM can be processed by the basic direct access method (BDAM), and vice versa. Another example is UNIX files, which can be processed using BSAM, QSAM, basic partitioned access method (BPAM).

Basic Direct Access Method

BDAM arranges records in any sequence the program indicates, and retrieves records by actual or relative ad-dress. If the exact location of a record is not known, the programmer can specify a point in the data set where a search for the record is to begin. Data sets organized this way are called direct data sets.

Optionally, BDAM uses hardware keys. Hardware keys are less efficient than the optional software keys in virtual sequential access method (VSAM).

Basic Partitioned Access Method

Basic partitioned access method (BPAM) arranges records as members of a partitioned data set (PDS) or a partitioned data set extended (PDSE) on DASD. BPAM can be used to view a UNIX directory and its files as if it were a PDS. Each PDS, PDSE, or UNIX member can be viewed sequentially with BSAM or QSAM. A PDS or PDSE includes a directory that relates member names to locations within the data set. Use the PDS, PDSE, or UNIX directory to retrieve individual members. For program libraries (load modules and program objects), the directory contains program attributes that are required to load and rebind the member. Although UNIX files can contain program objects, program management does not access UNIX files through BPAM.

Basic Sequential Access Method

BSAM arranges records sequentially in the order in which they are entered. A data set that has this organization is a sequential data set. The user organizes records with other records into blocks. This is basic access.

Object Access Method

Object Access Method (OAM) processes named byte streams (objects) that have no record boundary or other internal orientation. These objects can be recorded in a DB2 database or on an optical or tape storage volume.

Queued Sequential Access Method

QSAM arranges records sequentially in the order that they are entered to form sequential data sets, which are the same as those data sets that BSAM creates. The system organizes records with other records. QSAM anticipates the need for records based on their order. To improve performance, QSAM reads these records into storage be-fore they are requested. This is called queued access.

Virtual Storage Access Method

VSAM arranges records by an index key, relative record number, or relative byte addressing. VSAM is used for direct or sequential processing of fixed-length and variable-length records on DASD. Data that is organized by VSAM is cataloged for easy retrieval and is stored in one of five types of data sets.

- Entry-sequenced data set (ESDS). Contains records in the order in which they were entered. Records are added to the end of the data set.
- Key-sequenced data set (KSDS). Contains records in ascending collating sequence. Records can be accessed by a field, called a key, or by a relative byte address.
- Linear data set (LDS). Contains data that has no record boundaries. Linear data sets contain none of the control information that other VSAM data sets do.
- Relative record data set (RRDS). Contains records in relative record number order, and the records can be accessed only by this number. There are two types of relative record data sets.
    - Fixed-length RRDS: The records must be of fixed length.
    - Variable-length RRDS: The records can vary in length.
- z/OS UNIX files. Although UNIX files are not actually stored as entry-sequenced data sets, the system at-tempts to simulate the characteristics of such a data set. To identify or access a UNIX file, specify the path that leads to it.

Access to z/OS UNIX Files

Programs can access the information in UNIX files through z/OS UNIX System Services (z/OS UNIX) calls, such as open(pathname), read(file descriptor), and write(file descriptor). Programs can also access the information in UNIX files through the BSAM, BPAM, and QSAM access methods. When using BSAM or QSAM, a UNIX file is simulated as a single-volume sequential data set. When using BPAM, a UNIX directory and its files are simulated as a partitioned data set directory and its members.

More general information on DFSMS can be found in [DFSMS.INTRO], [DFSMS.DS], and DFSMS.CAT]. Information on the DFSMS interfaces to untrusted programs can be found in [DFSMS.MAC], [DFSMSDFP.AS] and [DFSMS.AMC]

8.3.4 Resource Access Control Facility (RACF)

8.3.4.1 Introduction

RACF is the central component within z/OS responsible for the identification and authentication of users, for access control, and for the generation of security event related audit records (which RACF sends to SMF to get those audit records included in the SMF audit).

RACF operates as an identification and authentication and access control server which is invoked by components of z/OS when they need those services to be performed. For this purpose RACF exports interfaces that those components can invoke. Those interfaces are the RACROUTE services (invoked using the RACROUTE macro) and the RACF callable services (invoked as function calls). With a few exceptions those interfaces are restricted to callers that operate in a privileged mode of z/OS.

The RACF commands are used for the management of RACF and RACF data. The use of most RACF commands and specific parameters of RACF commands is restricted to users with specific privileges. The privileges required are defined in the TOE summary specification as well as in [RACF.CMD].

8.3.4.2 User Identification and Authentication - General Aspects

RACF is the central facility used within z/OS for the identification and authentication of users. RACF manages users within the RACF database using user profiles. User profiles contain the security attributes of a user and are managed using the RACF ADDUSER and ALTUSER commands. In addition the following commands can be used to manage some user related security attributes:

- PASSWD to manage a user's password
- PHRASE to manage a user's passphrase
- RACDCERT for the management of digital certificates associated with a user
- RACMAP for managing user identity mappings

z/OS and trusted servers executing as part of the z/OS TSF use the following functions of RACF to perform user authentication:

- RACROUTE REQUEST=VERIFY
- RACROUTE REQUEST=VERIFYX
- initACEE

Note that components of the z/OS TSF like UNIX System Services provide their own interfaces for user authentication to trusted programs, which then internally call one of three RACF interfaces listed above.

Authentication in the TOE can have different meanings, depending on the context. For example, it can refer to:

- A user providing a user ID and an authenticator (password, password phrase, PassTicket).
- Authentication via a client digital certificate, where:
  - A user (or a client application running on the user’s behalf) presenting a digital certificate to a server, via TLS V1.1 or TLS V1.2 protocols, where TLS processing simply validates the contents of the certificate and ensures that the user/application possesses the proper private key.
  - Or, following on from the prior case, the server then asks RACF to map the client’s digital certificate to a RACF user ID so the server can perform work on the client’s behalf using that RACF identity, rather than using the server’s own identity or a surrogate identity configured in the server by the administrator.
- Authentication via Kerberos where:

 - ◦ A user authenticates to a Kerberos user registry (KDC), which may be implemented on z/OS via Network Authentication Services, and receives a Ticket Granting Ticket (TGT) allowing the user to request service tickets to authenticate subsequently to servers.. The client (user) may be on z/OS or elsewhere.
 - ◦ Or, once the user (client application) has requested a service ticket for a specific application server on z/OS, the user may present that ticket to the application server, and the server will validate that ticket using Kerberos and GSS-API functions.
 - ◦ Or, following on from that level of validation by the application server, the application server may map the user's Kerberos service ticket to a RACF user ID so the server can perform work on the client's behalf using that RACF identity, rather than using the server's own identity or a surrogate identity configured in the server by the administrator.

When RACF has successfully identified and authenticated a user, it creates a control block called Accessor Control Environment Element (ACEE), which contains the user's security attributes.

8.3.4.3 Access control - General Aspects

RACF is also the central facility in z/OS responsible for controlling access to resources. RACF manages data set and general resource profiles in its database. Those profiles are structured into classes where each class represents a "type" of resource and where the individual profiles within a class represent a resource. To make management of resource profiles more efficient, several resources within a resource class that require the same access protection can be protected by a generic profile that covers such a defined set of resources.

Together with the profiles RACF manages an access control list for each profile that defines the types of access users or groups may have to a resource protected by the profile.

A resource manager will then use the RACROUTE REQUEST=AUTH and RACROUTE REQUEST=FASTAUTH functions to request RACF to check if a user or group has the requested type of access to a RACF protected resource.

RACF by itself is basically a database that has a query interface. The caller queries RACF if a specific subject is allowed to perform an action upon an object. RACF, by examining its rules provides an answer.

While the possible operations upon objects are defined by RACF the object is not. An object is simply a name that consist of the name of a class and the name of a resource in that class. Not taking into account classes and re-sources that are relevant to RACF itself, these names are to RACF exactly that: names, nothing else. That means that each resource manager can define its own names and link certain semantics to that name. RACF only provides answers as described above. What the calling resource manager does with that answer is of no further concern to RACF.

When queried about non-existing classes or non-existing resources, RACF provides answers according to the following general rules:

- If a (non-existing) resource in a non-existing class is queried, RACF provides the answer the class and re-source does not exist

- If a non-existing resource in an existing class is queried, RACF applies the default access rule for that class to the resource and returns an answer that indicates the authorization of the accessing subject. By doing so, it is possible to protect complete classes of resources by one default access rule (if the resource name is defined to RACF or not does not matter).

UNIX file System and UNIX IPC objects are not protected by RACF profiles but the security attributes that are used in the algorithm that determines access are stored with the object. The z/OS UNIX System Services will extract those security attributes from the object and pass them to RACF via the appropriate RACF callable services. RACF then performs the access check also for those objects.

8.3.4.4 RACF Commands

The main interface used for the management of RACF user, group, and resource profiles as well as RACF configuration data and certificates used by RACF for user authentication is the RACF command interface. The use of RACF commands and command parameters is restricted specifically for each command and sometimes even for specific parameters of a RACF command. The authorizations required for the use of each command and the individual command parameters are defined later in this chapter.

Guidance on the configuration and management of RACF functionality can be found in [RACF.SAG]. Guidance on the configuration and management of the audit functionality of RACF can be found in [RACF.AUDIT].

The description of the format of the audit records generated by RACF can be found in chapters 5 to 8 of [RACF.MAC]. The appendices of this document also describe the format of the different profiles stored in the RACF database.

The description of the RACF commands can be found in [RACF.CMD].

The description of the RACF callable services can be found in [RACF.CS]. The description of the RACROUTE interfaces to RACF can be found in [RACF.RACROUTE].

Messages generated by RACF are described in [RACF.MSG].

8.3.5 Integrated Cryptographic Control Facility (ICSF)

ICSF is the main provider of basic cryptographic services within z/OS and for the functions specified in the SFRs is used for the basic cryptographic services for certificate/key generation for certificates used for user authentication as well as certificates used in the establishment of trusted channels. Also the basic cryptographic functions used for the IPSec and TLS protocols are provided by ICSF, unless the basic functions are already provided by CPACF. ICSF is also the only interface to a cryptographic coprocessor if installed on a z/OS system.

Although ICSF exports interfaces for cryptographic functions that can be used by unauthorized user programs (described in [ICSF.APG]), those functions are not related to SFRs defined in this Security Target except that those functions are used within the TSF by the Communications Server component to provide the cryptographic services for setting up a trusted channel. To make those services available for those TSF components, ICSF has to be configured as described in the configuration guidance for IPSec and AT-TLS. The configuration and management of ICSF itself is described in [ICSF.SAG].

8.3.6 Communications Server

The Communications Server component of z/OS is responsible for the implementation of the TCP/IP stack and (except for SSH) the higher level protocols. As security functionality the Communications Server provides:

- access control on the objects:
  - TCP/IP stack,
  - ports,
  - networks and hosts (IP addresses)
- Trusted channels:
  - based on the TLS protocol
  - based on IPSec
- IP filtering capabilities

For access control the Communications Server uses RACF to verify a user's access rights which are defined by access control lists for dedicated profiles that represent the controlled resources mentioned above. The security claims later in this chapter define the names of those profiles.

The Communications Server can establish a trusted channel using either the TLS or the IPSec protocol. IPSec supports the following features:

- Authentication with pre-shared keys (supported in the evaluated configuration)
- Authentication with RSA and ECDSA digital signatures (ECDSA only supported for IKEv2)
- Use of IKEv1 main mode (identity protect mode) and aggressive mode
- Support for Diffie Hellman groups 1, 2, 5, 14, 19, 20, 21 and 24 (supported in the evaluated configuration)
- ESP support for AES_CBC and triple DES encryption with HMAC_SHA1, HMAC_SHA2 and AES128_XCBC_96 authentication (supported in the evaluated configuration)
- ESP support for AES_GCM combined mode encryption and authentication (supported in the evaluated configuration)
- ESP support for null encryption with AES_GMAC authentication (supported in the evaluated configuration
- AH support for HMAC_SHA1, HMAC_SHA2 and AES128_XCBC_96 authentication (supported in the evaluated configuration)

The TLS protocol is supported via the "Application-Transparent Transport Layer Security (AT-TLS) feature. The AT-TLS feature of the z/OS TCP/IP stack uses z/OS System SSL to implement the TLS protocols inside the TCP layer of the stack. Based on configured policy, these protocols are performed underneath the sockets API on behalf of applications using TCP/IP. Most applications do not need to know when they are using a secure transport and require no change to take advantage of this support. An ioctl service (SIOCTTLSCTL) is provided for those applications that wish to exploit AT-TLS but need to be aware of or control its use.

Application Transparent TLS (AT-TLS) consolidates TLS implementation in one location, reducing or eliminating application development overhead, maintenance, and parameter specification. AT-TLS is based on z/OS System SSL, and transparently implements these protocols in the TCP layer of the stack. As shown in below, many existing applications are able to take advantage of this support with no change to the application. They can be completely unaware of AT-TLS support. Other applications need to negotiate the use of TLS through their application-specific protocol prior to initiating the TLS handshake, or the application might need to extract and examine the partner’s client certificate after handshake completion. These applications need to be aware of AT-TLS and re-quire small changes using ioctl support to control AT-TLS.

In all cases, an application using a socket enabled with AT-TLS continues to send and receive text data in the clear while encrypted data flows over the network. This allows the use of TLS with applications that cannot be modified or that cannot incorporate one of the available tool kits. The partner application must also support TLS protocols, either by using AT-TLS or an available TLS toolkit.

The guidance on the configuration and management of Communications Server functions is described in [CSIP.SAG] with the detailed reference to the configuration data syntax and semantics in [CSIP.CFG]. Commands related to Communications Server functionality are described in [CSIP.CMDS] and [CSIP.ADMIN]. Programming interfaces for untrusted programs are described in in [CSIP.API].

8.3.7 Directory Services

The z/OS Lightweight Directory Access Protocol (LDAP) server, part of IBM Tivoli Directory Server for z/OS (IBM), is based on a client/server model that provides client access to an LDAP server. An LDAP directory provides an easy way to maintain directory information in a central location for storage, update, retrieval, and exchange.

The LDAP server provides the following functions:

- Interoperability with any Version 2 or Version 3 LDAP directory client
- Access controls on directory information, using static, dynamic, and nested groups
- Transport Layer Security (TLS) communication (TLS V1.1 and TLS V1.2)
- Start TLS (Transport Layer Security) activation of secure communication
- Client and server authentication using TLS
- Password encryption or hashing
- Password policy
- Basic Replication
- Advanced Replication

- Referrals
- Aliases
- Change logging
- LDAP Version 2 and Version 3 protocol support
- Schema publication and update

- Native authentication
- CRAM-MD5 (Challenge-Response Authentication Method) and DIGEST-MD5 authentication
- Root DSE information
- LDAP access to information stored in RACF®
- Support for sharing directory data in a sysplex
- Plugin support

Configuration and administration of the Directory Server are described in [LDAP.SA].

8.3.8 Public Key Infrastructure

The z/OS Cryptographic Services PKI Services allows to use z/OS to establish a PKI infrastructure and serve as a certificate authority for your internal and external users, issuing and administering digital certificates in accordance with your own organization's policies. End users utilize the PKI Services Web application to request and obtain certificates through their Web browsers, while authorized PKI administrators approve, modify, or reject these requests through their own Web browsers. The Web applications provided with PKI Services are highly customizable, and a programming exit is also included for advanced customization. You can allow automatic approval for certificate requests from certain users and, to provide additional authentication, add host IDs, such as RACF user IDs, to certificates you issue for certain users. You can also issue your own certificates for browsers, servers, and other purposes, such as virtual private network (VPN) devices, smart cards, and secure e-mail. PKI Services sup-ports Public Key Infrastructure for X.509 version 3 (PKIX) and Common Data Security Architecture (CDSA) cryptographic standards. It also supports the following:

- The delivery of certificates through the TLS protocol for use with applications that are accessed from a Web browser or Web server.
- The delivery of certificates that support the Internet Protocol Security standard (IPSEC) for use with secure VPN applications or IPSEC-enabled devices.
- The delivery of certificates that support Secure Multipurpose Internet Mail Extensions (S/MIME), for use with secure e-mail applications.

All PKI Services requests go through the RACF R_PKIServ callable service, and the authorization mechanism for this callable service varies depending on the type of function requested (end user versus administrative). For the end user functions, this interface is protected by FACILITY class profiles (resources) of the form IRR.RPKISERV.*(function)[.ca-domain]*, where (function) is one of the end user function names described under Function_code below. If the CA_domain parameter supplied on the R_PKIServ call is not null (has a length greater than 0), then the profile is qualified with the CA domain name. If the CA_domain parameter supplied on the R_PKIServ call is null, then the qualifier is not used. For example, if the function name is GENCERT and the CA_domain parameter is “Customers”, then the FACILITY class resource is IRR.RPKISERV.GENCERT.CUSTOMER. However, if the CA_domain parameter is null, then the FACILITY class resource is IRR.RPKISERV.GENCERT. The user ID (from the ACEE associated with the address space) for the application, is used to determine access.

For the administrative functions, this interface is protected by a single FACILITY class profile (resource), IRR.RPKISERV.PKIADMIN[*.ca-domain]*. If the CA_domain parameter supplied on the R_PKIServ call is not null (has a length greater than 0), then the profile is qualified with the CA domain name. If the CA_domain parameter supplied on the R_PKIServ call is null, then the qualifier is not used. For example, if the CA_domain parameter is “Customers”, then the FACILITY class resource is IRR.RPKISERV.PKIADMIN.CUSTOMER. However, if the CA_domain parameter is null, then the FACILITY class resource is IRR.RPKISERV.PKIADMIN.

If the caller is not RACF SPECIAL, then the caller will need READ access to perform read operations (QUERYREQS, QUERYCERTS, REQDETAILS, and CERTDETAILS) and UPDATE access for the action operations, (PREREGISTER, MODIFYREQS and MODIFYCERTS). To determine the caller, the current TCB is checked for an ACEE. If one is found, the authority of that user is checked. If there is no ACEE associated with the current TCB, the ACEE associated with the address space is used to locate the user ID.

Authorization to key generation for the certificate is controlled by the CRYPTOZ class and the CSFSERV class (if active). The keys are stored in the Token Dataset (TKDS).

The PKI daemon needs to have UPDATE access to the SO.<token name> profile and CONTROL access to the USER.<token name> profile in the CRYPTOZ class.

If the CSFSERV class is active, the daemon needs to have READ access to the following profiles: CSF1TRC, CSF1TRD, CSF1TRL, CSF1SAV, CSF1GAV, CSF1GKP, CSFOWH.

The configuration, administration and use of the z/OS PKI is described in [PKI.SGR].

8.3.9 Job Entry Subsystem 2 (JES2)

8.3.9.1 Overview

z/OS uses a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by z/OS, and to control their output processing. JES2 is descended from HASP (Houston automatic spooling priority). HASP is defined as: a computer program that provides supplementary job management, data management, and task management functions such as: scheduling, control of job flow, and spooling. HASP remains within JES2 as the prefix of most module names and the prefix of all messages sent by JES2 to the operator.

JES2 (job entry subsystem 2) is a functional extension of the HASP II program that receives jobs into the system and processes all output data produced by the job. JES2 is that component of z/OS that provides the necessary functions to get jobs into, and output out of, the z/OS system. It is designed to provide efficient spooling, scheduling, and management facilities for the z/OS operating system.

By separating job processing into a number of tasks, z/OS operates more efficiently. At any point in time, the computer system resources are busy processing the tasks for individual jobs, while other tasks are waiting for those resources to become available. In its most simple view, z/OS divides the management of jobs and resources be-tween the JES and the base control program of z/OS. In this manner, JES2 manages jobs before and after running the program; the base control program manages them during processing.

To manage job input/output responsibilities for z/OS, JES2 controls a number of functional areas, all of which can be customized to the installation's need. Some of the major functional areas and processing capabilities are:

- Getting work into and out of z/OS (input/output control)
- Maximizing efficiency through job selection and scheduling
- Offloading work and backing up system workload
- Supporting advanced function printers (AFPs)
- Running multiple copies of JES2
- System security.

JES2 has the flexibility to fulfill an installation's unique processing needs. An installation can virtually control every JES2 function. Many customization tasks can be performed when JES2 is installed, and can be dynamically customized whenever the processing needs change. JES2 provides initialization statements, commands, IBM-supplied exit points, the ability to add installation-defined exit points that require minimal source code modification, and the ability to change many commands and system messages, all without the need to modify the IBM-supplied code.

No large data processing system or subsystem can continually operate independently of system programmer or operator intervention. A two-way communication mechanism between the operator and JES2 is required. Based on system workload and device requirements, JES2 must communicate its status to the operator and/or system programmer. JES2 issues messages to communicate job and device status, problem situations, system resource constraint situations, and performance status. Through commands, current status can be requested and through the use of various tools further information can be obtained to diagnose and correct problem and system failure situations.

8.3.9.2 JCL

The Job Control Language, is a way to describe or to tell a z/OS system what to do. A defined collection of JCL statements is called a job. For every job submitted, the submitter needs to tell z/OS where to find the appropriate input, how to process that input (that is, what program or programs to run), and what to do with the resulting out-put.

JCL is used to convey this information to z/OS through a set of statements known as job control statements. JCL's set of job control statements is quite large and enables to provide a great deal of information to z/OS. Most jobs, however, can be run using a very small subset of these control statements.

Within each job, the control statements are grouped into job steps. A job step consists of all the control statements needed to run one program. If a job needs to run more than one program, the job would contain a different job step for each of those programs.

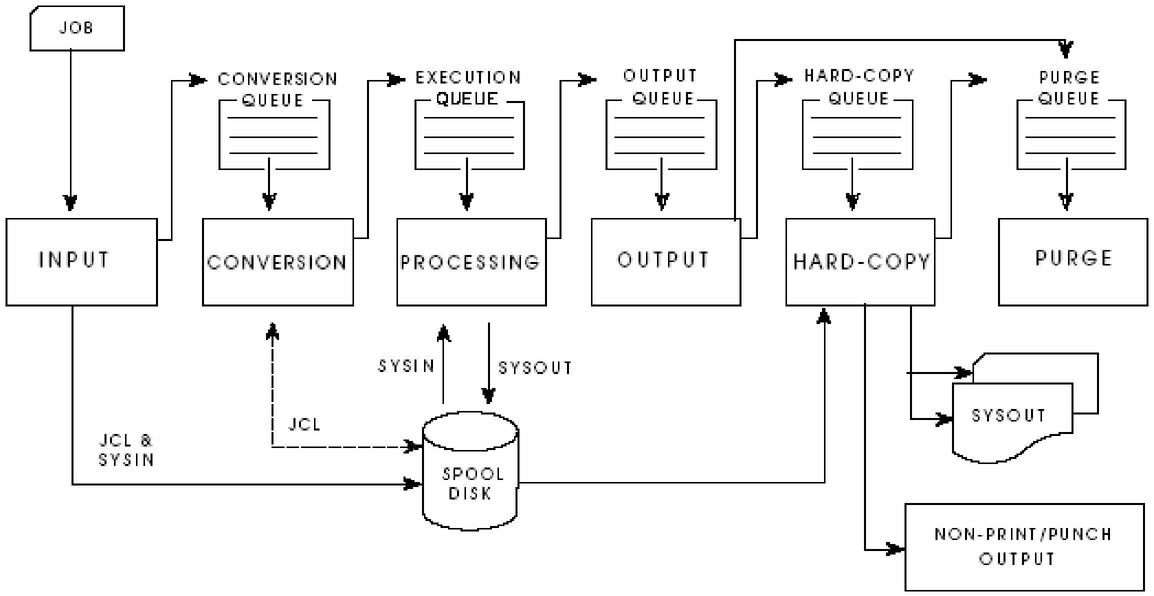
Every job must contain a minimum of the following two types of control statements:

- A JOB statement, to mark the beginning of a job and assign a name to the job. The JOB statement is also used to provide certain administrative information, including security, accounting, and identification information. Every job has one and only one JOB statement.
- An EXEC (execute) statement, to mark the beginning of a job step, to assign a name to the step, and to identify the program or procedure to be executed in the step. Various parameters can be added to the EXEC statement to customize the way the program executes. Every job has at least one EXEC statement.

In addition to the JOB and EXEC statements, most jobs usually also contain:

- One or more DD (data definition) statements, to identify and describe the input and output data to be used in the step. The DD statement may be used to request a previously-created data set, to define a new data set, to define a temporary data set, or to define and specify the characteristics of the output.

8.3.9.3 Phases of JES2 Job Processing



The following picture provides an overview of the JES2 processing phases, the individual phases are further explained in the flow of this section.

Figure 12 : JES2 Processing Phases

Input phase

JES2 accepts jobs (in the form of an input stream) from input devices such as card readers, remote terminals, or other programs. Input streams can also come from other nodes in a job entry network and from internal readers. An internal reader is a program that other programs can use to submit jobs, control statements, and commands to JES2. Any job running in z/OS can use an internal reader to pass an input stream to JES2, and JES2 can receive multiple jobs simultaneously through multiple internal readers.

z/OS uses internal readers, allocated during system initialization, to pass to JES2 the job control language (JCL) for started tasks, START and MOUNT commands, and TSO LOGON requests.

The system programmer defines internal readers used to process all batch jobs other than STCs and TSO re-quests. JES2 initialization statements define these internal readers which JES2 also allocates during its initialization processing. The internal readers for batch jobs can be used for STCs and TSO requests, if not processing jobs.

As JES2 reads the input stream, it assigns a job identifier to each job and places each job’s JCL, optional JES2 control statements, and SYSIN data onto DASD data sets called spool data sets. JES2 then selects jobs from the spool data sets for processing and subsequent running.

Conversion phase

JES2 uses a converter program to analyze each job’s JCL statements. The converter takes the job’s JCL, merges it with JCL from a procedure library (such as SYS1.PROCLIB), and converts the composite JCL into converter/interpreter text that both JES2 and the job scheduler functions of z/OS can recognize. JES2 then stores the converter/interpreter text on the spool data set. If JES2 detects any JCL errors, JES2 issues messages, and the job is queued for output processing rather than run. If there are no errors, JES2 queues the job for execution. JES2 supports multiple converters; therefore, jobs may not always be processed in a first-in-first-out (FIFO) order. When work load manager (WLM) batch management is in use, JES2 queues the job according to its arrival time.

Processing phase

In the processing phase, JES2 responds to requests for jobs from the z/OS initiators. JES2 selects from a job queue, jobs that are waiting to run and sends them to z/OS.

By recognizing the current processing phase of all jobs on the job queue, JES2 can manage the flow of jobs through the system.

JES2 Job Scheduling: To process the jobs on the job queue, JES2 communicates with an initiator. An initiator is a system program that starts a job to allow it to compete for system resources with other jobs that are already running.

Initiators are controlled by JES2 or by z/OS workload management (WLM).

- JES2 initiators are started by the operator or by JES2 automatically when the system initializes. The initiators select jobs based on the job class(es) that are assigned to the initiator and the priority of the queued jobs. The installation associates each initiator with one or more job classes in a way to encourage the efficient use of available system resources.
- WLM initiators are started by the system automatically based on the performance goals, relative importance of the batch workload, and the capacity of the system to

do more work. The initiators select jobs based on their service class and the order they were made available for execution.

After JES2 selects a job and passes it to the initiator, the initiator invokes the interpreter to build control blocks from the converter/interpreter text that the converter created for the job.

The initiator then allocates the resources specified in the JCL for the first step of the job. This allocation ensures that the devices are available before the job step starts running. The initiator then starts the program requested in the JCL EXEC statement.

Priority Aging: When all initiators are busy, throughput of certain jobs might fall below normal expectations. To help in these situations, JES2 uses the additional scheduling function of priority aging. Priority aging can help ensure that jobs that have been waiting to run have a chance of being selected to run before those jobs that just entered the system. By using priority aging, an installation can increase the priority of a waiting job. The longer the job waits, the higher its priority becomes, up to a limit, and the greater its chances of being selected to run.

JES2-Base Control Program Interaction: JES2 and the base control program communicate constantly to control system processing. The communication mechanism, known as the subsystem interface, allows z/OS to request services of JES2. For example, a requestor can ask JES2 to find a job, do message or command processing, or open (access) a SYSIN or SYSOUT data set. Further, the base control program notifies JES2 of events such as messages, operator commands, the end of a job, or the end of a task.

Output phase

JES2 controls all SYSOUT processing. SYSOUT is system-produced output; that is, all output produced by, or for, a job. This output includes system messages that must be printed, as well as data sets requested by the user that must be printed or punched. After a job finishes, JES2 analyzes the characteristics of the job's output in terms of its output class and device setup requirements; then JES2 groups data sets with similar characteristics. JES2 queues the output for print or punch processing.

Hard-copy phase

JES2 selects output for processing from the output queues by output class, route code, priority, and other criteria. The output queue can have output that is to be processed locally or output to be processed at a remote location (either an RJE workstation or another node). JES2 handles each of these situations in different ways.

- Local Output: When output is to be processed at a local or remotely-attached output device, JES2 uses these local and remotely-attached output devices to produce a job's output. JES2 queues a job's print and punch data sets on the output queue for the local and remote output devices. The active devices, that are attached locally or through RJE connections, select the output data sets with characteristics that best match their selection criteria.
- Network Job Entry Output: Job output passing through to another JES2 node resides on the network out-put queue. JES2 selects a job's output from the network output queue for transmission to another node based upon the priority and the desirability of reaching the output-processing node over the available transmission line. After the

receiving node signals that it has accepted total responsibility for the output, the transmitting JES2 node releases the resources used to represent the output.

After processing all the output for a particular job, JES2 puts the job on the purge queue.

Purge phase

When all processing for a job completes, JES2 releases the spool space assigned to the job, making the space available for allocation to subsequent jobs. JES2 then issues a message to the operator indicating that the job has been purged from the system.

8.3.9.4 Network Job Entry (NJE)

This section provides a very brief overview of NJE, please see [JES2.NJE] and [JES2.SAG] for more information.

An NJE network is a group of two or more complexes or systems that communicate with each other. An NJE net-work is comprised of nodes that can transmit or receive a unit of work. The nodes in an NJE network use protocols to communicate with each other. Protocols are rules a node uses to:

- Become part of an NJE network
- Receive a unit of work
- Send a unit of work
- Indicate it was removed from the network.

NJE Protocols

There are three ways to send and receive data in NJE. One is binary synchronous communication (BSC), the second is Systems Network Architecture (SNA), and the third is Transmission Control Protocol/Internet Protocol (TCP/IP). A network can consist of any combination of SNA, BSC, and TCP/IP connections.

A JES2 complex can use the BSC, SNA or TCP/IP protocol, or all. A user submitting an NJE job is not aware of whether JES2 is using BSC, SNA or TCP/IP.

The difference between logical and physical configurations is the most significant distinction between SNA, BSC, and TCP/IP protocols. In BSC NJE, the physical configuration is the logical configuration. In SNA or TCP/IP NJE, the logical and physical configurations can be quite different. Various routing tables in the host and the communication controllers and routers determine the logical configuration. See [JES2.SAG] for more details.

Major NJE Data Elements

An NJE transfer unit is a unit of work that is transmitted across the network. An NJE transfer unit can be either an NJE job or a nodal message record (NMR).

An NJE job is a transfer unit that contains data to be processed at another node in the NJE network. It begins with a job header, is followed by data, and ends with a job trailer. The type of data contained in the NJE job further defines the type of NJE job. The data between the job header and job trailer can be either SYSIN or SYSOUT data. An NJE SYSIN job is an NJE job that contains JCL for a job and may have one or more SYSIN data sets. An NJE

SYSOUT job is an NJE job that contains one or more SYSOUT data sets. Each SYSOUT data set is preceded by a data set header.

A nodal message record (NMR) is a unit of work that begins with an NMR header and is followed by message text. The message text can be either a message or a system command.

Functions of a Node

A node is a system or complex that is defined to an installation. A node in the network can be another complex or system within a single location or it can be a complex that resides in a remote location. Each node that a complex can access must be identified to other complexes by a unique NJE node name.

Note: If a node uses SNA protocols, the node has two names:

- An LU name (as defined to VTAM 2), and
- An NJE node name created during initialization processing

The NJE node name appears in job headers, data set headers, and NMRs. Each node in the network can do the following with an NJE transfer unit:

Transmit The node packages the NJE transfer unit and transmits it to another node.

Receive The node recognizes the NJE transfer unit, receives, and stores it.

Store-and-forward The node accepts the NJE transfer unit, stores it, and schedules it to be forwarded to another node.

Types of NJE Users

Originating User is the user that submits an NJE transfer unit at the originating node. The originating user submits the NJE transfer unit at an operator console, terminal, or an RJE workstation. An NJE transfer unit may originate from another NJE transfer unit.

Destination User is a user or device (printer or punch) that is the target of an NJE SYSOUT job.

Notification User is the user who receives messages that notify the user of the status of the NJE transfer unit.

Accounting User is the user that receives the notification of the amount or cost of system resources used in processing an NJE transfer unit.

8.3.9.5 Authentication

JES2 issues a request to RACF to validate a job’s userid and password when a job enters the system from any source. This ensures that JES2 does not process any unauthorized work. For jobs submitted locally, to run locally, JES obtains the submittor’s security attributes and passes it to RACF on the RACROUTE RE-QUEST=VERIFYX issued when the job is submitted. For jobs submitted to run remotely, much the same happens. JES knows the identity of the submittor, and passes that to the networked system along with the job. Additionally, if the JOB statement contains a user ID and/or a password, JES will pass them, too. The password (if present) can be passed encrypted (in standard RACF format, via RACROUTE REQUEST=ENCRYPT) if the administrator wishes. At the receiving system, JES will issue a RACROUTE REQUEST=VERIFYX specifying the information received from the originating system as the submittor ID, and also specifying the originating node name. The administrator at the receiving system controls whether RACF will trust the submittor's identity or not. If the administrator chooses to trust the identity (which may be set globally for a complete originating node, or may be set specifically by user or group) then RACF will do standard job submission checking, propagation, etc. If the administrator chooses not to trust the originating system's authentication, RACF will require user ID and password information. Additionally, the administrator can translate originating system information (user ID, group name) to local system conventions. All these functions for NJE processing utilize the RACF NODES class.

Propagation of security information: If RACF is active and a job enters the system with some or all security information missing, SAF propagates the submittor's security information to the job.

Any user in a currently active session (TSO/E, executing batch job or INTRDR) can have SAF propagate the security information associated with the session by omitting the information on the JOB statement for the job. If SAF can not determine the group information from the current session, SAF uses the default information in the RACF profile. However, SAF will not propagate security information to a job that enters the system from an RJE workstation or a physical card reader.

Propagating security information across a network: To allow users to submit jobs without specifying security information such as a userid, JES2 can propagate the submittor's security information.

There are 2 basic ways to get a job to submitted over the network. They are loosely described as the 1 or 2 job card cases.

The one job card case has a /*XEQ or /*ROUTE XEQ card to tell JES2 to send the job into the network. The submitter of the job that is being sent is the identity that is normally associated with a batch job being submitted (TSO user, ID associated with an STC or batch job, etc). The information on the JOB card is sent and processed on the receiving node and not validated on the sending node.

The 2 job card case is created by using the /*XMIT or // XMIT control card. As the name implies, there are 2 job cards in this case. One that is processed on the sending node and one that is processed on the receiving node. The first job card is processed like any batch job and a RACF token is obtained for it. The second job card comes right after the /*XMIT or // XMIT card and is not looked at on the submitting node. It is sent, as is, to the receiving node for processing. When this job is sent. the submitter information that is associated with the job in the NJE data structures is the RACF token obtained from validating the first job card.

When a node receives a job through a network, RACF determines who submitted the job – as just described. After determining the submitting userid, RACF can translate the submitting userid to a valid userid on this system if a profile on the receiving node specifies that the userid must be translated. RACF uses the submitting userid or its translation to supply any missing security information. Security information RACF propagates is:

- Userid
- Group

If JES2 receives a request from an application or subsystem, JES2 propagates any security information provided by the application or subsystem to SAF.

If SAF is unable to verify the security information on the first JOB statement:

- If a TSO/E user submitted the job, SAF passes the TSO/E user's security information to the second JOB statement
- If the job entered the system through a local card reader, SAF uses a default userid for propagation purposes.

8.3.9.6 Authorization

The following sections describe how JES2 authorizes the use of resources it controls. A detailed description can be found in [JES2.SAG].

SAF

During its processing, JES2 will pass information to the security authorization facility (SAF) to perform password validation, to request authority to access a resource, to build or obtain a security token for a resource, or to deter-mine security information in a certain environment. When SAF receives the request, it determines if RACF is active. If RACF is active, SAF passes the request to RACF; if RACF is not active, SAF returns immediately to JES2 with a response indicating that SAF could not validate the request and any existing JES2 security processing controls the resource. When SAF indicates a decision on a security request, JES2 bypasses its own security processing.

RACF

JES2 resources can be secured by creating RACF resource profiles. Each profile (or generic profile) contains:

- The name that identifies a resource or group of resources
- The class of the resource
- The availability of the resource to all users
- A list of userids or groupIDs that can access the resource and their authorization level, if needed
- Other security-related information.

The profile name identifies a resource or set of resources to RACF. The name that identifies a resource to JES2 (defined in the initialization data set) is the basis of the profile name the RACF administrator uses to define the RACF profile. The security administrator defines different types of resources (printers, nodes, and SYSOUT, for example) to different RACF classes. The following table shows the JES2 resource type and the class(es) the RACF administrator can use to define the resource.

JES2 Resource	RACF Profile Name Format	RACF Classes	Class Purpose
Commands from Network Job Entry (NJE) Nodes	NJE.nodename jesname.command[.qualifier] node.RUSER.userid	FACILITY OPERCMDS	Allows a node to issue

JES2 Resource	RACF Profile Name Format	RACF Classes	Class Purpose
		NODES	commands to the system
Commands from RJE Workstations	jesname.command[.qualifier]	OPERCMDS FACILITY	Restrict commands to authorized users
Data sets JES2 uses: o Initialization dataset o Spool data sets o Checkpoint data sets o Procedure libraries o Module libraries o Parameter libraries o Spool offload data sets	’data set name’	DATASET (always active)	Prevents unauthorized access to data sets
Data Sets Residing on Spool o SYSIN o SYSOUT o JESNEWS o Trace data sets o SYSLOG	localnodeid.userid.jobname.jobid.dsnumber.name localnodeid.jesid. $JESNEWS.STCtaskid.Dnewslvl.JESNEWS localnodeid.jesname. $TRCLOG.taskid.dsnumber.JESTRACE localnodeid. +MASTER+.SYSLOG.jobid.dsnumber.SYSLOG	JESSPOOL	Restrict access to data on spool to authorized users
Input Sources o Readers o Internal Readers (INTRDR) o STCINRDR o TSUINRDR o Remote Job Entry (RJE) Workstations o Network Job Entry (NJE) Nodes o Spool Offload Receivers Note: TSUINRDR and TSOINRDR are used	RDRnn as defined on the RDR(nn) initialization statement INTRDR STCINRDR TSUINRDR OFFn.SR and OFFn.JR as defined in the initialization stream Note: TSUINRDR and TSOINRDR are used interchangeably.	JESINPUT (see note)	Restricts users submitting specific jobs to specific devices

JES2 Resource	**RACF Profile Name Format**	**RACF Classes**	**Class Purpose**
interchangeably.			
Job submission and cancellation Job Modify SSI (SSI 85) Job class access	SUBMIT.localnodeid.jobname.userid CANCEL.localnodeid.userid.jobname MODIFY.localnodeid,userid.jobname HOLD.localnodeid,userid.jobname RELEASE.localnodeid,userid.jobname PURGE.localnodeid,userid.jobname RESTART.localnodeid,userid.jobname SPIN.localnodeid,userid.jobname REROUTE.localnodeid,userid.jobname START.localnodeid,userid.jobname JOBCLASS.localnodeid.jobclass.jobname	JESJOBS (see note)	Controls which jobnames and userids users can use when submitting or cancelling jobs. Can also control which users can submit any jobs. Controls which jobnames and userids users can use when altering jobs by using SSI 85 (Job Modify SSI). Controls which jobnames have access to a job class based on submitter or owner of the job.
Local Commands	jesname.command[.qualifier]	OPERCMDS	Restrict commands to authorized users
Network Job Entry (NJE) Nodes	o nodeid.keyword.entity*. o NJE.ownnode.othernode SESSKEY(key)	NODES APPCLU	o Prevent processing of unauthorize d jobs or

JES2 Resource	RACF Profile Name Format	RACF Classes	Class Purpose
			sysout from another node o Extracting the encryption key to control NJE signons
Output Devices o Local and FSS devices o RJE devices o NJE devices o Spool Offload Transmitters	jesname.LOCAL.devicename jesname.RJE.devicename jesname.NJE.nodename	WRITER	Restrict processing of output to specific devices
Remote Job Entry (RJE) Workstations Job class access	RJE.workstation-id JES.JOBCLASS.OWNER JES.JOBCLASS.SUBMITTER	FACILITY	Prevent unauthorized sign-on by remotes Enables a jobs access to a job class.
Update JESNEWS	jesname.UPDATE.JESNEWS	OPERCMDS	Restrict ability to create, update, and delete JESNEWS
Note: At least one profile that defines all jobs must exist in this class when this class is active or all jobs will fail.			

Table 14: JES2 Resources protected by RACF

8.3.9.7 JES2 Input Control

Job Submission and Cancel Control

The installation might want to control the users who can submit or cancel certain jobs. By defining the jobname in class JESJOBS and a list of users authorized to submit the job to

RACF, SAF can determine if the user submitting (or canceling) the job has the authority to do so. The format of the RACF profile name for jobs is:

SUBMIT.nodename.jobname.userid

or

CANCEL.nodename.userid.jobname

where:

SUBMIT Controls which users can submit jobs.

CANCEL Controls which users can cancel jobs.

nodename The name of the node on which the submit or cancel will occur.

jobname The job name to be secured.

userid The userid associated with the job to be secured.

A profile for a user to submit or cancel jobs that have their own userid on the JOB statement is not needed. When granting a user the ability to submit a job, that userid should be permitted to the profile with an access level of READ. When granting access to cancel a job, the userid should be permitted to the profile with an access level of ALTER. The UACC and access lists associated with these profiles also allow to control which userids a user can use on JOB statements or whether the user can submit or cancel any jobs.

Authorizing Users to Submit Jobs for other Users

The RACF administrator can allow users to submit jobs on behalf of other users without compromising the pass-word of the user whose jobs are being submitted. The RACF administrator must define the userid of the person who will be submitting the jobs in the RACF SURROGAT class. Once this is done, the user just submits the job with the original userid and without a password on the JOB card. RACF then validates the job(s) as if the original user had submitted it and will use that userid's password and default security information when verifying the jobs authority for resources.

Authorizing Input Sources

RACF can be used to limit which sources of input are valid for job submission, including RJE workstations, device readers, nodes, and internal readers.

To authorize the submission of work from specific input sources, the security administrator needs to activate the RACF JESINPUT class and define a profile for each input source. The following information is relevant:

- The name of the device.
- The userid or groupid of the users to be authorized or restricted.
- The universal access authority to associate with each device. Valid access values for input devices are:
    - NONE Specifies that only those users explicitly permitted through the access list can use the input device.
    - READ Specifies that only those users assigned an access authority of read, update, or control can use the input device.

A user must have READ access to a device (either through the UACC or explicitly in an access list) in order to use an input device. If the device has a UACC of READ, the system administrator can restrict users from a device by listing them in an access list and specifying NONE for their access.

Note: By default, JES2 accepts all work from an input source when the RACF JESINPUT class is inactive.

JES2 Commands are described in [JES2.CMDS]. JES2 configuration is described in [JES2.IATG] and [JES2.IATR]. JES2 Messages are described in [JES2.MSG]. The Job Control Language (JCL) is defined in [MVSJCL.REF].

8.3.10 Time Sharing Option (TSO/E)

TSO/E is the primary user interface to the z/OS system. TSO/E provides numerous commands for both end users and system programmers that allow them to interact with TSO/E and the z/OS system. The ALLOCATE, FREE, and EDIT commands are examples of commands that allow users to manage their data sets. The TEST and TESTAUTH commands let system programmers test assembler language programs, including command processors, APPC/MVS transaction programs, and other programs written in assembler language.

The CONSOLE command lets users with CONSOLE command authority perform z/OS operator activities from a TSO/E session.

The system administrator can use RACF to protect TSO resources relevant to authentication and logon. These resources include TSO logon procedures, account numbers, and performance groups. In addition, the system administrator can protect resources called TSO user authorities, whose settings determine whether a user can is-sue certain authorized TSO commands. See [TSO.CUST] for further information.

With RACF installed (as is required for the evaluated configuration), the information that is required for users to log on to TSO/E is stored in the RACF data base and RACF is called by TSO for user authentication.

Interfaces of TSO/E are:

- The TSO/E commands described in [TSO.CMDS] and [TSO.SYSPROG].
- The programming interface for TSO services is described in [TSO.PROG] (Note: This document also describes a number of services provided by TSO utilities. Those are not part of the TSF, since those utilities execute as normal user programs with the authorization of the user that invokes them).

8.3.11 UNIX System Services (USS)

The z/OS support for z/OS UNIX enables two open systems interfaces on the z/OS operating system:

- An application – XPG4 UNIX 1995 conforming – program interface (API). The application interface is composed of C interfaces. Some of the C interfaces are managed within the C Run-Time Library (RTL), and others access kernel interfaces to perform authorized system functions on behalf of the unauthorized caller.

- An interactive z/OS shell interface

With the APIs, programs can run in any environment—including in batch jobs, in jobs submitted by TSO/E users, and in most other started tasks—or in any other MVS™ application task environment. The programs can request:

- Only MVS services
- Only z/OS UNIX
- Both MVS and z/OS UNIX

The shell interface is an execution environment analogous to TSO/E, with a programming language of shell commands analogous to the Restructured eXtended eXecutor (REXX) Language. The shell work consists of:

- Programs run by shell users
- Shell commands and scripts run by shell users
- Shell commands and scripts run as batch jobs

The programming interfaces to UNIX System Services are defined in [UNIX.ASSEM] (for Assembler programming). Library interfaces for C (which then invoke the interfaces described in [UNIX.ASSEM]) are described in [C.LIB]. Note: [C.LIB] describes a large number of run-time library functions where many of those functions may not perform any call to a TSFI and some functions may perform fairly complex operations before calling specific TSFI. From the description of the functions in [C.LIB] it remains unclear if they call TSFI and if they do, which TSF functions are invoked. Only the analysis of the entry points within the TSF shows that the functions described in [UNIX.ASSEM] map almost one-to-one to TSF functions implemented using the Program Call system interface.

8.3.11.1 Kernel Call Interface

Connecting to and disconnecting from z/OS UNIX System Services

To connect to the kernel for z/OS UNIX System Services, the system administrator makes an address space known to it. This process is called dubbing. Once dubbed, an address space is considered to be a process. Ad-dress spaces created by fork are automatically dubbed when they are created; other address spaces become dubbed if they invoke a z/OS UNIX service. Dubbing also applies to MVS™ tasks. A dubbed task is considered a thread. Tasks created by pthread_create are automatically dubbed threads; other tasks are dubbed if they invoke a z/OS UNIX service.

Undub is the inverse of dub. Normally, a task (dubbed a thread) is undubbed when it ends. An address space (dubbed a process) is undubbed when the last thread ends.

If, when a thread or process is being dubbed, the calling task has a task-level ACEE that does not have a USP connected to it, an INITUSP is done against the task-level ACEE. This causes UNIX System Services security in-formation to be associated with the task-level ACEE.

8.3.11.2 Authentication

z/OS UNIX services provide authentication as part of the programming interface only. In particular the __passwd service is provided, which internally calls the RACF initACEE service.

The RACF user profile definition was expanded with a segment called OMVS for z/OS UNIX support. All users and programs that need access to z/OS UNIX must have a RACF user profile defined with an OMVS segment which has, as a minimum, a UID specified. A user without a UID cannot access z/OS UNIX.

8.3.11.3 Security Credentials Representation

The security credentials (CRED) structure is used in the z/OS UNIX file system to pass data from the logical file system (LFS) through the physical file system (PFS) to the RACF callable services.

The CRED is built by the LFS, and is created for each system call entry to the LFS. The CRED is used for all vm_ops called (and most RACF callable service calls by the PFS) for the system call. The CRED is not kept across multiple LFS system calls.

The CRED contains:

- User information: a user type field that indicates whether the caller is a standard z/OS UNIX process known to RACF, or a system function that is not a process.

  Functions that accept a system caller process the request as if the caller is a superuser. If an audit record is written, the user z/OS UNIX user identifier (UID) and z/OS UNIX group identifier (GID) values in the record are set to -1.
- Audit data: data known by the LFS that needs to be passed through the PFS to the RACF callable services for auditing. This data is:
  - Audit function code: a code that identifies the system call being processed.
  - Name flag: a flag used on path resolution calls to ck_access to indicate whether the first or second file name is being checked.
  - Requested path name: the path name the user passed on the system call. For link, vlink, rename, and vrename, this is the old path name. When the caller of lookup is getcwd, ioctl, or ttyname, this field is not filled in.
  - File name the part of the requested path name currently being checked. This may be part of the path name or may be part of a symbolic link encountered when resolving the path name. The first directory checked in a path name resolution is either the root directory (/ROOT) or the current working directory (/CWD). The names /ROOT and /CWD—the only file names that contain a slash (/)—are provided to indicate these directories in the audit record. This field is included only in audit records produced by ck_access. This field contains the file name of:
    - The directory being checked on calls from lookup. When the caller of lookup is getcwd, ioctl, or ttyname, this field is not filled in.

        - The parent directory of the object identified by the pathname for calls for mkdir, mknod, vcreate, open(new file), rename, vrename, rmdir, symlink, vsymlink, unlink, and vremove.
        - The object identified by the path name for calls for open(old file), opendir, link, vlink, and utime.
        - Second path name: for rename, vrename, link, and vlink, this is the new path name passed on the system call. For symlink and vsymlink, this is the content of the symlink. For mount and unmount, this is the data set name of the HFS data set being mounted or dismounted.
        - Second file name: this is the same as the file name above, except that it is for the second part of the path name being checked. This field contains the file name of:
            - The directory being checked on calls from lookup
            - The parent directory of the object identified by the new pathname for calls for link, vlink, rename, and vrename.
        - Access Control List information: pointers to ACL buffers are used for the ck_access, makeFSP, and R_Setfacl callable services.
    - Address and ALET of the File System Name. These fields are provided for vn_ops issued for a file system ROOT of a zFS file system. The file system name is used in the ck_access processing to control access to a file system if the FSACCESS class is active and the file system is defined to RACF as a restricted resource.

8.3.11.4 User Authorization

z/OS UNIX users are TSO/E user IDs with a RACF segment called OMVS defined for z/OS UNIX use. All users that want to use z/OS UNIX services must be defined as z/OS UNIX users. Similar to users in a UNIX system, z/OS UNIX users are identified by a UID (user identification). The UID has a numerical value.

There are two types of users:

- User (regular user)
    - Identified by a non-zero UID.
- Superuser (authorized/privileged user). A superuser can be any of the following:
    - A z/OS UNIX user with a UID=0.
    - A started procedure with a trusted or privileged attribute in the RACF started procedures table.

8.3.12 Ported Tools (OpenSSH)

Secure Shell (SSH) is a network protocol which provides an alternative for insecure remote login and command execution facilities, such as telnet, rlogin and rsh. SSH encrypts traffic in both directions, preventing traffic sniffing and password theft. The SSH that is provided for z/OS is a port of OpenSSH 5.0p1, available from www.openssh.org.

On a local system, the user starts the ssh client to open a connection to a remote server running the sshd daemon. If the user is authenticated successfully, an interactive session is initiated, allowing the user to run commands on the remote system. ssh is not a shell in the sense of a command interpreter, but it permits the use of a shell on the remote system. In addition to interactive logins, the user can run remote commands, tunnel TCP network connections through the existing channel (allowing the use of X11 and other network-based applications), and copy files through the use of the scp and sftp tools. OpenSSH is configured to use SAF for password user authentication. Password expiration and locking are handled through the appropriate SAF functions. OpenSSH can also be configured to use SAF key rings to store DSA and RSA keys for user and host authentication.

8.3.12.1 Cryptographic Algorithms

The default cipher used by ssh is aes128-ctr (Advanced Encryption Standard (AES) in CTR mode with 128-bit key.) AES is an iterative, symmetric-key block cipher that can use keys of 128, 192, and 256 bits, and encrypts and decrypts data in blocks of 128 bits (16 bytes).

The "3des-cbc" cipher is threekey triple-DES (encrypt-decrypt-encrypt), where the first 8 bytes of the key are used for the first encryption, the next 8 bytes for the decryption, and the following 8 bytes for the final encryption. This requires 24 bytes of key data (of which 168 bits are actually used). To implement CBC mode, outer chaining is used (i.e., there is only one initialization vector). This is a block cipher with 8 byte blocks. This algorithm is defined in [FIPS-46-3].

The encryption algorithm can be assigned by providing the list in the server configuration file with keyword “Ciphers”.

The default setting of Ciphers is:

- "aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcfour

The following MAC algorithms are currently defined:

- hmac-sha1
    - HMAC-SHA1 (digest length = key length = 20)
- hmac-sha1-96
    - first 96 bits of HMAC-SHA1 (digest length = 12, key length = 20)
- hmac-md5
    - HMAC-MD5 (digest length = key length = 16)
- hmac-md5-96
    - first 96 bits of HMAC-MD5 (digest length = 12, key length = 16)

The default public key formats used for user and host authentication are DSA (“ssh-dss”) and RSA (“ssh-rsa”).

8.3.12.2 Key exchange protocol

The key exchange method specifies how one-time session keys are generated for encryption and for authentication, and how the server authentication is done. The Diffie-Hellman Key Exchange is a method for exchanging secret keys over a non-secure medium without exposing the keys.

The default key exchange methods in OpenSSH are:

- diffie-hellman-group-exchange-sha1
- diffie-hellman-group1-sha1
- diffie-hellman-group-exchange-sha256
- diffie-hellman-group14-sha1

The configuration, management and use of OpenSSH for z/OS is defined in [PORTED].

8.4 Security Functionality

8.4.1 Identification and Authentication (FIA, FTA)

A user can interact with the TOE in one of the following ways:

- As a TSO user
- As an operator at a console
- By submitting a job to be initiated and scheduled by the Job Entry Subsystem (JES2)
- As a UNIX user, including access via the UNIX shell or as a client of a UNIX-based server such as FTP, HTTP, SSH, TN3270, rsh, rexec, etc.
- As an LDAP user
- By using a NFS client supporting the z/OS extensions
- As a CIM user
- As a user of the RMF Distributed Data Server
- As a Communications Server Policy Agent or Network Security Server or Load Balancing Advisor client
- As a Kerberos principal

In all cases (except for the HTTP server or LDAP server that the administrator may optionally configure to allow selected access by unauthenticated users as described elsewhere) users are identified and authenticated by the TOE {IA.1::IA.1.1} before being authorized to perform any other security relevant action. In the case of jobs submitted by an already-authenticated user, no additional authentication is required for jobs running with the ID of the user who submitted them. The internal reader accepts (and relies) in this case on the authentication performed when the user has logged on to the system {IA.1::IA.1.2}.

An exception to this rule are started tasks, which operate under a protected user ID and are started either at system startup or through an operator command. Those tasks are not executing on behalf of a human user and their protected user IDs are exempt from authentication {IA.1::IA.1.3}. They must only be started from trusted data sets.

When authenticating a user the TOE allows applications to accept:

- A user ID defined to RACF {IA.1::IA.1.4-R8-RACF-1} and the RACF password {IA.1::IA.1.4-R8-RACF-2} or password phrase {IA.1::IA.1.4-R10-RACF-4} or a PassTicket {IA.1::IA.1.4-R8-RACF-3}.
- A valid x.509v3 digital certificate that the application has validated using TLSv1.2-, TLSv1.1-based client authentication and presented to RACF via initACEE (or indirectly via __certificate()) or mapped to a RACF user ID R_usermap(), for those applications supporting TLSv1.2-, TLSv1.1-based client authentication (see Authentication via Client Digital Certificates) . {IA.1::IA.1.4-V2R1-MULTI-1}{IA.1::IA.1.4-V2R1-RACFEAL5-1}
- A valid Kerberos service ticket for the client Kerberos principal, which the R_usermap() service will convert to a RACF user ID for use by the application, for those applications supporting Kerberos (see Authentication via Kerberos) {IA.1::IA.1.4-R13-MULTI-2} {IA.1::IA.1.4-R13-RACFEAL5-2}. The application may also request entry of a valid RACF user ID and password/phrase {IA.1::IA.1.4-R10-MULTI-3} and if so the application must run the user's session under that ID {IA.1::IA.1.4-R8-MULTI-5}.
- For SSH login functions (ssh, scp, sftp) RACF will also verify the specified password/phrase {IA.1::IA-1.4-R10-SSH-1}. For clients authenticating using public/private keys, SSH will verify the private key using information from the RACF keyring when configured to allow this authentication method {IA.1::IA-1.4-R12-SSH-2}
- The NFS server must be configured with SECURITY(SAF or SAFEXP) and in this mode will support authentication of the client via either Kerberos or the mvslogin command. If the client uses mvslogin and does not use Kerberos, then the NFS server will validate the user ID and password/phrase using RACF {IA.1::IA.1.4-R12-NFS-1} . If the client uses Kerberos then the NFS server will use SAF and RACF functions to map the Kerberos principal to a local RACF user ID and will use that ID for all NFS security functions {IA.1::IA.1.4-R11-NFS-2}.
- For LDAP authentication:
  - With an SDBM DN, the z/OS LDAP server accepts the DN and a RACF password/phrase, and presents the user ID from the DN, together with the password/phrase, to RACF for authentication. {IA.1::IA.1.4-R10-LDAP-1}.
  - With an LDBM DN, the z/OS LDAP server accepts the DN and a RACF password/phrase. It transforms the LDAP-style DN into a RACF user ID by lookup within the LDBM database, and presents the resulting RACF user ID and the supplied password/phrase to RACF for verification {IA.1::IA.1.4-R10-LDAP-2}.
  - With the ICTX plug-in (for remote authorization or remote auditing extended-operation requests) the z/OS LDAP server accepts an ICTX-format DN of the

form racfid=userid,cn=ictx and the RACF user’s password/phrase, and the plug-in presents the user ID from the DN, together with the password, to RACF for authentication. {IA.1::IA.1.4-R10-EIM-1}.

- Additionally, LDAP will allow authentication via a digital certificate presented over a TLS connection when doing an external SASL bind, and will map the certificate to a RACF user ID using TOE functions, failing the bind if RACF does not recognize the certificate. This will work for access to SDBM or LDBM data {IA.1::IA.1.4-R10-LDAP-3}. For SDBM, LDAP will provide that RACF user ID when accessing the SDBM back-end {IA.1::IA.1.4-R10-LDAP-4}. For LDBM, LDAP will transform the RACF user ID into an SDBM-style DN which (based on administrator-supplied LDAP configuration options) can either replace or supplement the DN contained in the certificate {IA.1::IA.1.4-R10-LDAP-5}.

- For authentication to the CIM server, CIM accepts a RACF user ID and password or PassTicket and uses RACF to validate them before allowing connection {IA.1::IA.1.4-R8-CIM-1}. Subsequently, if RACF validates the ID and password, the CIM server continues authentication by ensuring that the user has access to the CIMSERV resource in the (customer-defined) WBEM RACF class according to the type of request {IA.1::IA.1.4-R8-CIM-2}. In addition the CIM server uses pthread_security to process requests that access/manipulate system resources under the requestor’s user ID {IA.1::IA.1.4-R8-CIM-3}. For use of PassTickets the administrator can configure CIM to use either the standard z/OS UNIX application ID of OMVSAPPL, or to use a CIM-specific application ID of CFZAPPL {IA.1::IA.1.4-R10-CIM-4}.
- The RMF Distributed Data Server (DDS) authenticates users via a RACF user ID and password or PassTicket {IA.1::IA.1.4-R11-RMF-1}.
- The Communications Server Policy Agent Server {IA.1::IA.1.4-R9-CS-POLCEN-1} and Network Security Server {IA.1::IA.1.4-R9-CS-NSS-1} accept a RACF user ID and password or PassTicket from their clients during session initiation and use RACF to validate them before allowing connection.
- The Communications Server Network Security Services (NSS) XMLAppliance discipline SAFAccess service accepts a RACF user ID and password or PassTicket and validates them at the request of the XML Appliance client {IA.1::IA.1.4-R10-CS-XMLApp-1}. The SAFAccess service can also perform an access control check for a specified resource when requested to do so {AC.4::AC-R10-CS-XMLApp-4}.
- The Communications Server NSS also provides XMLAppliance services to:
  - Provide certificate management operations {SM.4::SM-R11-CS-XMLApp-5}
  - Provide private key operations to allow retrieval of private keys from RACF certificates {SM.4::SM-R11-CS-XMLApp-6}, and to perform RSA signature creation and RSA signature verification using ICSF-protected keys {SM.4::SM-R11-CS-XMLApp-7}.
- The Communications Server Load Balancing Advisor accepts a client digital certificate via TLS from its clients (Load Balancing Agents, external load balancers) and uses RACF to validate them before allowing connection {IA.1::IA.1.4-R10-CS-LBA-1}. Following a successful authentication, the Load Balancing Advisor further restricts

access by requiring that the user have READ access to SERVAUTH resource EZB.LBA.[one of: AGENTACCESS, LBACCESS]. <system-name>.<tcp-sysplex-group-name> {IA.1::IA.1.4-R10-CS-LBA-2}.

Some additional considerations:

- For access to UNIX functions, the user must have a valid UID and his default group must have a valid GID {IA.1::IA.1.6}.
- If the user is in additional groups they may have GIDs, too, and if so UNIX access checking will make use of those additional GIDs {IA.1::IA.1.6-R8-USS-3}.
- If the user ID is in REVOKE status, RACF prevents user from entering the system at all or entering the system with certain groups {IA.1::IA.1.7}.
- For a user defined as a system administrator (that is, one who has the system SPECIAL attribute) a message is displayed on the console asking the operator if the user shall be revoked if he exceeds the number of failed login attempts due to incorrect passwords {IA.1::IA.1.7-R8-RACF-1} or if he exceeds the system inactivity interval {IA.1::IA.1.7.R8-RACF-2}.
- For users in the TSO environment the administrator can impose restrictions on whether the user can use the system on this day and at this time of the day. This is checked only when using a terminal from a defined set. This does not apply to operator console login, telnet, rlogin, rsh, rexec, ssh, scp, sftp, LDAP, z/OS Network Authentication Service, HTTP, ftp, or to batch jobs {IA.1::IA.1.8}.
- RACF also checks if the user is authorized to access the terminal (which can also include day and time restrictions for accessing that terminal) or other port of entry {IA.1::IA.1.9}.
- RACF also checks if the user is authorized to use the application (if specified) {IA.1::IA.1.10}.
- A user may have SURROGAT authority for another user. This allows him to submit a job under the user ID of this other user without specifying the password or to use the z/OS UNIX su command to switch to this user’s ID without specifying the password {IA.1::IA.1.11}. It also allows appropriately-authorized servers (e.g., HTTP) to switch a session to run under a pre-specified ID {IA.1::IA.1.11-R8-MULTI-1}. The job runs with the user ID that the job card specifies, not the surrogate user’s user ID. The audit record for surrogate job submission identifies both the surrogate user and the jobcard user ID {IA.1::IA.1.13}.

8.4.1.1 RACF Passwords and Password Phrases

In RACF, the user selects his own password/phrase and only the user knows the value chosen. If the user has forgotten his password/phrase and it needs to be reset, the security administrator will reset the password/phrase {IA.2::IA.2.1-R10}. When the system administrator follows the rules for the evaluated configuration, this new password/phrase should be in an expired state, thus forcing the user to enter a new password/phrase on the next logon {IA.2::IA.2.2-R10}. When creating a new user ID for a pseudo-user that is not a protected user ID, the initial password/phrase may be marked as non-expired, allowing it to be used without being changed first. {IA.2::IA.2.3-R10}.

Password Quality

A system administrator can set a variety of system-global rules for forming valid passwords using the SETROPTS command (for system-wide settings) or (to a lesser extent) using the password command to affect only one user. He can change such parameters as the number of days a password is valid for, how long to maintain password history to prevent the user from reusing the same password again, the minimum number of days between password changes, and syntax rules for password content.

When a user changes a password, RACF treats the new, user-supplied password as an encryption key to transform the RACF user ID into an encoded form using the DES algorithm that it stores on the database. The password is not stored in clear text {IA.2::IA.2.4}.

The following system-wide options can be set to enforce a minimum strength of passwords using the PASSWORD option in the SETROPTS command:

- Minimum and maximum length of passwords (LENGTH(m1:m2) as part of a RULE suboption) {IA.2::IA.2.5}
- Maximum password lifetime (INTERVAL suboption) {IA.2::IA.2.6} and minimum passwordchange time (MINCHANGE option) {IA.2::IA.2.V1R7-1}
- Number of passwords from the user’s password history that are not allowed for a new password (HISTORY suboption) {IA.2::IA.2.7}
- Maximum number of consecutive failed authentication attempts until the REVOKE attribute is set in the user’s profile (REVOKE suboption) {IA.2::IA.2.8}
- Differentiate between upper- and lowercase characters with the PASSWORD(MIXEDCASE) option {IA.2::IA.2.V1R7-2}
- Type of character for each character position of a password. Possible types are {IA.2::IA.2.9}:
    - ALPHA
    - ALPHANUM (which includes also the special characters $, # and @)
    - VOWEL
    - NOVOWEL
    - CONSONANT
    - NUMERIC
    - MIXEDCONSONANT
    - MIXEDVOWEL
    - MIXEDNUM
    - NATIONAL

If the value ALPHANUM is defined for more than one position in the password, at least one alphabetical value and one numeric value are required by RACF.

When the commands are called in a way that allows the TOE to suppress printing, passwords are not displayed:

- when entered at a TSO terminal as part of the login process {IA.2::IA.2.10}, or
- when entered at a TSO terminal as part of the ADDUSER, ALTUSER, or PASSWORD commands when the command contains the PASSWORD keyword but no value {IA.2::IA.2-R10-RACF-21}, or
- when entered into one of the RACF-supplied ISPF panels that allows specification of a password {IA.2::IA.2-R10-RACF-22}, or
- when entered at a system operator console as part of the operator logon {IA.2::IA.2-R8-BCP-1}, or
- when the content of a jobcard is displayed as part of a job's output {IA.2::IA.2.13}.

Note that the TSF can not ensure that passwords entered into programs executing with the user's privilege are fully protected from being spoofed. The user has to take care about his password in those cases as explained in the guidance.

Note that, as previously mentioned, for a local Kerberos user, when using RACF as the KDC's registry, the user's RACF password/phrase and Kerberos password are the same.

Password Phrase Quality

Many of the system rules for passwords set by SETROPTS apply to password phrases, too. However, RACF does not provide support for content syntax rules when using password phrases.

When a password phrase is established for a user, RACF treats the new phrase as a sequence of encryption keys to transform the RACF user ID into an encoded form using the DES algorithm with chaining, that it then stores on the database. The password phrase is not stored in clear text {IA.2::IA.2-R10-RACF-1}.

The following system-wide options that can be set to enforce a minimum strength of passwords using the PASSWORD option in the SETROPTS command also apply to password phrases:

- Maximum password phrase lifetime (INTERVAL suboption) {IA.2::IA.2-R10-RACF-2} and minimum password phrase change time (MINCHANGE option) {IA.2::IA.2-R10-RACF-3}
- Number of password phrases from the user's password phrase history that are not allowed for a new password phrase (HISTORY suboption) {IA.2::IA.2-R10-RACF-4}
- Maximum number of consecutive failed authentication attempts using a password or password phrase until the REVOKE attribute is set in the user's profile (REVOKE suboption) {IA.2::IA.2-R10-RACF-5}

Rather than having an administrator specify syntax rules to specify valid password phrase content, RACF enforces the following set of predefined rules:

- maximum length: 100 characters in the absence of exit ICHPWX11 {IA.2::IA.2-R10-RACF-6}

Note: The evaluated configuration of the TOE generally does not allow customers to implement exits to change the system processing. However, RACF supplies a sample ICHPWX11 exit and a sample REXX exec IRRPHREX that the sample ICHPWX11 will invoke. The administrator may install the sample ICHPWX11 unmodified, and may specify tailoring options in IRRPHREX to apply some additional syntax/content rules.

- minimum length:
  - 14 characters in the absence of exit ICHPWX11 {IA.2::IA.2-R10-RACF-7}
  - 9 characters if exit ICHPWX11 is present and allows the phrase {IA.2::IA.2-R10-RACF-8}
- The phrase may not contain the user ID, in either sequential uppercase or sequential lowercase characters {IA.2::IA.2-R10-RACF-9}
- The phrase must contain at least two alphabetic characters (A-Z, a-z) {IA.2::IA.2-R10-RACF-10}
- The phrase must contain at least two non-alphabetic characters (numeric, punctuation, special (including blanks)) {IA.2::IA.2-R10-RACF-11}
- The phrase may not contain more than two consecutive identical characters {IA.2::IA.2-R10-RACF-12}

If the administrator chooses to install the supplied sample exit ICHPWX11, the sample REXX exec IRRPHREX may then apply the following additional checks, if selected by the administrator, and may then accept a shorter phrase or reject a phrase that RACF would have accepted:

- The administrator can set the minimum allowable phrase length to a value between 9 and 100 inclusive by setting variable Phr_minlen {IA.2::IA.2-R10-RACF-26}
- The administrator can set the maximum allowable phrase length to a value between 9 and 100 inclusive by setting variable Phr_maxlen {IA.2::IA.2-R10-RACF-13}
- The administrator can set a more restrictive set of characters for password phrases by setting the variables numbers, letters, special, and Phr_allowed_chars {IA.2::IA.2-R10-RACF-14}
- The administrator can prevent leading or trailing blanks in password phrases by setting the variables Phr_leading_blanks or Phr_trailing_blanks to “no” IA.2::IA.2-R10-RACF-15}
- The administrator can prevent use of password phrases that contain a case-insensitive character string from the user's name by setting the variable Phr_name_allowed to “no” and setting the variable Phr_name_minlen to the longest substring allowed {IA.2::IA.2-R10-RACF-16} Example: if the user's name is John Smith the administrator could prevent the user from specifying a phrase containing John or john or jOhn or Smith by appropriate settings of the variables.
- The administrator can enable a triviality check by setting the variable Phr_triviality to “yes”. This will prevent use of a new password phrase that differs from the old one only insertion/deletion of spaces or changing character case. It also will reject a new

phrase when the shorter of the old and new phrases is simply a substring of the other. {IA.2::IA.2-R10-RACF-17}

- The administrator can prevent use of new phrases that do not differ in a significant number of characters from the old phrase by setting the variable Phr_min_unique to the number of positions that must differ. In addition, if the variable Phr_min_unique_norm has the value “yes” the exec will first normalize the old and new phrases to be checked by converting them to uppercase and removing spaces. {IA.2::IA.2-R10-RACF-18}
- The administrator can prevent the user of a new phrase which simply reorders the words of the old phrase by setting the variables Phr_unique_words (number of words that must be unique), Phr_word_minlen (minimum length of the unique words), and Phr_word_unique_upper (if “yes” then the exec will convert the old and new phrases to uppercase for this check {IA.2::IA.2-R10-RACF-19}
- The administrator can provide a list of disallowed words by setting the variables Phr_dict.0 to the number of words in a supplied list, and supplying the list in variables Phr_dict.1, Phr_dict.2, etc. {IA.2::IA.2-R10-RACF-20}

When the commands are called in a way that allows the TOE to suppress printing, the phrase is not displayed:

- when entered at a TSO terminal as part of the login process {IA.2::IA.2-R10-TSO-23}, or
- when entered into one of the RACF-supplied ISPF panels that allows specification of a password phrase {IA.2::IA.2-R10-RACF-25}.

Note that the TSF can not ensure that password phrases entered into programs executing with the user's privilege are fully protected from being spoofed. The user has to take care about his password phrase in those cases as explained in the guidance.

RACF PassTickets

PassTickets provide a one-time {IA.2::IA.2.14-R8-RACF-1} (by default, though administrators can change that for selected applications {IA.2::IA.2.14-R8-RACF-2}), cryptographically-computed, password substitute that may be used to authenticate a user {IA.2::IA.2.14-R8-RACF-3}. The computed value comprises information about the user ID, the application to which the user is authenticating, and the date and time-of-day {IA.2::IA.2.14-R8-RACF-4}. A given PassTicket is usable only within a time interval of plus-or-minus 10 minutes from the time of generation {IA.2::IA.2.14-R8-RACF-5}.

The cryptographic computation of a PassTicket requires usage of a secret key assigned by the administrator, and (for computations on z/OS) maintained within a profile in the PTKTDATA class. PassTicket evaluation also uses PTKTDATA profiles to determine the appropriate secret key to use.

For PassTicket generation, RACF locates a PTKTDATA profile whose name matches the application name, and extracts the secret key from it. The generation of the PassTicket then proceeds, using the user ID, application name, time/date, and selected key as inputs to the generation algorithm.

For PassTicket evaluation, RACF receives a user ID, application name, and optionally a group name, and locates a PTKTDATA profile to determine the secret key using a series of profile lookups, until a matching profile is found:

1. application-name.group-name.user-ID {IA.2::IA.2.14-R8-RACF-6}
2. application-name.user-ID {IA.2::IA.2.14-R8-RACF-7}
3. application-name.group-name {IA.2::IA.2.14-R8-RACF-8}
4. application-name {IA.2::IA.2.14-R8-RACF-9}

z/OS UNIX uses, by default, an application name (APPLID) of OMVSAPPL {IA.2::IA.2.14-R10-USS-1} when authenticating users via:

- The __login(), or pthread_security_np() services.
- The _passwd() service if issued from a thread created by pthread_create() which subsequently issued pthread_security_np(), and if the _passwd() call does not specify a new password.

The application may override this default in one of these ways:

- For pthread_security_np() and __passwd(), the application can
    - update the BPXYTHLI control block to indicate that z/OS UNIX should instead use the job name as the APPLID value {IA.2::IA.2.14-R10-USS-2}, or
    - update the BPXYTHLI control block to indicate a specific APPLID to use {IA.2::IA.2.14-R10-USS-3}.
- By changing to use one of the corresponding new services pthread_security_applid_np(), __login_applid(), and __passwd_applid() the application can specify an APPLID value directly as a parameter on the call {IA.2::IA.2.14-R10-USS-4}.

RACF provides two services for generation of PassTickets:

1. An internal service usable only by key 0 callers and located via the RCVT (RCVTPTGN); {IA.2::IA.2.14-R8-RACF-10}
2. An external service usable by appropriately authorized users or servers, and invoked by R_ticketserv() or R_gensec() {IA.2::IA.2.14-R8-RACF-11}. To use one of these services for PassTicket generation the caller needs UPDATE authority to resource IRRPTAUTH.application-name.target-user-ID in the PTKTDATA class. {IA.2::IA.2.14-R8-RACF-12}

RACF also allows applications to evaluate PassTickets by using the R_ticketserv() or R_gensec() services {IA.2::IA.2.14-R8-RACF-13}. Use of these services for PassTicket evaluation requires READ authority to IRRPTAUTH.application-name.target-user-id in the PTKTDATA class {IA.2::IA.2.14-R8-RACF-13a}.

z/OS also allows Java applications running on z/OS to generate or evaluate PassTickets, using a JNI interface to R_ticketserv() and R_gensec() {IA.2::IA.2.14-R8-RACF-14}.

The Communications Server uses PassTickets as part of its participation in the Express Logon Facility (ELF) and Web Express Logon (WEL) single signon solutions.

- Express Logon Facility (ELF) --This function is provided in a Two-Tier or Three-Tier model for single-signon to a z/OS application. With either model, a user presents an X.509v3 digital certificate to the z/OS ELF service, which in the two-tier model is a TN3270 server and in the three-tier model is a Digital Certificate Access Server (DCAS). When the TN3270 server or DCAS receives the certificate and a target application name, it will invoke RACF to : 1) map the certificate to a RACF user ID 2) Generate a PassTicket for the user ID and target application. {IA.2::IA.2.14-R9-ELF-1} In the two-tier model DCAS is not involved and the TN3270 server runs on z/OS. Here, the ELF function is agreed upon by the TN3270 sever and client. When the TN3270 server receives the logon panel (by examining the input data), it will invoke the RACF services to map the certificate to a user ID and generate a PassTicket, which it will then insert the user ID and PassTicket into the logon panel, subsequently passing the panel to the target application for logon. (IA.2.61) {IA.2.14-R9-ELF-2} In the case of the three-tier model, the TN3270 server does not run on z/OS, but runs on a distributed platform. In this case, the distributed TN3270 server (upon receipt of the logon panel), invokes DCAS, passing it the certificate and target application name (on behalf of the end user). DCAS then invokes RACF to map the certificate to a User ID and generate a PassTicket. DCAS passes this information back to the TN3270 server which inserts the User ID and PassTicket into the logon panel, and subsequently passes the panel to the target application for logon. {IA.2::IA.2.14-R9-ELF-3}
- Web Express Logon (WEL) --In this model (non certificate-based), a DCAS client is requesting a PassTicket on behalf of an end user. Note that as part of the single-signon architecture, that end user has already been authenticated at some point prior to the DCAS client requesting the PassTicket. In this case, the DCAS server supports two types of requests:
    1. It can receive a valid z/OS user ID from the client and the target application name. It will pass these to RACF requesting a PassTicket for that user ID and application. {IA.2::IA.2.14-R9-WEL-1}
    2. It can receive a principal name from the client along with the target application name. It will pass these to RACF requesting a z/OS user ID that has been mapped to the principal name and a PassTicket which will be returned to the requesting client. In this case, it is required that the z/OS user ID be mapped to a principal name using the RACF KERBLINK class. {IA.2::IA.2.14-R9-WEL-2}
- Additional details:
    - For ELF, in the two-tier model, communication between the TN3270 client and server requires SSL with client authentication at a minimum. {IA.2::IA.2.14-R9-ELF-4}
    - For ELF, in the three-tier model, communication between the distributed TN3270 server and DCAS requires SSL with client authentication at a minimum. The SSL and DCAS client in this case is the TN3270 server itself. In the evaluated configuration the DCAS server will also verify that that its client (the TN3270 server) is authorized to SERVAUTH resource EZA.DCAS.system-name. {IA.2::IA.2.14-R9-ELF-5}

- For WEL, communication between the DCAS server and client (WEL server) requires SSL with client (WEL server) authentication at a minimum. In the evaluated configuration the DCAS server will also verify that its client (the WEL server) is authorized to SERVAUTH resource EZA.DCAS.system-name. {IA.2::IA.2.14-R9-WEL-3}

Additionally, the Communications Server provides the DCAS server (Digital Certificate Application Server) which can be used by applications running in the network, perhaps as part of a single-signon service. DCAS provides two functions:

1. Generate a PassTicket for an application-specified user ID and application name; {IA.2::IA.2.14-R8-DCAS-1}
2. Map an application-specified digital certificate for the server's client to a RACF user ID, and generate a PassTicket for that user and an application-specified application name. {IA.2::IA.2.14-R8-DCAS-2}In order to use DCAS, the network-based application must connect to DCAS using an SSL session with client authentication, and provide its own digital certificate that maps to a RACF user ID {IA.2::IA.2.14-R8-DCAS-3}. In the evaluated configuration that mapped user ID must be authorized to resource EZA.DCAS.system-name in the SERVAUTH class {IA.2::IA.2.14-R8-DCAS-4}.

The Communications Server's Network Security Services Server provides a SAFAccess service as part of its XMLAppliance discipline. The SAFAccess performs RACF userid authentication on behalf of the XMLAppliance client and supports passwords and PassTickets as authentication tokens. NSS clients must connect to the NSS server using a TLS-protected session and must also authenticate themselves to the server using their own RACF userid and password or PassTicket {IA.2::IA.2.14-R10-CS-XMLApp-2}.

PassTickets are also used internally by the Kerberos KDC server as part of the processing when users change their Kerberos passwords.

Authentication via Client Digital Certificates

In the evaluated configuration, TLS-aware applications, or the Application-Transparent TLS (AT-TLS) functions of the Communications Server, can accept client certificates and map them to RACF user IDs as part of the client authentication process. Such applications must be configured to use RACF to store the keyrings that contain the application private key and the allowed Certificate Authority (CA) certificates that may be used to provide the client certificates that the application will support. The security administrator will use RACDCERT to establish those keyrings, which may reside in RACF profiles in the DIGTRING class or in PKCS#11 tokens maintained in ICSF, and thus to approve of any CAs that will be used. Any CA used in the evaluated configuration must support Certificate Revocation Lists (CRLs) maintained in an LDAP registry, and the security administrator must configure the application to use the CRLs. This configuration may be application-specific, or may be done by establishing LE environment variables that System SSL will use in the absence of specific application-provided CRL configuration information.

The first step in the client authentication process is for the server or AT-TLS to acquire the client certificate via the standard TLS data flows. As part of that processing, System SSL will validate the client certificate using the gsk_validate_certificate_mode() function, passing the validation mode to be applied to the validation processing.

System SSL can validate certificates using either the processing specified by [RFC2459], by [RFC3280], or by [RFC5280], under control of an environment variable or specifications provided by the application. In the absence of an environment variable or application-provided specification, System SSL will first validate using [RFC2459] and if that fails will then retry using [RFC3280]. If this also fails it attempt validating against [RFC5280] {IA.2::IA.2.15-V2R1-SSL-20}.

gsk_validate_certificate_mode will perform the following checks against the client certificate and certification chain:

1. The certificate's subject name must be identified by either a non-empty distinguished name (with an optional SubjectAltname certificate extension) or an empty distinguished name with a SubjectAltName certificate extension: RFC2459: {IA.2::IA.2.15-R8-SSL-1}.
   RFC3280: {IA.2::IA.2.15-R11-SSL-15} SubjectAltName Extension must be critical.
2. Certificate Authority certificates must have a non-empty subject name {IA.2::IA.2.15-R8-SSL-2}.
3. The certificate issuer name must not be an empty distinguished name {IA.2::IA.2.15-R8-SSL-3}.
4. The CertificatePolicies extension, if present, must not be marked critical (RFC2459) {IA.2::IA.2.15-R8-SSL-4}. For RFC3280, if the certificate policies extension is present, it must be marked critical and must satisfy the policies defined by the issuing certificate chain. {IA.2::IA.2.15-R11-SSL-16}
5. The current time must not be earlier than the start of the certificate validity period {IA.2::IA.2.15-R8-SSL-5}.
6. The issuer's certificate must be a valid CA certificate, and the root certificate and any intermediate signing certificates not in the client’s message must be present in the server’s key ring {IA.2::IA.2.15-R8-SSL-7}. The server's certificate store can either be a RACF key ring (DIGTCERT class) or a PKCS#11 token in ICSF TKDS (CRYPTOZ class) {IA.2::IA.2.15-R9-SSL-14}.
7. The certificate signature must be correct and using a supported signature (RSA 1024-4096 bit key with hashing algorithm --MD5, SHA-1, SHA-224, SHA-256, SHA-384 or SHA-512 or DSA 1024-bit key with hashing algorithm SHA-1) {IA.2::IA.2.15-R10-SSL-8}
   or ECDSA 160-521 bit keys with hashing algorithm SHA-1, SHA-224, SHA-256, SHA-384, SHA-512) {IA.2::IA.2.15-R12-SSL-21}
8. No certificate in the certification chain can be revoked or expired {IA.2::IA.2.15-R8-SSL-10}.
9. For CA certificates, the BasicConstraints extension, if present, must have the CA indicator set and the path length constraint must not be violated by subordinate certificates in the certification chain RFC2459 – {IA.2::IA.2.15-R8-SSL-11}
   RFC3280 -- {IA.2::IA.2.15-R11-SSL-17} --Basic Constraints extension must be present.
10. If the issuing certificate chain has defined any name constraints through Name Constraints extensions, the constraints must not be violated by the subject certificate {IA.2::IA.2.15-R8-SSL-12}.

11. The key usage extension, if present in a CA certificate, must specify signing capability
RFC2459 – {IA.2::IA.2.15-R8-SSL-13}
RFC3280 -- {IA.2::IA.2.15-R11-SSL-18} --Key Usage extension must be present.
12. Certificate Authority certificates with a Key Usage extension present which has the keyCertSign bit set must have a Basic Constraints extension present which has the CA indicator set. RFC3280 -- {IA.2::IA.2.15-R11-SSL-19}

After System SSL has validated the client certificate, the application (or AT-TLS) can map it to a RACF user ID via the R_usermap() callable service {IA.2::IA.2.16-R8-RACF-1}. Or the application can directly create a security environment for the user by using the pthread_security_np() service {IA.2::IA.2.16-R8-USS-1}, the InitACEE() service {IA.2::IA.2.16-R8-RACF-3}, or the _certificate() service {IA.2::IA.2-16-R9-USS-1} which will accept the certificate as input. In either case, RACF will:

1. Examine the RACF database and determine whether the certificate is installed and registered to a specific user. If so, return that user ID {IA.2::IA.2.17-R8-RACF-1}
2. Otherwise, try to find the best-matching mapping profile (DIGTNMAP class), and return the user ID specified in the profile's APPLDATA field:
   a) Check for a filter of subject's-full-name.issuer's-full-name {IA.2::IA.2.17-R8-RACF-2}
   b) Iteratively remove nodes from the subject's name and check for a filter of the form: subject's-partial-name.issuer's-full-name {IA.2::IA.2.17-R8-RACF-3}
   c) Check for a filter of the form: subject's-full-name {IA.2::IA.2.17-R8-RACF-4}
   d) Iteratively remove nodes from the subject's name and check for a filter of the form: subject's-partial-name {IA.2::IA.2.17-R8-RACF-5}
   e) Check for a filter of the form: issuer's-full-name {IA.2::IA.2.17-R8-RACF-6}
   f) Iteratively remove nodes from the issuer's name and check for a filter of the form: issuer's-partial-name {IA.2::IA.2.17-R8-RACF-7}
3. Otherwise, try to find the best-matching mapping profile (DIGTNMAP, DIGTCRIT class) that matches the mapping criteria specified by the application that presented the certificate to RACF, and if found return the user ID specified in the DIGTNMAP profile's APPLDATA field {IA.2::IA.2.17-R8-RACF-8}.
4. Otherwise, if the certificate contains at least one hostIDMappings extension with a host-name and user ID {IA.2::IA.2.17-R8-RACF-9} and the certificate was issued by a CA defined to RACF as having the HIGHTRUST status {IA.2::IA.2.17-R8-RACF-10}, then RACF will examine each of the hostIDMappings extensions, in order {IA.2::IA.2.17-R8-RACF-11}. RACF will determine whether the application has READ access to IRR.HOST.host-name in the SERVAUTH class, and if so RACF will return the user ID associated with that host-name {IA.2::IA.2.17-R8-RACF-12}.

Authentication via Public/Private Key (SSH)

OpenSSH supports authentication via public/private keys, however for the evaluated configuration OpenSSH on z/OS must be configured to obtain those public/private key pairs from digital certificates associated with RACF key rings. The existing RACDCERT command can be used to generate the keys and a certificate, or the certificates may be generated elsewhere and imported into RACF using RACDCERT. Public/private keys stored directly in the UNIX file system must not be used.

When a remote user authenticates to the OpenSSH server, the server will use the public key, obtained via a digital certificate which is associated with the user's configured key ring, to perform the authentication. {IA.2::IA.2-R12-SSH-KEY-1}

When a z/OS user acts as an SSH client, connecting to an SSH server, the client will obtain the necessary private key via a digital certificate which is associated with the user's configured key ring to perform the authentication. {IA.2::IA.2-R12-SSH-KEY-2}

The private key is not needed at the server when a client authenticates. The public keys must be distributed to remote hosts. When stored in a key ring, the certificate must be exported (via RACDCERT) and manually copied (e.g. via FTP) to the remote host, where it will be imported into that system's key ring.

When configured to use key rings, the OpenSSH server and client code will use existing System SSL interfaces to pull the keys from the RACF key ring, and the server and client will need authority to those key rings to use the RDATALIB service. {IA.2::IA.2-R12-SSH-KEY-3}.

Authentication via Kerberos

In the evaluated configuration Kerberos-aware applications can accept Kerberos service tickets from Kerberos clients (principals), map them to RACF user IDs, and allow them to access the system using their RACF identities. In addition, users running on z/OS may have Kerberos identities, and act as clients (Kerberos principals) to Kerberos-aware servers.

For authentication via Kerberos:

1. The client (principal) will obtain a Ticket Granting Ticket (TGT) by authenticating to the assigned Kerberos registry, which may be a z/OS Network Authentication Service instance KDC {IA.2::IA.1.4-R8-KERB-1} or some non-z/OS KDC. This initial authentication will follow standard Kerberos protocols, using one of the encryption protocols specified for the KDC {IA.2::IA.1.4-R8-KERB-2}. If the z/OS Network Authentication Service KDC is used for initial principal authentication, the z/OS Network Authentication Service will map the Kerberos principal name to a RACF user ID and the password used to derive the key info for the Kerberos authentication exchanges will be the user’s RACF password or phrase, whichever was last established for the user {IA.2::IA.1.4-R10-KERB-3}.
2. As is standard with the Kerberos protocol, the client will then acquire a service ticket for the desired server, and will present that ticket to the server for validation and mapping to a RACF identity.
3. The z/OS Network Authentication Service will validate the addresses in the service ticket, as required by RFC4120, if the administrator has requested the validation by specifying CHECKADDRS(YES) on the RDEFINE/RALTER for the KERB segment in the KERBDFLT profile in the REALM class. {IA.2::IA.1.4-R13-KERB-9}
4. {IA.2::IA.1.4-R12-KERB-8} If the service principal in a request is different from the principal specified on the krb5_rd_req, krb5_rd_req_verify, or gss_accept_sec_context API call, but the request is otherwise valid, the request will still be approved if and only if all the following are true:

- use_dvipa_override=1 is specified in the libdefaults section of the krb5.confconfiguration file,
- both principals are in the standard “Primary/Instance@Realm”format (with only one instance),
- both the primary and the realm in the request are the same as those provided in the API call, and
- the KRB5_SERVER_KEYTAB environmental variable is set to 2 and the application server that invoked the API call has at least READ access in the KERBLINK class to the principal specified in the request
  -OR-
  an entry in the keytab file matches the service principal name, version number, and encryption type from the incoming service ticket.

5. If the user is assigned to a foreign Kerberos realm (with respect to the TOE server application), the user will first use kinit to acquire a TGT from his local KDC. If a peer trust relationship is defined between the two KDCs, the client application can use this initial TGT to obtain a TGT for the remote z/OS KDC from its local KDC, which is then used by the client application to obtain a service ticket from the remote z/OS KDC. The z/OS KDC will only issue a service ticket for a TGT produced by a KDC in another realm if the administrator for each realm has configured a trust relationship between the two KDCs {IA.2::IA.1.4-R8-KERB-4}. This trust relationship may be transitive and involve the client contacting a series of KDCs before finally obtaining the TGT for the remote z/OS KDC {IA.2::IA.1.4-R8-KERB-5}.
6. If the application server is running on z/OS, once it has validated the client principal’s service ticket, it uses the R_usermap() service to determine the local RACF user ID associated with the Kerberos principal that may be defined to the z/OS Network Authentication Service {IA.2::IA.1.4-R8-KERB-6} or foreign {IA.2::IA.1.4-R8-KERB-7} principal that is defined to another Kerberos realm with an established trust relationship with the z/OS Network Authentication Service.

Started procedures

With the concept of a started procedure, the TOE provides a mechanism where a defined task can be started by an operator, but then operates under a defined user ID that is specifically assigned to the started procedure itself {IA.3::IA.3.1}.

A started procedure consists of a set of job control language statements that are frequently used together to achieve a certain result. Started procedures usually reside in the system procedure library, SYS1.PROCLIB, which is a partitioned data set. A started procedure is usually started by an operator, but can be associated with a functional subsystem. For example, SMS is treated as a started procedure even though it does not need to be specifically started with a START command.

Only RACF-defined users and groups can be specifically authorized to access RACF-protected resources {IA.3::IA.3.2}. Other users can access those resources with the authority allowed in the UACC entry of the RACF profile controlling access to the resource. However, started procedures have system-generated JOB statements that do not contain the USER, GROUP, or PASSWORD parameter.

To enable started procedures to access RACF-protected resources with other authorities than those defined in the UACC entry of the profile protecting the resource, started procedures must have RACF user IDs and group names {IA.3::IA.3.4}. By assigning them RACF identities, an installation can give started procedures specific authorization to access RACF-protected resources. For example, one can allow JES to access spool data sets.

To associate the names of started procedures with specific RACF group names and user IDs, an administrator can do one of the following:

- Set up the STARTED class (the recommended method)
- Create a started procedures table (ICHRIN03)

Assigning RACF user IDs to started procedures

As with any other user ID and group name, the user ID and group name that is assigned to a started procedure must be defined to RACF using the ADDUSER and ADDGROUP commands, and the user must be connected to the group. The administrator also needs to use the PERMIT command to authorize the users or groups to get access to the required resources.

Protected user IDs

The user IDs that an administrator assigns to started procedures should have the PROTECTED attribute unless the started procedure is required to have a user ID with a password defined. Protected user IDs are user IDs that have both the NOPASSWORD and NOOIDCARD attributes {IA.3::IA.3.5}. They are defined or modified using the ADDUSER and ALTUSER commands. Protected user IDs can not be authenticated via a password, password phrase, or RACF PassTicket, and are protected from being revoked through incorrect password attempts {IA.3::IA.3.6-R12-RACFEAL5}.

Authentication by trusted servers

Trusted servers of z/OS may be required to perform user authentication. They all use RACF to verify the credentials presented by a user for authentication. Those trusted servers may have some special configuration options that are explained in this section.

Handling of user authentication in the HTTP server

Users may connect to the HTTP server of the TOE. The server will assign an installation-defined pseudo-user ID to a user unless the user is authenticated with his user ID and password {IA.3::IA.3.V1R7.1}. Access checks to protected resources the HTTP server accesses on behalf of an unauthenticated user will be performed using the access rights of this installation-defined pseudo user ID {IA.3::IA.3.V1R7.2}.

The HTTP server also provides a function to identify and authenticate users using their user ID and password when the PROTECT directive specifies UserID %%CLIENT%% {IA.3::IA.3.V1R7.3}. Once authenticated successfully, the access rights of the authenticated user are checked when the HTTP server attempts to access resources protected by that PROTECT directive {IA.3::IA.3.V1R7.4}. The HTTP server uses RACF for user identification and authentication {IA.3::IA.3.V1R7.5}. Once the user has been successfully authenticated the HTTP server, when acting on behalf of the user, switches to the MVS user ID of the authenticated user and all access checks to protected resources are performed by RACF checking the access rights of this user {IA.3::IA.3.V1R7.6}.

The HTTP Server also supports client authentication via TLS client authentication using digital certificates. {IA.3::IA.3-R8-HTTP-1}. To enable this support, the administrator would specify UserID %%CERTIF%% on the PROTECT directive {IA.3::IA.3-R8-HTTP-2}. The HTTP server will present the certificate to RACF to map into a RACF user ID {IA.3::IA.3-R8-HTTP-3} and then proceed with access checking using that RACF identity as above for UserID %%CLIENT%% {IA.3::IA.3-R8-HTTP-4}.

Handling of user authentication in the FTP server

Users may connect to the FTP server and authenticate with a user ID and password or a Kerberos service ticket, or a digital certificate as previously described. The FTP server also supports unauthenticated, or anonymous, access to data. Administrators who have certain data that they want to serve to unauthenticated users via FTP may enable this anonymous access. Data access will then occur under a RACF ID that the administrator has specified in the FTP server configuration file, and only data accessible to that user will be served to the FTP client. Additionally, as this is intended to be “public” data with unrestricted access, no audit logs showing the actual human user who accessed the data can be maintained, but the administrator will have accepted the loss of auditing by configuring anonymous access.

In the evaluated configuration, if the administrator wishes to allow anonymous FTP access, the following parameters must be specified:

1. ANONYMOUSLEVEL 3
2. ANONYMOUS user-id/SURROGAT (Note: the administrator can choose any user-id he wants, but the user must have the RESTRICTED attribute, and an OMVS segment with a unique UID, a default group with a unique GID, a home directory to which the user has access, and should have no other group connections.).
3. ANONYMOUSFILEACCESS HFS or MVS or BOTH
4. ANONYMOUSFILETYPEJES FALSE
5. ANONYMOUSFILETYPESQL FALSE

With these settings:

- When the user specifies USER ANONYMOUS the FTP server will prompt for an email address {IA.3::IA.3-R9-FTP-1}.
- The FTP server will then establish a security environment for the chosen user ID from the ANONYMOUS statement {IA.3::IA.3-R9-FTP-2}.
- The FTP server’s ID must have SURROGAT authority to BPX.SRV.user-ID {IA.3::IA.3-R9-FTP-3}.
- The user ID must have an OMVS UID and its default group must have a GID {IA.3::IA.3-R9-FTP-4}.
- If starting in the UNIX file system or if the user issues a “cd” to switch to the UNIX file system, the FTP server will issue chroot() to restrict the user to his home directory as specified in the user’s OMVS segment {IA.3::IA.3-R9-FTP-5}.
- The user will only be able to access data in that home directory {IA.3::IA.3-R9-FTP-6}, or if the user switches to the MVS file system (assuming the administrator specified ANONYMOUSFILEACCESS MVS or BOTH) the user will have access to only that data to

which the user ID or his group(s) are explicitly permitted {IA.3::IA.3-R9-FTP-7}. No access to other MVS data via UACC or ID(*) or GLOBAL will be permitted {IA.3::IA.3-R9-FTP-8}.

The user will not be able to specify SITE FILETYPE JES nor SITE FILETYPE SQL (IA.3.24) {IA.3-R9-FTP-9}

If desired, the administrator can configure the FTP server to verify an authenticated user's authority to access the server via a SERVAUTH resource check before allowing access to data. To do this, the administrator would specify VERIFYUSER TRUE in the FTP configuration parameters and define a RACF SERVAUTH profile to protect the resource EZB.FTP.<system-name>.<ftp-daemon-name>.PORT<nnnn> where nnnn represents the port number assigned to that FTP daemon. The user will need READ access to that resource if it is protected by a SERVAUTH profile {IA.3::IA.3-R10-FTP-9}.

The administrator may also prevent the user from accessing data in the UNIX file system (thus restricting the user to accessing traditional MVS data sets). To do this, the administrator would define a profile in the SERVAUTH class to protect the resource EZB.FTP.<system-name>.<ftp-daemon-name>.ACCESS.HFS and deny the user access to the resource via the profile UACC or access list {IA.3::IA.3-R10-FTP-10}.

Handling of user authentication in the CIM server

Users may connect to the CIM server and authenticate with a RACF user ID and password or with a RACF user ID and PassTicket {IA.3::IA.3-R10-CIM-1}. The CIM server first uses RACF services to validate the user ID and password or PassTicket. In addition for all user requests that are to obtain or manipulate system management data, the CIM server dispatches the request to an extra thread, for which the effective userid is switched to that of the requestor using the pthread_security_np() service . This way the access to system resources occurs on behalf of the user's identity rather than under the identity of the CIM server {AC.1::AC.1-R10-CIM-2}.

Depending on the type of request the CIM server then ensures that the user has the proper level of access to the CIMSERV resource in the (customer defined) WBEM RACF class. For read access to the system data exposed by the CIM server the user requires READ access, for manipulation of system resources the user requires UPDATE access and for performing administrative tasks against the CIM server itself the user requires CONTROL access to the CIMSERV RACF resource {AC.1::AC.1-R10-CIM-3}.

Handling of user authentication in the LDAP server

LDAP user authentication in the evaluated configuration will occur via digital certificates over TLS (LDAP SASL bind with EXTERNAL verification) or via an LDAP DN and a RACF password/phrase.

For users initiating a bind operation with a DN and a password/phrase, the processing that occurs will depend on the style of DN presented: LDBM, SDBM, or ICTX:

- If the user presents an SDBM-style DN (such as racfid=ID1,profiletype=user,SDBM-suffix) then LDAP will extract the racfid value. Note that the SDBM-suffix is configured by the administrator. LDAP will pass that user ID and password/phrase to RACF for authentication {IA.3::IA.3-LDAP-1}.
- Similar processing happens when users present an ICTX-style DN. Again, LDAP recognizes this based on a configured suffix value, and invokes the ICTX plug-in,

which will pass the RACF user ID from the DN and the password to RACF for authentication {IA.3::IA.3-R9-EIM-1}.

- For LDBM users, the “native authentication” functions of the server are required for any authenticated access to LDBM in the evaluated configuration. For each LDBM user, the LDAP administrator will define the user’s distinguished name (DN) in the LDBM database, together with the RACF user ID that corresponds to that DN. The LDAP LDBM user will provide his LDAP DN and the RACF password/phrase for the user ID specified by the administrator. LDAP will find the user-specified DN, then call RACF passing the administrator-specified user ID and the user-specified password/phrase {IA.3::IA.3-R8-LDAP-2}. Note that the evaluated configuration allows the administrator to configure selected LDBM data for access by users who have not authenticated, if the administrator decides that such access meets the security policies in effect for that data.

For a SASL bind with a digital certificate (possible only for SDBM or LDBM in this evaluation), the evaluated configuration requires the administrator to configure LDAP to map the certificate to a RACF user ID. The administrator must specify the configuration option sslMapCertificate with a first operand of CHECK, ADD, or REPLACE and a second operand of FAIL. With this configuration in effect:

- If no RACF user ID is associated with the certificate, LDAP will fail the bind operation {IA.3::IA.3-R10-LDAP-2}.
- The mapped RACF user ID will be used for any access to the SDBM back-end {IA.3::IA.3-R10-LDAP-3}.
- For LDBM operations:
    - With ADD specified, LDAP will convert the mapped RACF user ID into an SDBM DN, and will add that DN to the DN from the certificate, using both DNs for group gathering and access decisions {IA.3::IA.3-R10-LDAP-4}.
    - With REPLACE specified, LDAP will convert the mapped RACF user ID into an SDBM DN, and will use only that DN for group gathering and access decisions {IA.3::IA.3-R10-LDAP-5}.

Authentication Method Summary

The following TOE applications support client authentication via Kerberos in the evaluated configuration:

- FTP {IA.3::IA.3-R8-FTP-AUTHKERB}
- ORSH {IA.3::IA.3-R8-RSH-AUTHKERB}
- OTELNET {IA.3::IA.3-R8-TELNET-AUTHKERB}
- NFS {IA.3::IA.3-R8-NFS-AUTHKERB}

The following TOE applications support client authentication via digital certificates when using TLS sessions in the evaluated configuration:

- TN3270, when using a TN3270 emulator that supports the Express Logon Facility (ELF) {IA.3::IA.3-R8-TN3270-AUTHSSL}

- FTP {IA.3::IA.3-R9-FTP-AUTHSSL}
- HTTP Server {IA.3::IA.3-R8-HTTP-AUTHSSL}
- LDAP Server, for SDBM or LDBM access {IA.3::IA.3-R10-LDAP-AUTHSSL}

The following TOE functions support authentication using passwords/phrases in the evaluated configuration:

- TSO/E {IA.3::IA.3-R10-TSO-AUTHPHRASE}
- NFS {IA.3::IA.3-R12-NFS-AUTHPHRASE}
- OpenSSH {IA.3::IA.3-R10-SSH-AUTHPHRASE}
- The z/OS UNIX shell commands su and passwd {IA.3::IA.3-R10-USS-AUTHPHRASE-1}
- The z/OS UNIX rlogin command {IA.3::IA.3-R10-USS-AUTHPHRASE-2}
- The C runtime functions __login(), __passwd(), pthread_security_np() (and the variants that accept an APPL ID), and getpass() {IA.3::IA.3-R10-LE-AUTHPHRASE}
- LDAP Server for SDBM or LDBM (via native authentication) access {IA.3::IA.3-R10-LDAP-AUTHPHRASE-1}
- FTP Server {IA.3::IA.3-R13-FTP-AUTHPHRASE}
- TN3270 Server {IA.3::IA.3-R13-TN3270-AUTHPHRASE}

OpenSSH supports authentication via public/private key pairs stored in digital certificates when it is configured to store the certificates in RACF key rings {IA.3::IA.3-R12-SSH-AUTHRINGS}.

Handling of Groups During Authentication

During authentication, RACF and LDAP construct security information that represents the user (subject) for subsequent use during access checking.

- During RACF authentication, RACF determines whether list-of-groups processing is in effect or not. If list-of-groups is not in effect, RACF puts the user's default group into the subject's ACEE, or the group specified by the caller of the RACF interfaces for user authentication. If list-of-groups is in effect, RACF gathers a list of all the groups to which the user is connected, and makes a copy of that list in the subject's ACEE. During access checking (DAC) for MVS resources, RACF can then base its decisions on both the user ID and on the group membership of the user {IA.3::IA.1.14-R12-RACFEAL5-1}.
- When a user attempts to use UNIX functions, RACF selects from the group(s) in the subject's ACEE up to the first 300 (alphabetically) which have OMVS segments with GIDs defined. During access checking (DAC) for UNIX resources, RACF can then base its decisions on the user's UID and the selected groups' GIDs {IA.3::IA.1.14-R10-RACF-2}.
- For access to LDAP LDBM resources, the LDAP server gathers a list of groups based on the authentication data supplied by the user.

- If the user supplied an LDBM-format DN and a RACF password/phrase, LDAP uses that DN to determine the LDBM groups to use on subsequent LDBM access checks, unless the DN is a member of the LDAP administrative group {IA.3::IA.1.14-R13-LDAP-1}.
- If the user supplied an SDBM-format DN and a RACF password/phrase, LDAP retrieves the subject's group(s) from the ACEE, and converts them into SDBM-format DNs, which become the groups to use for subsequent LDBM access checks {IA.3::IA.1.14-R10-LDAP-2}. Additionally, if LDAP is configured for extended group searching, LDAP derives additional groups by determining the LDBM groups to which the SDBM user belongs {IA.3::IA.1.14-R10-LDAP-3}.
- If the user supplied a digital certificate for a SASL external bind, LDAP maps the DN in the certificate to a RACF user ID {IA.3::IA.1.14-R10-LDAP-4}. Subsequent processing depends on the value of the LDAP sslMapCertificate configuration parameter.
    - If “check” is specified, the certificate DN becomes the LDBM bind DN, and LDAP derives LDBM groups solely from that DN {IA.3::IA.1.14-R10-LDAP-5}.
    - If “add” is specified, LDAP creates an SDBM DN from the mapped RACF user ID, and adds that SDBM DN as an alternate bind DN for authorization processing. LDBM processing derives LDBM groups from both the LDBM DN (in the certificate) and the SDBM DN {IA.3::IA.1.14-R10-LDAP-6}.
    - If “replace” is specified, LDAP creates an SDBM DN from the mapped RACF user ID, and that SDBM DN becomes the only bind DN for LDBM authorization processing. LDAP gathers the mapped user's RACF groups, converts them to SDBM DNs, and uses them as LDBM groups for authorization checking {IA.3::IA.1.14-R10-LDAP-7}.
    - Additionally, if configured to do so, LDAP also derives LDBM groups from the SDBM DN for the RACF user {IA.3::IA.1.14-R10-LDAP-8}.

Authentication-related differences between z/OS UNIX and typical non-z/OS UNIX systems

There are a few security aspects that are handled different in z/OS than in “standard” UNIX implementations. Those differences are:

1. Definition of users in /etc/passwd. In other UNIX systems, the file /etc/passwd contains the users defined and some of the user’s attributes. Within z/OS, the file /etc/passwd does not exist (or if it exists, does not contain any values used by the system). All user attributes are stored in the RACF user profile and managed solely by RACF {IA.4::IA.4.1}.
2. Handling of the su command: the handling of the su command depends on the existence of specific profiles in RACF.
3. Switching to a user identity by specifying a new user ID.
   The su command allows the change if the user provides the correct password (like most other UNIX systems) {IA.4::IA.4.2}, or if the original user ID has read access to

the BPX.SRV.newuser resource profile in the SURROGAT class {IA.4::IA.4.3}. Note that, unlike in most other UNIX systems, this also applies to subjects running with UID 0.

4. Switching to a superuser identity (UID 0) without specifying a new user ID. The su command allows the change if
   a) the user is already running with UID 0 {IA.4::IA.4.4}
   b) the original user ID has read access to the BPX.SUPERUSER resource profile in the FACILITY class {IA.4::IA.4.5}.

When a user executes a program that has the setuid bit set, only the effective user ID is changed to that of the owner of the file containing the program while the real user ID remains that of the caller {IA.4::IA.4.v111.1}. The RACF user ID is neither changed by the su command when changing to UID 0 using the su command without specifying a user ID {IA.4::IA.4.V1R7.1} nor by executing a program that has the setuid or setgid bit set {IA.4::IA.4.v111.2}.

A file or link found in a file system mounted as NOSECURITY is not considered trusted for this type of invocation {IA.4::IA.4.V2R1.1}

If a UNIX set-user-ID privileged program switches its UID to other than that of the set-user-ID program and it causes the real, effective and saved UID of the caller to be equal, then the set-user-ID privilege of the program is given up {IA.4::IA.4.V2R1.2}.

If a UNIX set-group-ID privileged program switches its GID to other than that of the set-group-ID program and it causes the real, effective and saved GID of the caller to all be equal, then the set-group-ID privilege of the program is given up {IA.4::IA.4.V2R1.3}.

When executing the su command to a user with a non-zero UID, or when specifying the userid and password with the su command when switching to a user with UID 0, all credentials including the RACF user ID are reset to the new user {IA.4::IA.4.V1R7.2}.

An executable file can have additional attributes (setuid and setgid bits) used to allow a program temporary access to files that are not normally accessible to other users. Those permission bits sets the effective user ID or group ID of the user process executing a program to that of the file whenever the file is run {IA.4::IA.4.V1R7.3}. The setuid and setgid bits are only honored for executable files containing load modules or REXX execs. These bits are not honored for shell scripts that reside in the file system {IA.4::IA.4.V1R7.4}.

When authorized to do so, a process executing in the z/OS UNIX System Services environment can change its real, effective, and saved set user IDs or the real, effective and saved user ID of process spawned off using dedicated system services. The following restrictions apply:

- The process is executing with UID 0 or the current subject has the trusted or privileged attribute {IA.4::IA.4.V1R7.5} or
- If User_ID is the same as the real UID of the process or the saved set UID, the setuid service sets the effective UID to be the same as User_ID {IA.4::IA.4.V1R7.6}.

The RACF user ID is changed if one of the following conditions is satisfied

- The calling process is executing with an effective UID 0, the calling user ID has been authorized to the BPX.DAEMON profile in the FACILITY class and the calling program has been loaded from a controlled library in a clean environment {IA.4::IA.4.V1R7.7}.
- The target user ID has been successfully authenticated by the password service {IA.4::IA.4.V1R7.8} or has SURROGAT authority to the new user ID {IA.4::IA.4.V1R7.9}. The TOE may also allow to change the real, effective, and saved set group IDs (GIDs) for the calling process. The following restrictions apply:
    - the process is executing with UID 0 or the current RACF user ID has the trusted or privileged attribute {IA.4::IA.4.V1R7.10} or
    - If Group_ID is equal to the real group ID or saved set group ID of the process, the effective group ID is set to Group_ID the process is executing with UID 0 or the current RACF user ID has the trusted or privileged attribute {IA.4::IA.4.V1R7.11}.

The setgid service does not change any supplementary group IDs of the calling process {IA.4::IA.4.V1R7.12}.

User identification and authentication are also performed by the telnet, rlogin, rsh, rexec, and ftp z/OS UNIX services (as described in Authentication function), the LDAP server, the HTTP Server, and the SSH daemon (sshd).

The sudo Command

With the Ported Tools for z/OS Supplementary Toolkit Feature installed, the security administrator can create the sudoers file, and through configuration statements in that file the administrator can allow selected users the ability to use the sudo command to run specified UNIX commands, with specified operands, under the authority of different user IDs. This provides a more restricted, controlled, version of the su command functionality mentioned earlier, because the administrator can limit both the UNIX shell commands and the command operands available, rather than allowing all UNIX shell commands as su normally would {IA.4::IA.4-R13-SUDO-1}

The BPX.DAEMON Profile in the FACILITY Class

When the BPX.DAEMON profile is defined in the FACILITY class of RACF, z/OS allows for a finer granularity of handling privileges of z/OS UNIX System Services.

Any superuser permitted to this profile has the daemon authority to change MVS identities via z/OS UNIX services without knowing the target user ID’s password {IA.4::IA.4.V1R7.13}. This identity change can only occur if the target user ID has an OMVS segment defined {IA.4::IA.4.V1R7.14}.

If the BPX.DAEMON FACILITY class profile is defined, then z/OS UNIX will verify that the address space has not loaded any executables that are uncontrolled before it allows any of the following services that are controlled by z/OS UNIX to succeed:

- seteuid
- setuid
- setreuid
- pthread_security_np()

- auth_check_resource_np()
- __login()
- spawn with user ID change
- __passwd()
- __certificate()

{IA.4::IA.4.V1R7.15}

Daemon authority is required only when a program does a setuid(), seteuid(), setreuid(), or spawn with user ID to change the current UID without first having issued an ___certificate() or an __passwd() call to the target user ID. In order to change the MVS identity without knowing the target user ID’s password, the caller of these services must be a superuser. Additionally, if a BPX.DAEMON FACILITY class profile is defined and the FACILITY class is active, the caller must be permitted to use this profile {IA.4::IA.4.V1R7.16}. If a program comes from a controlled library and knows the target UID’s password, or supplied the target’s certificate, it can change the UID without having daemon authority {IA.4::IA.4.V1R7.17}.

Assertion of User Identity

{IA.5::IA.5-R12-IDPROP-RACF-1} RACF supports specification on initACEE and RACROUTE REQUEST=VERIFY of a distributed identity via a structure called an IDID (containing a user's distinguished name (DN) and a domain/realm name (DC)):

- If an IDID is specified on initACEE but a RACF user ID is not specified, then initACEE will perform a mapping operation using the IDIDMAP class to determine the associated RACF user ID to use during RACROUTE REQUEST=VERIFY processing and will also include the IDID information.
- If both an IDID and a RACF user ID are specified on initACEE, then initACEE will create an ACEE for that user ID as it usually would and not perform mapping. Again, it will include the IDID information on the RACROUTE REQUEST=VERIFY call.
- When an IDID is specified on RACROUTE REQUEST=VERIFY, RACF uses the other parameters to create the ACEE as it normally does, but will anchor the IDID information in the ACEE for later use during auditing.

{IA.5::IA.5-R12-IDPROP-RACF-2} RACF provides a ‘RACMAP’ command to allow the security administrator to define ‘mapping filter rules’ to RACF that will support the mapping of distributed user identities, as specified within the IDID data area, into RACF user IDs as required by the customer. This new RACF command is similar to the existing RACDCERT command, which allows the specification of mapping filter rules that RACF uses to map distributed user identities based on the ‘subject’ and ‘issuer’ information within Digital Certificates. But instead of being limited to only user identities within Digital Certificates, the new command supports the definition of mapping filter rules within the IDIDMAP class based on an x.500 representation of the user identity and the ‘Name-Space’ that the user is defined within.

{IA.5::IA.5-R12-IDPROP-RACF-3} The RACF R_cacheserv callable service provides a function (function code 7) that will extract a copy of the ACEE for the currently active user in the form of a RACF environment object (aka RACO), save that RACO in a data space, and return a

Classification: Public
© Copyright IBM Corp. 2015

context reference (ICRX) that will uniquely identify that saved RACO. Subsequently an invoker of RACROUTE REQUEST=VERIFY can provide that ICRX and RACF will recreate the security environment (ACEE) of the original user from the RACO or from the IDID information in the ICRX if necessary. R_cacheserv will also allow deletion of a cached security environment.

{IA.5::IA.5-R12-IDPROP-RACF-5}The RACF R_cacheserv service can also return a pseudo-userID and pseudo-password that RACF authentication functions (initACEE, RACROUTE REQUEST=VERIFY) will subsequently accept and use to create an ACEE for the previously specified RACF user ID with an ICTX data area cached on the earlier R_cacheserv invocation. The pseudo-userID and pseudo-password may be used at most once on a subsequent authentication request.

{IA.5::IA.5-R12-IDPROP-RACF-4} RACF will provide an ENF 71 signal when an administrator has issued an ALTUSER REVOKE or a CONNECT or REMOVE command that changes a user's group connections, allowing applications that have cached ACEEs locally or via R_cacheserv to remove their cache entries and recreate the ACEEs if needed.

{IA.5::IA.5-V2R1-IDPROP-RACF-6} RACF will provide an ENF 71 signal when an administrator has issued a DELUSER or DELGRP command allowing applications that have cached ACEEs local or via R_cacheserv to remove cache entries.

{IA.5::IA.5-V2R1-IDPROP-RACF-7} RACF will provide an ENF 79 signal when an administrator has issued a PERMIT command that changes a user's or group's authorization to resources in a resource class that has been defined in the RACF Class Descriptor Table with the SIGNAL=YES option.

{IA.5::IA.5-R12-IDPROP-USS-1} The UNIX System Services __passwd (BPX1PWD) and pthread_security_np() (BPX1TLS) function allows appropriately authorized servers to assert a user identity and create a security environment by specification of the pseudo-userID and pseudo-password obtained via a prior authentication and use of R_cacheserv.

8.4.2 Access Control (FDP)

8.4.2.1 Discretionary Access Control

Access control principles

z/OS provides the Resource Access Control Facility (RACF) as the component that performs access control between subjects acting on behalf of a user and resources protected by the discretionary access control policies. RACF uses user and resource profiles it stores in the RACF database to decide if a subject has access to a non-UNIX resource. For UNIX resources, the access permissions are carried with the resource itself (permission bits)

All z/OS components that have to make access decisions will call RACF through a z/OS interface. The following figure shows the flow of requests and replies within z/OS when a request to access a protected resource is made.

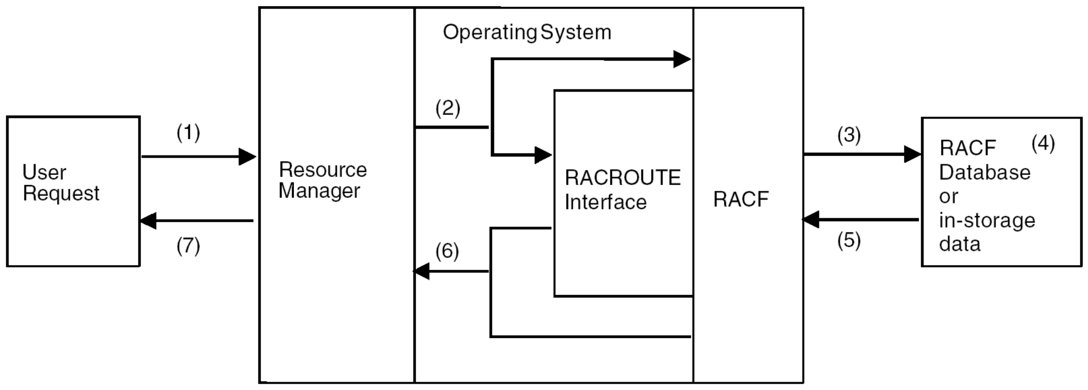


Figure 13 : RACF Request Flow

A program that wants to access a resource uses a function that is part of the external interface provided by the z/OS operating system to one of the z/OS components (1). An example is a program that wants to open a data set.

The z/OS component responsible for managing the resource calls the RACF component using the internal interface to RACF (mainly the RACROUTE interface) to check the access rights of the user that initiated the user request and passes the name and type of the resource and the requested type of access to RACF {AC.1::AC.1.1}. The caller may also pass the ID of the user or an explicit user security context (ACEE), or RACF obtains those values from the security context of the user that has been established during user authentication (2) {AC.1::AC.1.2}.

RACF extracts the user information from the security context of the user or (in a few cases) from the user profile, extracts the resource profile from its external database or the internal cache (3), and checks to see if the user with his current security attributes is allowed to access the resource in the requested access mode (4 and 5).

If the resource is known to RACF, RACF returns either a “yes” or a “no” decision for the access request {AC.1::AC.1.3}. If the resource is not known to RACF, RACF may return a “don’t know” return code unless there are specific options set that allow RACF to take a yes or no decision (6) {AC.1::AC.1.4}. In the case of a “don’t know” result, the resource manager needs to make its own decision whether to allow access or not. Depending on the decision, the resource manager will either perform or reject the access request of the user program (7) {AC.1::AC.1.5}.

The protection philosophy of RACF is based on “profiles” that represent protected resources but also users and groups. Profiles are organized in profile classes, where each class represents a type of resource (such as data sets or terminals) or other entity (such as users or groups). A profile stores attributes of the subject or object it represents.

For profiles that represent a protected resource, an access list can be assigned {AC.1::AC.1.6}. This access list specifies the type of access subjects may have to the resource represented by the profile.

RACF handles 6 different types of access which are hierarchically ordered. Those access types are (from low to high):

- NONE

- EXECUTE
- READ
- UPDATE
- CONTROL
- ALTER

Hierarchically ordered means that a higher access type implies also the lower access types.

To check access to a z/OS resource, a resource manager will call RACF specifying:

- the resource class
- the name of the resource
- a pointer to the user's ACEE (which represents the user)
- the requested type of access

If RACF knows the resource class and if this resource class is active, RACF will identify the resource profile protecting the resource in that class, extract the access control list for that resource profile and checks if the user has the requested type of access or a higher type of access. The exact details of this access check algorithm are defined later in this section.

The semantics of a specific type of access are defined by the resource manager. This allows RACF to used also for privilege management by defining a specific privilege as a specific type of access to a specific profile in a specific class which represents the privilege. A resource manager then can call RACF to check if a user has the required type of access to this profile and allow the user to perform a specific privileged function only if the user has that type of access. Quite a number of management activities defined in the Security Management section are implemented that way and the Security Management section describes the classes, profiles in the classes and the semantics z/OS assigns to specific access types in those classes that are used for specific management privileges.

Access control to UNIX file system objects and IPC objects are also handled by RACF, but in the case of these objects, the access rights are stored with the object itself. RACF still performs the access check. For details, see the description of access control for UNIX objects.

RACF also allows LDAP clients (typically servers outside of the TOE, residing on the network) that have authenticated using an ICTX-style DN to request RACF to perform an access check on its own behalf or on behalf of another user (typically a client of the server making the request). Note that these requests do not represent actual resource accesses that will occur within the TOE, but merely allow the TOE to provide access controls to processes running externally within the network if desired. Additionally, the client can specify any resource class known to RACF, except DATASET, and any resource name with legal RACF syntax that it chooses.

The LDAP client uses an LDAP extended-operation (which gets routed by the LDAP server to the ICTX plug-in) to request this remote authorization function which can:

- Check the client's own authority to access a specified resource name in a specified RACF resource class {AC.2::AC.2-R9-EIM-1}. This usage of the remote authorization service requires the LDAP client to have READ authority to FACILITY resource IRR.LDAP.REMOTE.AUTH {AC.2::AC.2-R9-EIM-2}.

- Check a specified user's or group's authority to access a specified resource name in a specified RACF resource class {AC.2::AC.2-R10-EIM-3}. This usage of the remote authorization service requires the LDAP client to have UPDATE authority to FACILITY resource IRR.LDAP.REMOTE.AUTH {AC.2::AC.2-R9-EIM-4}.

Through the remote crypto plug-in, the z/OS LDAP server provides client applications the ability to utilize PKCS#11 crypto provided through ICSF (Integrated Cryptographic Security Facility).

PKCS#11 is a Public Key Cryptographic Standard (PKCS) that defines a platform independent API for using cryptographic tokens. This standard defines the types of cryptographic tokens supported and how to create, delete and use (encrypt, decrypt, hash data) tokens in cryptographic operations.

The remote crypto plug-in supports the RemoteCryptoPKCS#11 extended operation which allows any LDAP client application authorized (successfully bound and authenticated) to perform any PKCS#11 API by invoking the appropriate ICSF callable service {AC.2::AC.2-V2R1-EIM-5}. ICSF access checking is enforced for those calls {AC.2::AC.2-R9-EIM-6}. The RemoteCryptoPKCS#11 extended operation is a generic extended operation that allows an LDAP client application to specify the same data as if invoking the ICSF callable service locally.

The remote crypto plug-in also supports the RemoteCryptoCCA extended operation which allows any LDAP client application authorized (successfully bound and authenticated) to perform any CCA callable service by invoking the appropriate ICSF callable service {AC.2::AC.2-R9-EIM-7}. ICSF access checking is enforced for those calls {AC.2::AC.2-R9-EIM-8}.

The RemoteCryptoCCA extended operation is a generic extended operation that allows an LDAP client application to specify the same data as if invoking the ICSF callable service locally.

Protected resources

The protected resources considered in detail in this Security Target are:

- Data sets
- Volumes
- Devices
- Terminals
- TCP/IP connections
- Operator commands
- Programs
- Consoles
- UNIX file system objects

- UNIX IPC objects
- LDAP LDBM objects
- System logger objects
- Communications Server Policy Agent data
- Hardware management interface functions

As a general-access control system, RACF is capable of protecting a number of other resources, but those are not considered in detail in this evaluation because it is up to the resource manager that uses them to determine the valid resource names and the semantics of the access control decisions. Instead, we will mention them below and later consider only the rules regarding their administration via the RACF commands.

RACF can also protect installation-defined resource classes, which we will not consider at all in this evaluation.

The reader should note that some other RACF classes are included in this evaluation that do not represent "resources" but represent privileges or restrictions, where assigning "access" to a resource in such a class to a user or a group just determines that the user or group has the privilege or restriction associated with the profile. Those classes and profiles are described in the relevant subsection of the access control section in this Security Target. The reader should also understand that granting privileges that are not described in this document should be done with care, and only for trusted users, as those privileges may allow administrative functions or extraordinary resource accesses.

Resource profiles of RACF are structured into an open set of "resource classes". IBM provides a set of resources classes used by z/OS (stored in the "static class descriptor table"), but RACF also allows for the definition and activation of additional resource classes using the RDEFINE or RALTER commands addressing the CDT general resource class (those are stored in the "dynamic class descriptor table"). The dynamically defined classes need to be "activated" using the command SETROPTS RACLIST(CDT) REFRESH. Resource classes represent "types" of objects that are access protected by RACF. IBM supplies a default static class descriptor table, which is structured into resources used by different components of z/OS as well as resources used by specific other IBM products like DB2 or CICS.

General z/OS Resource Classes

IBM supplies a class descriptor table that defines classes used by z/OS or other IBM products that use the services of RACF for controlling access to the resources they manage. The following tables list the classes of general resources used by RACF on z/OS and used by z/OS, DB2, and other products. Resource classes marked in red include resources that are used by RACF internally for managing access to RACF resources or use of RACF controlled privileges. Not all classes are used by z/OS, and some are used by products or z/OS elements that have not been evaluated under the Common Criteria. They are listed here, however, for completeness.

Class Name	Purpose
ALCSAUTH	Supports the Airline Control System/MVS (ALCS/MVS) product.
APPCLU	Verifying the identity of partner logical units during VTAM session establishment.

Class Name	Purpose
APPCPORT	Controlling which user IDs can access the system from a given LU (APPC port of entry). Also, conditional access to resources for users entering the system from a given LU.
APPCSERV	Controlling whether a program being run by a user can act as a server for a specific APPC transaction program (TP).
APPCSI	Controlling access to APPC side information files.
APPCTP	Controlling the use of APPC transaction programs.
APPL	Controlling access to applications.
CACHECLS	Contains profiles used for saving and restoring cache contents from the RACF database. See the description of the R_cacheserv RACF callable service.
CBIND	Controlling the client’s ability to bind to the server.
CDT	Contains profiles for installation-defined classes for the dynamic CDT.
CFIELD	Contains profiles that define the installation's custom fields.
CONSOLE	Controlling access to MCS consoles. Also, conditional access to other resources for commands originating from an MCS console.
DASDVOL	DASD volumes.
DBNFORM	Reserved for future IBM use.
DEVICES	Used by MVS allocation to control who can allocate devices such as: • Unit record devices (printers and punches) (allocated only by PSF, JES2, or JES3) • Graphics devices (allocated only by VTAM) Teleprocessing (TP) or communications devices (allocated only by VTAM)
DIGTCERT	Contains digital certificates and information related to them. See chapter 20 of [RACF.SAG] and the description of the RACDCERT command.
DIGTCRIT	Specifies additional criteria for certificate name filters. See chapter 20 of [RACF.SAG] and the description of the RACDCERT command.
DIGTNMAP	Mapping class for certificate name filters. See chapter 20 of [RACF.SAG] and the description of the RACDCERT command.
DIGTRING	Contains a profile for each key ring and provides information about the digital certificates that are part of each key ring. See chapter 20 of [RACF.SAG] and the description of the RACDCERT command.
DLFCLASS	The data look-aside facility.
FACILITY	Miscellaneous uses. Profiles are defined in this class so resource managers (typically elements of z/OS or z/VM) can check a user’s access to the profiles when the user takes some action. Examples are the profiles used to control execution of RACDCERT command functions and the profiles used to control privileges in the z/OS UNIX environment. RACF does not document all of the resources used in the FACILITY class by other products. For information on the FACILITY class resources used by a specific product (other than RACF itself), see that product’s

Class Name	Purpose
	documentation.
FIELD	Fields in RACF profiles (field-level access checking).
GDASDVOL	Resource group class for DASDVOL class.
GLOBAL	Global access checking table entry.
GMBR	Member class for the GLOBAL class.
GSDSF	Resource group class for SDSF class.
GTERMINL	Resource group class for TERMINAL class.
GXFACILI	Grouping class for XFACILIT resources.
IBMOPC	Controlling access to OPC/ESA subsystems.
IDIDMAP	Contains distributed identity filters created with the RACMAP command.
JESINPUT	Conditional access support for commands or jobs entered into the system through a JES input device.
JESJOBS	Controlling the submission and cancellation of jobs by job name. Controls the altering of jobs being processed using the Job Modify SSI (SSI 85). Controls a jobs access to a job class.
JESSPOOL	Controlling access to job data sets on the JES spool (that is, SYSIN and SYSOUT data sets).
KEYSMSTR	Contains profiles that hold keys to encrypt data stored in the RACF database, such as LDAP BIND passwords and DCE passwords.
LDAP	Controls authorization roles for LDAP administration.
LDAPBIND	Contains the LDAP server URL, bind distinguished name, and bind password.
LOGSTRM	Controls system logger resources, such as log streams and the coupling facility structures associated with log streams.
NODES	Controlling the following on MVS systems: 1. Whether jobs are allowed to enter the system from other nodes 2. Whether jobs that enter the system from other nodes have to pass user identification and password verification checks
NODMBR	Member class for the NODES class.
OPERCMDS	Controlling who can issue operator commands (for example, JES and MVS, and operator commands).
PKISERV	Contains profiles that provide granular administrative access controls for performing administrative actions for a given PKI Services instance and a given certificate or certificate request type
PMBR	Member class for the PROGRAM class.
PROGRAM	Protects executable programs.
PROPCNTL	Controlling if user ID propagation can occur, and if so, for which user IDs (such as the CICS or IMS main task user ID), user ID propagation is not to occur.
PSFMPL	Used by PSF to perform security functions for printing, such as separator page labeling, data page labeling, and enforcement of the user printable area.

Class Name	Purpose
PTKTDATA	PassTicket key class enables the security administrator to associate a RACF secured sign-on secret key with a particular mainframe application that uses RACF for user authentication. Examples of such applications are IMS, CICS, TSO, z/VM, APPC, and MVS batch.
RACFEVNT	Contains profiles that control the following events: 1. LDAP change log notification for changes to certain RACF profiles New password and password phrase enveloping for a given user.
RACFHC	Used by IBM Health Checker for z/OS. Contains profiles that list the resources to check for each installation-defined health check.
RACFVARS	RACF variables. In this class, profile names, which start with & (ampersand), act as RACF variables that can be specified in profile names in other RACF general resource classes. See [RACF.SAG], chapter 7.
RACGLIST	Class of profiles that hold the results of RACROUTE REQUEST=LIST,GLOBAL=YES or a SETROPTS RACLIST operation.
RACHCMBR	Used by IBM Health Checker for z/OS. Member class for the RACHCMBR class.
RDATALIB	Used to control use of the R_datalib callable service (IRRSDL00 or IRRSDL64).
RRSFDATA	Used to control RACF remote sharing facility (RRSF) functions.
RVARSMBR	Member class for the RACFVARS class.
SCDMBR	Member class for the SECDATA class.
SDSF	Controls the use of authorized commands in the System Display and Search Facility (SDSF). See also GSDSF class.
SERVAUTH	Contains profiles used by servers to check a client's authorization to use the server or to use resources managed by the server. Also, can be used to provide conditional access to resources for users entering the system from a given server.
SERVER	Controlling the server's ability to register with the daemon.
SMESSAGE	Controlling to which users a user can send messages (TSO only).
SOMDOBJS	Controlling the client's ability to invoke the method in the class.
STARTED	Used in preference to the started procedures table to assign an identity during the processing of an MVS START command.
SURROGAT	If surrogate submission is allowed, and if allowed, which user IDs can act as surrogates.
SYSMVIEW	Controlling access by the SystemView for MVS Launch Window to SystemView for MVS applications.
TAPEVOL	Tape volumes.
TEMPDSN	Controlling who can access residual temporary data sets.
TERMINAL	Terminals (TSO or z/VM). See also GTERMINL class.
VTAMAPPL	Controlling who can open ACBs from non-APF authorized programs.
WRITER	Controlling the use of JES writers.
XFACILIT	Miscellaneous uses. Profile names in this class can be longer than 39 characters in length. Profiles are defined in this class so that resource managers (typically elements of z/OS) can check a

Class Name	Purpose
	user’s access to the resources when the users take some action.

Table 15: General z/OS Resources Classes

DB2 Resource Classes

DB2 Resource classes are mentioned here to support the evaluation of DB2. Within this Security Target they are not used for any security function. The IBM supplied class descriptor table just contains those classes (as well as other classes related to applications like CICS, IMS, or Websphere), which are not used unless those applications are installed and configured to use RACF.

DSNADM	DB2 administrative authority class.
DSNR	Controls access to DB2 subsystems.
GDSNBP	Grouping class for DB2 buffer pool privileges.
GDSNCL	Grouping class for DB2 collection privileges.
GDSNDB	Grouping class for DB2 database privileges.
GDSNGV	Grouping class for DB2 global variables
GDSNJR	Grouping class for Java archive files (JARs).
GDSNPK	Grouping class for DB2 package privileges.
GDSNPN	Grouping class for DB2 plan privileges.
GDSNSC	Grouping class for DB2 schema privileges.
GDSNSG	Grouping class for DB2 storage group privileges.
GDSNSM	Grouping class for DB2 system privileges.
GDSNSP	Grouping class for DB2 stored procedure privileges.
GDSNSQ	Grouping class for DB2 sequences.
GDSNTB	Grouping class for DB2 table, index, or view privileges.
GDSNTS	Grouping class for DB2 tablespace privileges.
GDSNUF	Grouping class for DB2 user-defined function privileges.
GDSNUT	Grouping class for DB2 user-defined distinct type privileges.
MDSNBP	Member class for DB2 buffer pool privileges.
MDSNCL	Member class for DB2 collection privileges.
MDSNDB	Member class for DB2 database privileges.
MDSNGV	Member class for DB2 global variables
MDSNJR	Member class for Java archive files (JARs).
MDSNPK	Member class for DB2 package privileges.
MDSNPN	Member class for DB2 plan privileges.
MDSNSC	Member class for DB2 schema privileges.
MDSNSG	Member class for DB2 storage group privileges.
MDSNSM	Member class for DB2 system privileges.
MDSNSP	Member class for DB2 stored procedure privileges.
MDSNSQ	Member class for DB2 sequences.
MDSNTB	Member class for DB2 table, index, or view privileges.
MDSNTS	Member class for DB2 tablespace privileges.
MDSNUF	Member class for DB2 user-defined function privileges.
MDSNUT	Member class for DB2 user-defined distinct type privileges.

Table 16: DB2 Resources Classes

Enterprise Identity Mapping (EIM) Classes

RAUDITX	Controls auditing for Enterprise Identity Mapping (EIM).

Table 17:EIM Resources Classes

ICSF Resource Classes

CRYPTOZ	Controls access to PKCS #11 tokens.
CSFKEYS	Controls access to ICSF cryptographic keys.
CSFSERV	Controls access to ICSF cryptographic services.
GCSFKEYS	Resource group class for the CSFKEYS class.
GXCSFKEY	Resource group class for the XCSFKEY class.
XCSFKEY	Controls the exportation of ICSF cryptographic keys.

Table 18: ICSF Resources Classes

PSF Resource Classes

PRINTSRV	Controls access to printer definitions for Infoprint Server.

Table 19: PSF Resources Classes

Kerberos Resource Classes

KERBLINK	Mapping class for user identities of local and foreign principals.
REALM	Used to define the local and foreign realms.

Table 20: Kerberos Resources Classes

DFSMS Resource Classes

MGMTCLAS	SMS management classes.
STORCLAS	SMS storage classes.
SUBSYSNM	Authorizes a subsystem (such as a particular instance of CICS) to open a VSAM ACB and use VSAM record level sharing (RLS) functions.

Table 21: DFSMS Resources Classes

TSO Resource Classes

ACCTNUM	TSO account numbers.
PERFGRP	TSO performance groups.
TSOAUTH	TSO user authorities such as OPER and MOUNT.
TSOPROC	TSO logon procedures.

Table 22: TSO Resources Classes

UNIX System Services Resource Classes

DIRACC	Controls auditing (using SETROPTS LOGOPTIONS) for access checks for read/write access to z/OS UNIX directories. This class need not be active to control auditing.
DIRSRCH	Controls auditing (using SETROPTS LOGOPTIONS) of z/OS UNIX directory searches. This class need not be active to control auditing.
FSACCESS	Allows control of access to the root of a zFS file system through RACF resource profiles rather than UNIX permission bits or ACLs.
FSOBJ	Controls auditing (using SETROPTS LOGOPTIONS) for all access checks for z/OS UNIX file system objects except directory searches. Controls auditing (using SETROPTS AUDIT) of creation and deletion of z/OS UNIX file system objects. This class need not be active to control auditing.
FSSEC	Controls auditing (using SETROPTS LOGOPTIONS) for changes to the security data (FSP) for z/OS UNIX file system objects. This class need not be active to control auditing. When this class is active, it also controls whether ACLs are used during authorization checks to z/OS UNIX files and directories.
IPCOBJ	Controls auditing (using SETROPTS LOGOPTIONS) of access checks for inter-process communication (IPC) objects and changes to security information of IPC objects. Controls auditing (using SETROPTS AUDIT) of the creation and deletion of IPC objects. This class need not be active to control auditing.
PROCACT	Controls auditing (using SETROPTS LOGOPTIONS) of functions that look at data from, or affect the processing of, z/OS UNIX processes. This class need not be active to control auditing.
PROCESS	Controls auditing (using SETROPTS LOGOPTIONS) of changes to UIDs and GIDs of z/OS UNIX processes. Controls auditing (using SETROPTS AUDIT) of dubbing and undubbing of z/OS UNIX processes. This class need not be active to control auditing.
UNIXMAP	Contains profiles that are used to map z/OS UNIX UIDs to RACF user IDs and z/OS UNIX GIDs to RACF group names.
UNIXPRIV	Contains profiles that are used to grant z/OS UNIX privileges.

Table 23: USS Resources Classes

Data sets

Standard data set naming conventions

By default, RACF expects a data set name (and the data set profile name) to consist of at least two qualifiers. RACF also expects the high-level qualifier of the data set profile name to be either a RACF-defined user ID or a RACF-defined group name.

If an installation has chosen to define data set profiles under the standard RACF naming conventions, they can create a group for each high-level qualifier that is not a user ID, and permit users to protect any data set that has that high-level qualifier by giving them CREATE authority in that group {AC.2::AC.2.1}.

Table-driven data set naming conventions

An installation can use the naming convention table to set up and enforce a data set naming convention other than that used by RACF {AC.2::AC.2.2}. The table can:

- Supply a qualifier to be used as the high-level qualifier for authorization checking {AC.2::AC.2.3}
- Convert data set names to RACF naming convention form for RACF use {AC.2::AC.2.4}
- Convert names in RACF form to the installation’s format for external display {AC.2::AC.2.5}
- Enforce a naming convention by not allowing the definition of data sets that do not conform to an installation’s rules {AC.2::AC.2.6}
- Reduce RACF overhead by determining whether a data set is a user or group data set

An installation can create a naming convention table (module ICHNCV00), which RACF uses to check and modify (internally to RACF) the data set name in all commands and macros that process data set names {AC.2::AC.2.7}. An installation can use the table to selectively rearrange data set names to “fit” the RACF convention without actually changing those names.

Protecting data sets that have single-qualifier data set names

If some of the data sets in an installation have names that consist of a single qualifier, one can still RACF-protect those data sets {AC.2::AC.2.8}. To get RACF protection for single-qualifier names, the SETROPTS command with the PREFIX operand must be issued.

This command defines a high-level qualifier to be used as a prefix for single-qualifier names and activates the facility {AC.2::AC.2.9}. Then, when RACF processes requests for the data set, RACF internally modifies single-qualifier names by adding the prefix, making the data set names acceptable to RACF routines {AC.2::AC.2.10}. All SMF log records and all messages from RACF contain the RACF-modified version of the data set name {AC.2::AC.2.11} unless the SETROPTS REALDSN option is in effect {AC.2::AC.2-R10-RACF-1}.

Protecting user data sets

A user data set is a data set whose high-level qualifier is a RACF user ID. The following rules apply to user data sets:

- In general, all RACF-defined users can protect their own data sets {AC.2::AC.2.12}
- A user can RACF-protect a data set for another user under any of the following conditions:
    - The user who is protecting the data set has the SPECIAL attribute. A discrete or generic profile can be created {AC.2::AC.2.13}.
    - The user who is protecting the data set has the group-SPECIAL attribute, and the high-level-qualifier of the data set name is a user within the group-SPECIAL user’s scope of authority. A discrete or generic profile can be created {AC.2::AC.2.14}.

- The user who is protecting a data set has the OPERATIONS attribute (or the group-OPERATIONS attribute if the data set is within his scope of authority) and is simultaneously creating the data set {AC.2::AC.2.15}.

In this case, the user can create a discrete profile:

- Through ADSP {AC.2::AC.2.16}
- By specifying the PROTECT operand on the TSO ALLOCATE command that creates the data set {AC.2::AC.2.17}
- By specifying the PROTECT=YES OR SECMODEL= profile-name operands on the JCL DD statement that creates the data set {AC.2::AC.2.18}

Protecting group data sets

A group data set is a data set whose high-level qualifier is a RACF group name. A RACF-defined user can RACF-protect a group data set under any of the following conditions:

- The user has JOIN, CONNECT, or CREATE authority in the group {AC.2::AC.2.19};
- The user has the SPECIAL attribute (or the group-SPECIAL attribute for that group) and the request is made using the ADDSD command {AC.2::AC.2.20};
- The user has the OPERATIONS attribute and is not connected to the group {AC.2::AC.2.21}.

Controlling the creation of new data sets

Using data set profiles, an administrator can control whether users can create (allocate) new data sets.

For cataloged data sets, creating, deleting, or renaming the data set involves access not only to the data set profile protecting the data set, but also to the catalog in which the data set is cataloged {AC.2::AC.2.22}. In general, users need the following:

- To add entries to the catalog, users need authority to create the data set as specified below and (except for SMS-managed data sets) UPDATE authority to the catalog {AC.2::AC.2.23}.
- To delete entries from the catalog, users need ALTER authority to the protecting profile or to the catalog {AC.2::AC.2.24}.

The following cases describe how RACF can be used to control the creation of new user and group data sets.

A user can create a new user data set in the following situations:

- The data set is covered by an existing generic profile and the user does not have ADSP {AC.2::AC.2.25}. The creation is allowed if (1) the user has ALTER authority to the data set through a generic profile or global access checking, or (2) the data set is the user’s own data set {AC.2::AC.2.26}.
- The data set name is not covered by an existing generic profile and the user does not have ADSP and the data set is covered by the Global Access check table granting ALTER. {AC.2::AC.2.27}

- The user has ADSP and the data set is the user's own data set. The creation is allowed and RACF creates a discrete profile for the data set {AC.2::AC.2.28}.
- The user has the OPERATIONS attribute. If the user has the group-OPERATIONS attribute instead of OPERATIONS, the high-level qualifier of the new data set must be the ID of a user who is within the scope of that group {AC.2::AC.2.29-R12-RACF}.

A user can create a new group data set in the following situations:

- The data set name is protected by an existing generic profile and the user does not have ADSP. The creation is allowed if at least one of the following is true:
    - The user has ALTER authority to the data set through the generic profile or global access checking {AC.2::AC.2.30}
    - The user has CREATE authority in the group {AC.2::AC.2.31}
- The data set name is not covered by an existing generic profile and the user does not have ADSP {AC.2::AC.2.32}
- The user has ADSP and the data set belongs to a group of which the user is a member. The creation is allowed only if the user has CREATE authority in the group. If the creation is allowed, RACF creates a discrete profile for the data set {AC.2::AC.2.33}
- {AC.2::AC.2.36-R12-RACF}The user has the OPERATIONS attribute , or the group-OPERATIONS attribute for the group in question (directly or via a superior group), except when both of the following are true: The user is connected to the group with less than CREATE authority {AC.2::AC.2.34-R12-RACF}, and the user has less than ALTER access to the data set if it protected by a generic profile {AC.2::AC.2.35-R12-RACF}

Data set profile ownership

Each data set profile defined to RACF requires a RACF-defined user or group as the owner of the profile. The owner (if a user) has full control over the profile, including the access list {AC.2::AC.2.37}.

If the owner of the data set profile is a group, users with group-SPECIAL in that group have full control over the profile {AC.2::AC.2.38}.

Ownership of data set profiles is assigned when the profiles are defined to RACF but may be changed later. Note that ownership of a data set profile does not mean that the owner can automatically access that data set. To access a data set, the owner must still be authorized by the DAC policy rules {AC.2::AC.2.39}.

Volumes

By defining profiles in the DASDVOL class, the system administrator can define non-SMS-managed DASD volumes to RACF and authorize users to perform maintenance operations (such as dump, restore, scratch, and rename) without having access to the data set profiles protecting the data sets on the volume {AC.2::AC.2.40}. If a user does not have the necessary DASDVOL authority to a non-SMS-managed volume, he or she must have the necessary authority in the DATASET class to each of the data sets on the volume {AC.2::AC.2.41}.

Tape volumes are protected by profiles in the TAPEVOL class in the following circumstances:

- when the RACF TAPEVOL class is active and the IEHINITT utility is used to reinitialize a tape volume that contains a standard label {AC.2::AC.2.42-R8-1}
- when the RACF TAPEVOL class is active, and SETR NOTAPEDSN is in effect, and TAPEAUTHDSN=NO is specified in SYS1.PARMLIB(DEVSUPxx), and the tape contains standard labels, and a user accesses data on the tape {AC.2::AC.2.42-R8-2}.

Special Considerations for Data on Tape

A Data file located on tape can be protected in several different ways, depending on RACF and system options:

a) TAPEVOL class active, and SETROPTS NOTAPEDSN, and TAPEAUTHDSN=NO in SYS1.PARMLIB(DEVSUPxx): In this mode the data is protected by the TAPEVOL profile for the standard-labeled tape {AC.2::AC.2-R8-TAPE-1} or is unprotected if no profile exists or the tape has no labels {AC.2::AC.2-R8-Tape-2}.

b) TAPEVOL class inactive, and SETROPTS TAPEDSN, and TAPEAUTHDSN=NO in SYS1.PARMLIB(DEVSUPxx): In this mode the data is protected by the DATASET profile for the data set if the tape has standard labels or is unprotected if the tape has no labels {AC.2::AC.2-R8-TAPE-3}. However, in this mode, protection may be ineffective for data sets with names longer than 17 characters, and the physical tape volume labels record only the last 17 characters of a data set name. Therefore, this mode should be used only if an active tape management system (DFSMSrmm for the evaluated configuration) is keeping track of tape contents, and will reject the tape volume request if the data set name does not match the name specified by the user {AC.2::AC.2-R8-TAPE-4}.

c) TAPEVOL class active, and SETROPTS TAPEDSN, and TAPEAUTHDSN=NO in SYS1.PARMLIB(DEVSUPxx), and with TAPEVOL profiles that contain RACF TVTOCs: In this mode RACF verifies that the user has specified the correct data set name, and then security for the data set is provided by the DATASET profile for the data set, if the tape has standard labels {AC.2::AC.2-R8-TAPE-5}.

d) TAPEAUTHDSN=YES specified in SYS1.PARMLIB(DEVSUPxx): In this mode the system will check access based on the data set name specified by the user, regardless of the SETROPTS tape-related options in effect {AC.2::AC.2-R8-TAPE-6}.

e) TAPEAUTHF1=YES specified in SYS1.PARMLIB(DEVSUPxx) and either SETROPTS TAPEDSN specified or TAPEAUTHDSN=YES specified in SYS1.PARMLIB(DEVSUPxx): In this mode, in addition to the access check for the data set name specified by the user, the system will perform an additional check for the first data set on the tape {AC.2::AC.2-R8-TAPE-7}. Note: This mode requires an active tape management system (DFSMSrmm for the evaluated configuration) which provides the data set name for the first file on the tape.

Devices

A user authorized to define profiles in the DEVICES class can use this class to control which users can allocate unit record devices, teleprocessing or communications devices, and graphics devices {AC.2::AC.2.43}. For example, the DEVICES class can be used to ensure that only authorized users can allocate devices by name. The DEVICES class can not be used to protect other kinds of devices, such as tape or DASD devices.

Terminals

Terminals are protected by profiles in the TERMINAL or GTERMINL class. A user must have at least read access authority assigned to a profile representing a terminal to be able to use the terminal {AC.2::AC.2.45}. The GTERMINL class is provided to protect a class of terminals in the same way without the need to define discrete profiles for each terminal in the TERMINAL class {AC.2::AC.2.46}. User access to terminals that are not protected by a profile in one of those classes is defined by the parameter in the TERMINAL operand in the SETROPTS command {AC.2::AC.2.47}. If this parameter is NONE, a user can not use such terminals to log in {AC.2::AC.2.48}. If the parameter is READ, a user can use those terminals to log in {AC.2::AC.2.49}.

Access to terminals can also be controlled for groups of users. If the option NOTERMUACC is defined in the group profile, users within this group can only use terminals to which they are specifically authorized on the access list in the TERMINAL profile protecting the terminal {AC.2::AC.2.50}.

The use of a terminal can also be restricted to specific days and a time period within those days using the WHEN and TIME options in the RDEFINE and RALTER command {AC.2::AC.2.51}.

TCP/IP connections

TCP/IP is a component of the Communications Server subsystem of the TOE. TCP/IP runs as a started task and provides the TCP, UDP, RAW, ICMP and IP functions. TCP/IP loads an INET Physical File System into the UNIX System Services kernel to handle socket requests. TCP/IP connects to the VTAM® component of the Communications Server subsystem of the TOE for physical communications device management services. Up to eight instances of the TCP/IP started task may be run concurrently on one instance of the TOE to isolate networks or stacks. Socket applications may be directed to a particular stack or may transparently span multiple stacks.

Several TCP/IP resources can be protected by resources in the SERVAUTH class:

- Access to a particular TCP/IP stack is controlled when an application opens a socket by read access to a profile in the form “EZB.STACKACCESS.system-name.stack-name” where system-name is the name of the TOE image and stack-name is the job name of the particular stack {AC.2::AC.2.53}.
- Access to a particular IP address is controlled when an application explicitly binds a socket to a local address and when an application sends data to or receives data from a peer address. IP addresses are configured into named security zones within the stack using NETACCESS profile statements. Access to a particular security zone is controlled by read access to a profile in the form “EZB.NETACCESS. system-name.stack-name.SAF-resname” where system-name is the name of the TOE image, stack-name is the job name of the particular stack and SAF-resname is the name configured on the NetAccess statement {AC.2::AC.2.54}.
- Access to the intranode management network is controlled when an application attempts to start a TCP connection, or read or write any data, that traverses the intranode management network using an OSA-Express chpid of type OSM. Access to the intranode management network is controlled by read access to a profile in the form “EZB.OSM.system-name.stack-name” where system-name is the name of the

TOE image, and stack-name is the job name of the particular stack. {AC.2::AC.2-R12-CS-62}

TCP/IP makes point of access information available on sockets for use when processing user login requests. This information may be requested by applications. The UNIX Systems Services subsystem will request this information on behalf of an application when it invokes the __poe() service. The information provided by TCP/IP includes {AC.2::AC.2.56}:

- The fully-qualified SERVAUTH resource name of the NETACCESS security zone containing the peer IP address, if it is in a security zone.
- The TERMINAL resource name of the peer IP address, if it is an IPv4 address.
- Access to a particular port is controlled when an application explicitly binds a socket to a local port. Applications binding to low ports (below 1024) must be a UNIX superuser or APF-authorized. Port usage may also be controlled by configuring the Port or Portrange statement in the TCP/IP profile. Control may be by user ID, job name, or read access to a profile in the form "EZB.PORTACCESS.system-name.stack-name.SAF-resname", where system-name is the name of the TOE image, stack-name is the job name of the particular stack, and SAF-resname is the name configured on the Port or Portrange statement {AC.2::AC.2.55}. The port access functions will work for both reserved and (if configured via PORT UNRSV) for unreserved ports {AC.2::AC.2-R10-CS-PORT-1}.
- {AC.2::AC.2-R13-CS-DVIPA-1} If a bind call uses an IP address which is defined in a VIPARANGE subnet, READ access to EZB.BINDDVIPARANGE.system-name.stack-name and EZB.BINDDVIPARANGE.system-name.stack-name.SAF-name is checked.

  If an ioctl call uses an IP address which is defined in a VIPARANGE subnet, READ access to EZB.MODDVIPA.system-name.stack-name and EZB.MODDVIPA.system-name.stack-name.SAF-name is checked.

Operator commands

Operator commands can be protected by resources in the OPERCMDS class. Resources in this class are the individual commands specified in the form "subsystem-name.command-name" where subsystem-name is the name of the processing environment of the command (JES2, RACF, or MVS, for example). Access to an operator command protected by a RACF profile requires the appropriate access authority in the access control list of the profile for the command {AC.2::AC.2.64}. Note that if the class is active and a command is not protected by a profile it is not allowed to be executed.

Programs

The ability of users to execute programs can be restricted by the RACF program control function. This feature is useful for programs operating with privileges like authorized programs. Program control can for example be used to restrict the ability of a user to start an authorized program from an authorized library in a way such that it executes with APF authorization {AC.2::AC.2-V1R7-1}. Users may still have read access to the library and may therefore copy the program into another library and execute it from this library. Although this is possible, the program will then not execute with the privileges it has when executed from the original library {AC.2::AC.2-V1R7.2}.

Program control (as described in this section) applies to programs residing in z/OS partitioned data sets or libraries, not to programs stored as part of z/OS UNIX file system. Mechanisms for program control for the z/OS UNIX subsystem are explained in another section of this Security Target.

z/OS allows for three modes for program control: BASIC, ENHANCED and ENHANCED-WARNING. The mode is defined by the strings 'BASIC', 'ENHANCED' or 'ENHANCED-WARNING' in the APPLDATA field of the IRR.PGMSECURITY profile in the FACILITY class {AC.2::AC.2.V1R7.3}. An empty value or any other value than 'BASIC' or 'ENHANCED' will result in the ENHANCED-WARNING mode {AC.2::AC.2.V1R7.4}. If the IRR.PGMSECURITY profile is not defined, BASIC mode is used {AC.2::AC.2.V1R7.5}. In ENHANCED-WARNING mode the access decisions made by the TOE are the same as in BASIC mode but a warning message is issued whenever the access would have been denied in ENHANCED mode {AC.2::AC.2.V1R7.6}.

The checks that RACF makes when a user makes a request to load (execute) a program are:

i. If program control has been activated with SETROPTS WHEN(PROGRAM) {AC.2::AC.2-V1R7.7}

ii. If program control is active, RACF checks to see whether the program is protected by a profile in the PROGRAM class {AC.2::AC.2-V1R7.8}

iii. If the program is not protected, RACF determines whether there are any data sets currently open using PADS or whether there are any execute-controlled programs in storage in the address space:

   - If there are no such data sets or programs, RACF marks the environment dirty (uncontrolled) and allows the user to execute the program {AC.2::AC.2-V1R7.9}.
   - If there are data sets currently opened using PADS, or programs to which the user has only EXECUTE authority, RACF fails the request and the system abends the task. RACF issues message ICH423I to document the execute-controlled programs, or message ICH424I to document the PADS data sets that caused the operation to fail. In this way, RACF prevents uncontrolled programs from gaining access to protected data or programs inappropriately {AC.2::AC.2-V1R7.10}.

iv. If the program is protected by a profile but the user does not have at least EXECUTE authority to the program, RACF causes the system to abend the task because the user is not authorized to execute the program {AC.2::AC.2-V1R7.11}.

v. If the program is protected by a profile and the user has only EXECUTE authority to the PROGRAM profile or to the library that contains the program (when the program is loaded from a JOBLIB, STEPLIB, or tasklib), and if the job step or TSO session is running in ENHANCED program security mode, RACF checks whether an appropriate program established the program environment. RACF determines if the first program executed in the job step had the ’MAIN’ attribute, or (if necessary) if the program invoked by TSOEXEC or IKJEFTSR had the ’MAIN’ attribute. If the program does not have MAIN, RACF next determines if the first program run in the current task (TCB) or the first program executed in some parent task had the ’BASIC’ attribute. If so, RACF allows the Program control request. Otherwise, RACF fails the request and issues message ICH429I to describe the problem and tell you what program established the environment {AC.2::AC.2-V1R7.12}.

vi. If the user is still authorized to execute the program and the program was defined with the PADCHK attribute, RACF checks whether any program-accessed data sets are open.

- If no program-accessed data sets are open, RACF allows the user to execute the program {AC.2::AC.2-V1R7.13}.
- If program-accessed data sets are open, RACF checks the user or program combination to verify that the combination has at least the same authority to each data set in the list that was required when each data set was opened.
- If the user or program combination has sufficient authority to all of the opened data sets, RACF allows the user to execute the program {AC.2::AC.2-V1R7.14}
- If the user or program combination does not have sufficient authority to all of the opened data sets, RACF causes the system to end the task (with abend code 306 or 806) {AC.2::AC.2-V1R7.15}.

With program control enabled, z/OS provides the ability to allow users to access data sets which they are not allowed to access directly by using program controlled programs {AC.2::AC.2.V1R7.16}.

The following algorithm is used to determine if a user has access to a data set via a controlled program:

Whenever the user has the requested access to the data set as determined by normal RACF access checking, access is granted {AC.2::AC.2.V1R7.17}.

If the user is not granted access to the data set with normal authorization checking, RACF checks the data set's conditional access list if program control is active and the program currently executing is executing as a RACF-controlled program in a clean environment. RACF authorizes the user to open the program-accessed data set with the currently executing program if all of the following conditions are met:

- The conditional access list contains the name of the currently running program, the name of the first program currently running in the current task (TCB), or the name of the first program currently running in a parent task, with the requested level of access or higher {AC.2::AC.2.V1R7.18}.
- The user's group or user ID is associated with the program name in the conditional access list {AC.2::AC.2.V1R7.19}.
- The current program environment (job step, or task established under TSO/E using TSOEXEC or IKJEFTSR) is controlled. In other words, it has not loaded an uncontrolled program. If either of these conditions are not met, the environment is considered uncontrolled. The user's attempt to open the program-accessed data set fails and the task ends with abend code 913. RACF issues message ICH417I, specifying what caused the environment to become uncontrolled {AC.2::AC.2.V1R7.20}.
- If the job step or TSO session is running in ENHANCED program security mode, one of the following is true:

    - The current environment (job step or task created by TSOEXEC or IKJEFTSR) first ran a program defined with the 'MAIN' attribute.
    - The current program running in the current task, or the first program run in the current task or a parent task, has the BASIC attribute. If neither of these conditions is met, the user's attempt to open the program-accessed data set fails and the task ends with abend code 913. RACF issues message ICH426I, specifying the non-MAIN program that established the current environment {AC.2::AC.2.V1R7.21}.
- If there is more than one controlled program running in the current environment (job step or task created by TSOEXEC or IKJEFTSR), all of those programs defined with the PADCHK attribute have conditional access list entries allowing them to access the data set. If one or more programs in the environment are not authorized, the attempt fails and the task terminates with abend code 913. RACF issues message ICH418I specifying one or more programs that were missing from the conditional access list {AC.2::AC.2.V1R7.22}.
- If all the conditions for program access to data set are met and the requested type of access is granted to the program by the profile protecting the data set, access is granted {AC.2::AC.2.V1R7.23}.

Consoles

When the CONSOLE class is active and a console being used is protected by a profile in the CONSOLE class, RACF ensures that the person attempting to logon at this console has the proper authority to do so {AC.2::AC2.V1R7.24}. Using RACF, the use of system consoles can be controlled {AC.2::AC2.V1R7.25}.

UNIX file system objects

UNIX file system objects in the HFS or zFS file system have their access control defined by:

- UNIX permission bits
- Access control list entries

All of those access-control-related attributes of file system objects are stored with the object. Access control lists are stored and managed as extended attributes of the file system object and are not stored in the RACF database {AC.2::AC.2.65}. RACF is still involved when an access decision is made to a UNIX file system object {AC.2::AC.2.66}. The UNIX System Services subsystem of the TOE extracts the permission bits, access control list entries from the file system object as well as the effective user ID and passes this information to RACF using the ck_access RACF callable service. RACF then evaluates this information, extracts other information relevant for the access decision from the RACF database, performs the auditing in accordance with the audit policy defined by the system administrator and returns the access decision to the calling UNIX System Services subsystem of the TOE {AC.2::AC.2.67}.

Besides the access control lists, additional privileges and restrictions may be defined to allow a finer granularity. Those privileges and restrictions are defined as profiles in the UNIXPRIV class and users can be granted those privileges or restrictions by giving them authority to those profiles. The ones that are considered in this Security Target are:

- SUPERUSER.FILESYS.ACL.ACLOVERRIDE

When this profile is defined and active in RACF, a user who has been given authority to this profile is able to override the access control defined by the access control lists for z/OS UNIX file system objects.

In z/OS, a UNIX superuser can access all z/OS UNIX files, but is still bound by his rights defined in RACF with respect to z/OS data sets and other resources {AC.2::AC.2.68}.

z/OS UNIX IPC objects

z/OS UNIX IPC objects are subject to discretionary access control. The permission bits associated with the IPC object define the discretionary access to those objects. The permission bits are determined by the creator of the IPC object and are saved in-memory by the UNIX Kernel. UNIX System Services will collect the permission bits from the IPC object and call RACF using the ck_IPC_access RACF callable service. RACF will then determine if the user can be granted the requested type of access and returns the decision to UNIX System Services. For security claims see DAC for UNIX objects.

LDAP LDBM objects

LDAP LDBM objects (objects in an LDBM back end for a z/OS LDAP server) exist in a single administrator-configured file (LDBM database) in the UNIX file system for each suffix the LDBM back end supports in each server {AC.2::AC.2-R8-LDAP-1} and are subject to discretionary access control by the LDAP server itself (not by RACF) using standard LDAP ACLs {AC.2::AC.2-R8-LDAP-2}and/or LDAP Filter ACLs {AC.2::AC.2-R12-LDAP-13}. LDAP objects are organized hierarchically in a tree format, and each object has a distinguished name (DN) which both names the object and locates it within the tree {AC.2::AC.2-R8-LDAP-3}.

[Note: LDAP also supports a CDBM back end, containing configuration information for the LDAP server. The CDBM back end looks and operates like the LDBM back end, except that it contains a directory information tree that holds LDAP configuration information and specifies the members of the LDAP administrative group and optionally their LDAP administrative roles. Members of the LDAP administrative group have specific access rights based on their LDAP administrative roles to the configuration data in the directory information tree, as described later in the LDAP Management section. Statements in this document about how LDBM works apply to CDBM, too, except where otherwise indicated.]

Users do not have direct access to the data (in the sense that they have for, say, data access via FTP or NFS). Rather, users make requests to the LDAP server specifying the named objects to retrieve, and the server interprets those requests, locates the named objects, and acts on them if the user has the proper authority {AC.2::AC.2-R8-LDAP-4}.

Users who have not performed a bind or have performed an anonymous bind are called unauthenticated or anonymous. There is no difference between the access rights given to unauthenticated and anonymous user {AC.2::AC.2-R8-LDAP-6}. Administrators may allow access to anonymous users {AC.2::AC.2-R8-LDAP-7} or deny access to anonymous users {AC.2::AC.2-R8-LDAP-8} anywhere they choose within the LDAP tree {AC.2::AC.2-R8-LDAP-9}. By default anonymous access is allowed {AC.2::AC.2-R8-LDAP-10}.

For further information see Algorithm to check for DAC access to LDAP LDBM objects.

System Logger objects

System logger resources, such as log streams and the coupling facility structures associated with them are subject to discretionary access control. For more information about those objects and RACF profiles used to protect them, see the section on the management of system logger objects in the management section

Communications Server Policy objects

Communications Server Policy objects can be read by users that have at least read access to the profiles protecting those objects. For more information about those objects and the RACF profiles that protect them see the section on the Communications Server Policy Agent later in this document.

Hardware Management Interface Functions

The System z processors provide privileged instructions that allow an operating system running in a logical partition (LPAR) to perform hardware management functions (e.g., activating or deactivating processors, IPLing an operating system into an LPAR, adjusting LPAR performance characteristics, adjusting LPAR definitions, etc.) When defining an LPAR to PR/SM an administrator specifies whether the LPAR can control other LPARs by setting the Cross-Partition Authority Flag in the new LPAR definition. LPARs without that flag set can not control other LPARs {AC.2::AC.2-R11-BCPii-1}.

DAC for MVS resources

RACF controls the types of access to all MVS (non-UNIX, non-LDAP) resources. The access types are ordered hierarchically, an access type listed higher in the list implies all the access types lower in this list (except for NONE access). The full semantics of each access type are defined by the resource manager. The semantics for MVS data sets are:

- ALTER

ALTER allows users to read, update, delete, rename, move, or scratch the data set.

When specified in a discrete profile, ALTER allows users to read, alter, and delete the profile itself including the access list {AC.4::AC.4.1}.

ALTER does not allow users to change the owner of the profile using the ALTDSD command {AC.4::AC.4.2}. However, if a user with ALTER access authority to a discrete data set profile renames the data set, changing the high-level qualifier to his or her own user ID, both the data set and the profile are renamed, and the OWNER of the profile is changed to the new user ID {AC.4::AC.4.3}.

When specified in a generic profile, ALTER gives users no authority over the profile itself {AC.4::AC.4.4}.

- CONTROL

For VSAM data sets, CONTROL is equivalent to the VSAM CONTROL password; that is, it allows users to perform improved control interval processing. This is control-interval access (access to individual VSAM data blocks), and the ability to retrieve, update, insert, or delete records in the specified data set {AC.4::AC.4.5}.

For non-VSAM data sets, CONTROL is equivalent to UPDATE {AC.4::AC.4.6}.

- UPDATE

Allows users to read from, copy from, or write to the data set {AC.4::AC.4.7}. UPDATE does not, however, authorize a user to delete, rename, move, or scratch the data set {AC.4::AC.4.8}.

- READ

Allows users to access the data set for reading only {AC.4::AC.4.9}. (Note that users who can read the data set can copy or print it.)

- EXECUTE

For a private load library, EXECUTE allows users to load and execute, but not to read or copy programs (load modules) in the library {AC.4::AC.4.10}.

- NONE

The specified user or group is not permitted to access the resource or list the profile {AC.4::AC.4.11}.

These access types can be defined per user, group or for all users not addressed specifically by a user or group access entry (“universal access”) {AC.4::AC.4.12}. It is also possible to specify ID(*) in an ACL, which then applies to all RACF defined users, while the value for UACC applies to users not defined in RACF {AC.4::AC.4.13}. To modify those entries (as well as other parts of the resource profile) a user must be the owner of the profile, have ALTER access to the discrete profile of the resource or must have the SPECIAL attribute in his user profile {AC.4::AC.4.14}.

The access lists defined in a profile can be either a standard access lists, allowing access in general or a conditional access lists allowing access under defined conditions. Possible conditions are:

- the user must be logged on using a defined terminal that the user has been granted access to {AC.4::AC.4.15}
- the user must be logged on to a defined console {AC.4::AC.4.16}
- the batch job requesting access must have been submitted from a defined JES input device {AC.4::AC.4.17}
- the user must have entered the system from a defined network port {AC.4::AC.4.18}
- the resource manager has asserted a criteria, such as the name of an SQL role (SQLROLE), which applies to this check, on the authorization request (note: this applies only to a FASTAUTH type of authorization check) {AC.4::AC.4-R8-RACF-1}.

Access to resources can be controlled by discrete resource profiles or generic profiles for a set of resources of the same type. Discrete profiles protect one single resource (e. g. one data set) while generic profiles can be used to define a whole set of resources and protect them using a single profile based on patterns in the resource name. Whenever a discrete profile exists for a resource it has precedence over a generic profile that also would apply for the resource {AC.4::AC.4.19}. If more than one generic profiles would apply, z/OS always chooses the most specific profile applicable based on a matching algorithm {AC.4::AC.4.20}.

The access types above also apply to MVS resources other than data sets (called general resources). However while the usages remain hierarchical in definition (ALTER includes UPDATE, UPDATE includes READ, etc.) the interpretation and usage of the access types is the responsibility of each resource manager. For most resource managers and resources, the meaningful access types are NONE (the user/group has no access) or READ (the user/group does have access). For most cases access levels higher than READ convey no added authority (except that ALTER allows administration of a discrete profile). In specific cases the resource manager may treat UPDATE, CONTROL, and ALTER as granting additional authority. This security target and evaluation will not address all of those cases.

Algorithm to check for DAC access to MVS resources

RACF performs the following checks to identify, if a subject has the requested type of access to an object protected by RACF. This algorithm is performed after RACF has checked that the resource is protected by RACF:

i. If users attempt to access their own resources, RACF grants the request {AC.4::AC.4.43}. For example:
    - For tape and DASD data sets, if the user ID of the requesting user is the high-level qualifier of the data set name, RACF grants the request
    - For spool data sets, if the JESSPOOL class is active, RACF compares the user ID and node of the requester with the user ID and node of the creator of the spool data set (using the security token). If the user IDs match, RACF grants the request.

ii. If the resource manager has performed the authorization check using RACROUTE REQUEST=FASTAUTH (rather than RACROUTE REQUEST=AUTH) and in addition has specified AUTHCHKS=CRITONLY for this check, and has specified a criteria value using the CRITERIA keyword, RACF uses only the criteria-related conditional access list entries to make the determination, and skips to the criteria checking step below {AC.4::AC.4-R8-RACF-2}.

iii. RACF checks the user's access authority in the standard access list. If the user is in the list and if the specified access authority is sufficient to allow access, RACF grants the request {AC.4::AC.4.44}. If the user is in the list and if the specified access authority is less than the requested access, RACF continues processing at Step 7 (conditional access list checking) {AC.4::AC.4.45}. This prevents access based on ID(*), UACC, or the OPERATIONS attribute.

    This could happen if, for example, user JOE requests UPDATE access, and the standard access list includes ID(JOE) ACCESS(READ).

iv. RACF determines whether the user has access to the resource because the user is a member of a group and the group is on the standard access list {AC.4::AC.4.46}.

    Which group is used depends on whether list-of-groups processing is in effect. (List-of-groups processing is in effect if the SETROPTS command has been issued with the GRPLIST operand.)

    If list-of-groups processing is not in effect, RACF uses only the user's current connect group {AC.4::AC.4.47}.

    If list-of-groups processing is in effect, RACF finds all of the groups to which the user is connected that are also in the access list. Of these groups, RACF uses the group that has the highest access authority to the resource {AC.4::AC.4.48}. (For example, assume that a user is a member of groups A, B, and C. If group A has NONE access authority, group B has READ access authority, and group C has UPDATE access authority, RACF uses group C to determine the user's access.)

If the highest access authority is sufficient to allow the requested access, RACF grants the request. If the highest group that was found in the list does not have the requested authority, RACF continues processing at Step 8 {AC.4::AC.4.49} (conditional access list checking). This prevents access based on ID(*), UACC, or the OPERATIONS attribute.

v. If a user ID of * is found on the standard access list, the current user is defined to RACF without the RESTRICTED attribute, and the access authority granted to * is:
   - Sufficient to allow the requested access, RACF grants the request {AC.4::AC.4.50}
   - Not sufficient to allow the requested access, RACF continues processing at Step 7 {AC.4::AC.4.51} (OPERATIONS attribute checking)

vi. If the universal access authority (UACC) for the resource provides sufficient access authority and the requesting user is not defined with the RESTRICTED attribute, RACF grants the request {AC.4::AC.4.52}

vii. If the requesting user has the OPERATIONS attribute (or group-OPERATIONS if the resource is within the scope of that group) and OPERATIONS access is allowed for the class, RACF grants the request {AC.4::AC.4.53}

viii. RACF checks the user’s access authority in the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(SERVAUTH), or WHEN(JESINPUT). If the user is in the list, if the user meets the specified condition (such as logged on at the specified terminal), and if the specified access authority is sufficient to allow access, RACF grants the request. If the user is in the list with insufficient authority RACF continues processing at step 11. {AC.4::AC.4-R12-RACF-54}

ix. RACF determines whether the user has access to the resource because the user is a member of a group that meets a condition specified on the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(SERVAUTH), or WHEN(JESINPUT).

    Which group is used depends on whether list-of-groups processing is in effect.

    If the group to be used according to the preceding rules has sufficient access authority to allow the requested access, RACF grants the request {AC.4::AC.4.55}. If none of the user’s groups has sufficient authority, RACF continues with the next step.

x. If a user ID of * is found on the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(SERVAUTH), or WHEN(JESINPUT), and if the current user is defined to RACF without the RESTRICTED attribute, and if the current user meets the specified condition (such as logged on at the specified terminal), and the access authority granted to * is sufficient to allow the requested access, RACF grants the request {AC.4::AC.4.56}

xi. RACF checks the user’s access authority in the conditional access list specified with WHEN(PROGRAM). If the user is in the list, if the user meets the specified condition (such as running the specified program), and if the specified access authority is sufficient to allow access, RACF grants the request {AC.4::AC.4.57}.

Note: For DASD data sets, if program control is active and a controlled program is executing, RACF performs authorization checking for program access to data sets. If the user/program combination is in the conditional access list with sufficient authority to allow access to the data sets, RACF grants the request {AC.4::AC.4.58}.

xii. RACF determines whether the user has access to the resource because the user is a member of a group that meets a condition specified on the conditional access list (such as running a specified program). Which group is used depends on whether list-of-groups processing is in effect.

 If the group to be used according to the preceding rules has sufficient access authority to allow the requested access, RACF grants the request {AC.4::AC.4.59}. If the group is in the list and if the specified access authority is NONE, RACF denies the request {AC.4::AC.4.60}.

xiii. If a user ID of * is found on the conditional access list specified with WHEN(PROGRAM), and if the current user is defined to RACF without the RESTRICTED attribute, and if the current user meets the specified condition (such as logged on at the specified terminal or running the specified program), and the access authority granted to * is sufficient to allow the requested access, RACF grants the request {AC.4::AC.4.61}

xiv. Criteria Checking: For RACROUTE REQUEST=FASTAUTH, if the resource manager has asserted an SQL role name (SQLROLE) via the CRITERIA keyword, RACF checks for authority (via the user ID, a group, or * (for non-RESTRICTED users)) in the conditional access list specified with WHEN(SQLROLE(…)), and if the specified access authority is sufficient to allow access, RACF grants the request {AC.4::AC.4-R8-RACF-3}. If the resource manager has also specified AUTHCHKS=CRITONLY, and this step did not grant access, RACF denies the request {AC.4::AC.4-R8-RACF-4}.

xv. For access to uncataloged data sets, if SETROPTS CATDSNS is in effect, and none of the following is true, then RACF denies the request {AC.4::AC.4.62}:

 - The data set is newly-created in this job, or is a system temporary data set;
 - The data set is protected by a discrete profile;
 - The data set is cataloged in the Master catalog;
 - The user has access to FACILITY resource ICHUNCAT.dataset-name (truncated to 39 characters total, if needed);
 - The user has the SPECIAL attribute

xvi. For the DATASET class, if no profile is found and the SETROPTS PROTECTALL(FAILURES) option is in effect, RACF denies the request {AC.4::AC.4.63}.

If none of the above steps has granted access and the call to RACF has provided a nested ACEE and RACF is called with RACROUTE REQUEST=FASTAUTH and the object is eligible for nested ACEE processing, the algorithm for discretionary access control is repeated using the user ID specified in the nested ACEE {AC.4::AC.4-V1R7.1}. If audit is configured to audit the access attempt, both user IDs (the original and the nested) are contained in the audit record {AC.4::AC.4.V1R7.2}.

DAC for System Logger Objects in the LOGSTRM class

DAC for System Logger objects in the LOGSTRM class uses the basic MVS DAC algorithm explained above. The DAC algorithms apply in two cases:

- application programs that merely need to read or write to a log stream. The standard MVS DAC algorithm applies, using READ access for reading only, or UPDATE access for reading and writing, to resource log_stream_name in the LOGSTRM class {AC.4::AC.4-R10-Logger-1}.
- application programs that want to perform system management functions: defining, deleting, or updating the log stream definitions. The Security Management section will cover those usages.

DAC for UNIX objects

DAC controls for UNIX objects involve the user's effective UID and effective GID (which may be different from the user's real UID and real GID) {AC.4::AC.4-R8-USS-1} and the user's supplemental GIDs. If the user is connected to 5 groups, and 3 of them have GIDs, then he would have one real GID and 2 supplemental GIDs {AC.4::AC.4-R8-USS-2}.

DAC checking for UNIX file objects (files, directories) involves permission bits that specify the permissions (read, write, execute/search) separately for the object's owner, the owning group, and everyone else (the world), and optional access list entries (ACLs) with similar permission settings.

DAC checking for UNIX IPC objects (semaphores, shared memory) involves only permission bits.

Algorithm to check DAC access to UNIX file system objects

The following algorithm is used in the evaluated configuration to check the access to UNIX file system objects. The checks are performed by RACF using the effective user and group ID respectively, except as noted below.

- If the file is in a file system mounted as NOSECURITY and the attempt to open the file is made by a setuid program and the file has an owning UID of 0 or the owning UID of that of the setuid program, access is denied {AC.4::AC.4-V2R1-1}.
- If the program control extended attribute is turned on for a file in a file system mounted as NOSECURITY or NOSETUID, this extended attribute is not honored {AC.4::AC.4-V2R1-2}. A user must have READ permission to the BPX.FILEATTR.PROGCTL profile in the FACILITY class to update the program control extended attribute of a file {AC.4::AC.4-V2R1-3}.
- If the FSACCESS class is active and SETROPTS RACLISTed, and if the user is accessing the root directory in a mounted zFS file system (but not the system root directory), then if the MVS data set name of the file system container is protected by a profile in the FSACCESS class the user must have UPDATE access to that FSACCESS profile or the request will fail. Note that this is a standard RACF access check using the capabilities described above for DAC for MVS resources {AC.4::AC.4-R13-UNIX-1}.
- If the user has the RACF AUDITOR attribute, and read or search access for a directory is requested, access is granted {AC.4::AC.4.22}.

- If the user has UID(0), or has the TRUSTED or PRIVILEGED attribute, then access is granted automatically unless the user is executing a file. If the user is executing a file, access is denied only if none of the permissions bits grant execute access, and, if an ACL is present and the FSSEC class is active, no ACL entry grants execute access. Otherwise, access is granted {AC.4::AC.4.23}.
- If the user does not have search permission to all directories in the path of the file system object, access is denied {AC.4::AC.4.24}.
- If the UID matches the file owner UID, the file's "owner" permission bits are checked. If the "owner" bits allow the requested access, then access is granted {AC.4::AC.4.25}. If the UID matches the file owner UID and the owner bits do not allow the requested access, go to Step 15 {AC.4::AC.4.26}.
- If the FSSEC class is active, and an ACL exists, and there is an ACL entry for the requesting UID, then the permission bits of that ACL entry are checked. If the ACL entry allows the requested access, then access is granted {AC.4::AC.4.27}. Otherwise, if the ACL for the UID exists, but does not allow access, go to Step 14 {AC.4::AC.4.28}.
- If the GID matches the file owner GID, the file's "group" permission bits are checked. If the "group" bits allow the requested access, then access is granted {AC.4::AC.4.29}.
- If the FSSEC class is active, and an ACL exists, and there is an ACL entry for the requesting GID, then the permission bits of that ACL entry are checked. If the ACL entry allows the requested access, then access is granted {AC.4::AC.4.30}. If not, then the next ACL entry is checked until there are no more entries {AC.4::AC.4.31}.
- If any of the user's supplemental GIDs match the file owner GID, the file's "group" permission bits are checked. If the "group" bits allow the requested access, then access is granted {AC.4::AC.4.32}.
- If the FSSEC class is active, and an ACL exists, and there is an ACL entry for any of the user's supplemental GIDs, then the permission bits of that ACL entry are checked. If the ACL entry allows the requested access, then access is granted {AC.4::AC.4.33}. If not, then the next ACL entry is checked until there are no more entries {AC.4::AC.4.34}.
- If at least one matching ACL entry was found for the GID, or any of the supplemental GIDs, then processing continues with Step 14 {AC.4::AC.4.35}. If the GID, or any of the supplemental GIDs, matched the file owner GID, then processing continues with Step 15 {AC.4::AC.4.36}. Otherwise (neither the GID nor any of the supplemental GIDs matched either the file owner GID or an ACL entry), processing continues with the next step {AC.4::AC.4.37}.
- If the requesting user has the RESTRICTED attribute, and the UNIXPRIV class is active and RACLISTed, and the RESTRICTED.FILESYS.ACCESS resource is protected by a profile in the UNIXPRIV class, and the user does not have at least READ access, then go to Step 15 {AC.4::AC.4.38}.
- The file's "other" permission bits are checked. If the "other" bits allow the requested access, then access is granted {AC.4::AC.4.39}. Otherwise, go to Step 15.

- If the UNIXPRIV class is active and RACLISTed, and if the SUPERUSER.FILESYS.ACLOVERRIDE resource is protected by a profile in the UNIXPRIV class, then the user must have the correct access level as documented for the ck_access (IRRSKA00) callable service in z/OS Security Server RACF Callable Services. If the profile exists, it determines whether file access is granted or denied {AC.4::AC.4.40}.
- If the UNIXPRIV class is active and RACLISTed, and if the SUPERUSER.FILESYS resource is protected by a profile in the UNIXPRIV class, then the user must have the correct access level as documented for the ck_access (IRRSKA00) callable service in z/OS Security Server RACF Callable Services. If the profile exists, it determines whether file access is granted or denied {AC.4::AC.4.41}.

Access is denied, if none of the above steps has explicitly granted access {AC.4::AC.4.42}.

Algorithm to check DAC access to UNIX IPC objects

The discretionary access control rules allow access to an IPC object,

- if the user has an effective user ID of zero {AC.4::AC.2.70}
- if the user is the owner or creator of the IPC object and the requested type of access is allowed by the owner related permission bits {AC.4::AC.2.71}
- if the user is neither the owner or creator of the IPC object but is a member of the IPC object's creating group or owning group and the requested type of access is allowed by the group related permission bits {AC.4::AC.2.72}
- if the user is neither owner nor creator of the IPC object and also is not a member of the IPC object's creating group or owning group and the access is allowed by the other related permission bits {AC.4::AC.2.73}

If none of the above mentioned conditions is satisfied, permission is denied by the discretionary access control rules for IPC objects {AC.4::AC.2.74}.

DAC for LDAP LDBM objects

Access to LDAP directory entries and attributes is defined by Access Control Lists (ACLs). Each entry in the directory contains a special set of attribute/value pairs which describe who is allowed to access information within that entry. Attributes associated with access control are aclEntry, aclPropagate, aclSource, entryOwner, ownerPropagate, and ownerSource. The aclEntry and entryOwner attributes appear to be part of the entry, but may in fact be logically associated with an entry, but physically present in some parent entry higher in the directory tree. When we talk about an LDAP ACL (Access Control List) we mean the combination of the entryOwner and aclEntry attribute values. If the user is the entryowner they have administrator level permissions to the entry. If they are not the entryOwner then we look to the aclEntry attribute values to determine the access.

The TOE controls access to all directory entry objects based on the following security attributes:

- Entry Owner Information
    - entryOwner: defines the DN(s) of the LDAP user(s) or group(s) considered to own this entry.

    - ownerPropagate: indicates whether to propagate the ownership of the entry to all descendant entries, until another entry with ownerPropagate is found.
- Access Control Attributes (ACA)
    - aclEntry: defines the access control information, which can specify access permissions (grant, deny) for LDAP users or groups that control access to the complete entry, specific named attributes in the entry, or all attributes in the entry that belong to a specific attribute class

      aclEntry specifications can also filter the user's access based on various characteristics such as the bind DN , alternate DNs , pseudo DNs , groups that the bind/alternate DNs belongs to , IP address of the client connection , time of day that directory entry was accessed , day of week that directory entry was accessed , the bind mechanism used , whether or not bind encryption was used.
    - aclPropagate: indicates whether to propagate access control information of the entry to all descendant entries, until another entry with aclPropagate is found.

Algorithm to check for DAC access to LDAP LDBM objects

The Access Control List for an LDAP LDBM object (entry DN) is determined in the following way:

a) If there is a set of explicit access control attributes for the object , then the object’s Access Control List applies {AC.4::AC.4-R8-LDAP-1}.

b) If there is no explicitly defined set of access control attributes, then traverse the directory tree upwards until an ancestor node is reached with a set of propagating access control attributes {AC.4::AC.4-R8-LDAP-2}.

If no such ancestor node is found, the default access rights will apply {AC.4::AC.4-R8-LDAP-3}. The default access rights are predefined as aclEntry: group:CN=ANYBODY:normal:rsc:system:rsc and cannot be changed by the Directory Administrator {AC.4::AC.4-R8-LDAP-4}.

When determining base access (before filtering), processing stops as soon as access can be determined {AC.4::AC.4-R12-LDAP-5} based on access evaluation as described below:

- The first check for access is done by comparing the subject’s LDAP user ID (bind DN) and LDAP groups with the effective entryOwner attribute values. If there is a match with any of the entryOwner values then the subject has full access to the object {AC.4::AC.4-R8-LDAP-6}. The LDAP Administrator is additionally considered to have ownership authority for all objects in the directory tree. If the bind DN is an LDAP Administrator with the appropriate administrative role authority, full access is first applied. Then permissions for all aclFilter values matching the LDAP administrator's bind DN and additional access information are applied prior to the server processing the ACL check. {AC.4::AC.4-R13-LDAP-7}.
- The subject may be granted different access permissions to an object, from specific access permissions for the subject’s DN and from group memberships (including the authenticated and anybody groups). The LDAP server uses the following algorithm to

© Copyright IBM Corp. 2015

determine which permissions to grant a DN based on the values in the aclEntry attribute:

- if there is a specific value for the subject's DN, the subject gets those permissions only {AC.4::AC.4-R8-LDAP-8}
- else if there is a cn=this value and the subject's DN is the distinguished name (DN) of the object, the subject gets those permissions only {AC.4::AC.4-R8-LDAP-9}
- else if there are one or more group values that the subject is a member of, the subject gets the union of the permissions for those groups {AC.4::AC.4-R8-LDAP-10}
- else if there is a cn=authenticated value and the subject is authenticated to the directory with an LDAP bind operation, the subject gets those permissions only {AC.4::AC.4-R8-LDAP-11}
- else if there is a cn=anybody value, the subject gets those permissions only {AC.4::AC.4-R8-LDAP-12}
- otherwise the subject gets no permissions {AC.4::AC.4-R8-LDAP-13}

Permissions may be add (a) or delete (d) or both at the object level {AC.4::AC.4-R8-LDAP-17}, or read (r), write (w), search (s), or compare (c) or a combination of these at the attribute {AC.4::AC.4-R8-LDAP-18} or attribute class {AC.4::AC.4-R8-LDAP-19} level.

Permissions may specify grant or deny for any of the above {AC.4::AC.4-R8-LDAP-23}.

Each of the access permissions is discrete. One permission does not imply another. {AC.4::AC.4-R8-LDAP-14}

Permissions may be specified for the attribute classes normal, sensitive, critical, restricted, or system {AC.4::AC.4-R8-LDAP-20}

Administrator-defined attributes may be specified to be in the normal, sensitive, or critical attribute classes {AC.4::AC.4-R8-LDAP-21}. The default attribute class for administrator-defined attributes is normal {AC.4::AC.4-R8-LDAP-22}.

With the support for attribute-level permissions as well as grant/deny support, the order of evaluation of the separate permissions clauses is important. The access control permissions clauses are evaluated in a precedence order, not in the order in which they are found in the ACL entry value {AC.4::AC.4-R8-LDAP-15}. With this support, there are four types of permissions settings: access-class grant permissions, access-class deny permissions, attribute-level grant permissions, and attribute-level deny permissions. The precedence for these types of permissions is as follows (from highest precedence to lowest): {AC.4::AC.4-R8-LDAP-16}

- attribute-level deny permissions
- attribute-level grant permissions
- access-class deny permissions
- access-class grant permissions

Using this precedence, a deny permission takes precedence over a grant permission (for the same item specified) while attribute-level permissions take precedence over access-class permissions.

After a user's base access has been determined, it may be modified by Filter ACL entries.

{AC.4::AC.4-R12-LDAP-24}Filter ACLs can set permissions based on any of the following:

- bind DN
- alternate DNs
- pseudo DNs
- groups that the bind or alternate DNs belong to
- IP address of the client connection
- time of day that directory entry was accessed
- day of week that directory entry was accessed
- the bind mechanism used
- whether or not bind encryption was used

Filters support wildcards. Filters will have the same syntax support as LDAP search filters, where logical rules can be specified, such as “&” (and), “|” (or), and “!” (not), to name a few {AC.4::AC.4-R12-LDAP-25}.

{AC.4::AC.4-R12-LDAP-26} To allow flexibility, the new aclEntry filtering mechanism will also support three operation values that allow users to specify the way in which filtered ACLs will take effect:

- replace --the base effective ACL is replaced by the filtered ACLs.

  If administrators want a client from a given IP address to only have a specific set of permissions, they would use replace. (This is similar to what Sun ONE supports.)
- union --the base effective ACL is unioned with the filtered ACLs, resulting in a new effective ACL. This would be used to expand permissions.

  If administrators want a client from a given IP address to have a specific set of permissions, at a minimum, they would use union.
- intersect --the base effective ACL is intersected with the filtered ACLs. This would be used to reduce permissions.

  If administrators want a client from a given IP address to have a specific set of permissions, if and only if they already have the permissions, they would use intersect.

As with the operator precedence described above, filter ACL entries specifying “deny” take precedence over entries specifying “grant” {AC.4::AC.4-R12-LDAP-27}.

8.4.2.2 Authority Checking for PKCS11 Cryptographic Tokens in the ICSF TKDS

DAC for PKCS#11 Cryptographic Tokens in the ICSF TKDS occurs using profiles in the CRYPTOZ resource class, using the basic MVS DAC algorithm described above.

The access control defined in the PKCS#11 standard was designed for systems that have no security manager. Access to token information in the standard is granted based on the knowledge of a PIN. In the definition there are two types of users, the standard user (User) and the security officer (SO). Each has their own PIN. The SO can initialize a token (zero the contents) and set the User's PIN. The SO can also access the public objects on the token but not the private ones. The User has access to the private objects on a token and has the power to change his or her own PIN. The User cannot reinitialize the token. The role one is allowed to take depends on the PIN entered. Thus a single person can fill both roles by having knowledge of both PINs.

On z/OS these two roles will be simulated by using SAF profiles in a new Class called CRYPTOZ. There will be no PINs. Each token defined will have a unique token name (label) up to 32 characters in length. The permitted characters are alphanumeric, national (@,#,$) or period (.). The first character must be alphabetic or national. Lowercase letters are permitted but will be folded to uppercase. (This is the same naming restriction as PKDS labels.) There will be two CRYPTOZ Class resources checks performed for tokens:

- USER.token-name -- Controls the User role
- SO.token-name -- Controls the SO role

The different access levels provide the following functionality:

- The 3 standard PKCS#11 access types (User R/W, SO R/W, User R/O)
    - R/O vs R/W not end-user controlled
- Plus 3 z/OS unique access types
    - Weak SO -- An SO that can modify CAs contained in a token but not initialize the token
    - Strong SO -- An SO that can add or remove private objects in a token (e.g., a server administrator)
    - Weak User -- A user that cannot change the trusted CAs contained in a token

CRYPTOZ DAC Table {SM.7::AC.4-R9-ICSF-1}:

CRYPTOZ Resource Name	**Access of: READ**	**Access of: UPDATE**	**Access of: CONTROL**
SO.token-label	Weak SO -- read / create / delete / modify / use public objects	SO R/W -- Weak SO plus create / delete token	Strong SO -- SO RW plus read (but not use) private objects, create / delete / modify private

CRYPTOZ Resource Name	Access of: READ	Access of: UPDATE	Access of: CONTROL
			objects
USER.token-label	User R/O -- read / use public and private objects.	Weak User -- User R/O plus create / delete / modify private and public objects (cannot add / delete / modify certificate authority objects)	User R/W -- Weak User plus add / delete / modify certificate authority objects

Table 24: CRYPTOZ Resources and Access Semantics

8.4.2.3 Access Control Considerations for the ISPF Client Gateway

The ISPF Client Gateway allows two additional ways for users to execute TSO/E commands and ISPF functions:

- The administrator can configure the HTTP server to allow the HTTP client to request invocation of the ISPF Client Gateway control program ISPZINT. In this case, ISPZINT and any commands it invokes will run with the user ID configured by the administrator (which may be the client user's identity or an identity specified by the administrator) {AC.4::AC-R10-ISPF-1}.

- An existing user session (batch job, UNIX process, etc.) can invoke ISPZINT directly. In this case, ISPZINT and any comands it invokes will run with the user ID of the invoking user {AC.4::AC-R10-ISPF-3}.

Based on information supplied by its client, ISPZINT will either run a single command or it will run a command and leave the session active to run subsequent commands. When leaving the session active for subsequent commands, ISPZINT will ensure that those commands run with the same user ID as the original command {AC.4::AC-R10-ISPF-5}.

8.4.3 Audit (FAU)

8.4.3.1 Generation of audit records

The TOE provides a general facility to collect data required for auditing and accounting services. This function, the System Management Facilities (SMF), collects and records system and job-related information that an installation can use for such tasks as the following:

- Billing users
- Reporting reliability
- Analyzing the configuration
- Scheduling jobs

- Summarizing direct access volume activity
- Evaluating data set activity
- Profiling system resource use
- Maintaining system security

This component is used by the TOE to collect security-related auditing information as required by FAU_GEN.1 and FAU_GEN.2.

Each SMF record consists of a standard header which contains (among other information) the type of the record and the time the record was produced {AU.1::AU.1.1}. SMF supports up to 256 different record types. SMF records can only be generated by authorized processes or processes specifically authorized to generate specific types of SMF records under the mediation of the TOE {AU.1::AU.1.2}.

One record type is usually reserved for a whole class of events where the individual events are identified by the record subtype or event code in the header of the SMF record.

RACF as the central access control function has three SMF record types reserved for its use (80, 81, 83), with record type number 80 being the most important one. The information recorded in this record type contains (among other non security related information):

- The record type
- Time stamp (time and date)
- System identification
- Event code and qualifier
- User identification
- Group name
- Authorities used to successfully execute commands or access resources
- Reasons for logging
- Command processing error flag
- Foreground user terminal ID or other port-of-entry information
- Job log number (job name, entry time, and date)
- RACF version, release, and modification number

Each record contains further data specific to the event code and qualifier {AU.1::AU.1.3}.

The administrator can configure RACF and other elements of the TOE to generate audit records for all events listed in the table in FAU_GEN.1 {AU.1::AU.1-R9-MULTI-1}.

z/OS provides the capability to search the audit trail for specific events and relate them such that events related to a specific user can be extracted from the audit trail {AU.1::AU.1.4}.

Tools exist that allow user with access to the audit trail data to search the audit trail for specific events, for audit events related to specific jobs / users and other criteria

{AU.1::AU.1.5}. Tools exist that transfer the audit data into human readable format {AU.1::AU.1.6}.

RACF also allows LDAP clients (typically servers outside of the TOE, residing on the network) that have authenticated using an ICTX-style DN to request RACF to generate audit records to record events that have occurred externally to the TOE. The requester provides information about the user involved with the event, the kind of event, and the resource name and resource class name (any class except DATASET) associated with the event.

The LDAP client uses an LDAP extended-operation to request this auditing function. Usage of the remote auditing service requires the LDAP client to have READ authority to FACILITY resource IRR.LDAP.REMOTE.AUDIT {AU.1::AC.2-R9-EIM.5}. The audit record will be created as an SMF type 83 subtype 4 record {AU.1::AU.1-R9-EIM-1}.

If an application has created an ACEE and specified ICTX= on the RACROUTE REQUEST=VERIFY to associate a X.500-format distributed identity with the RACF user's ACEE, RACF will include that distributed identity in the SMF records that it creates. {AU.1::AU.1-R12-RACF-1}

8.4.3.2 Protection of the audit trail

SMF writes audit records into either

1. Dedicated SMF data sets that have been defined during system configuration. At least two SMF data sets must be defined by the administrator for compliance with the evaluated configuration. Those data sets need to be protected against unauthorized access by appropriate RACF access control lists. The administrator guidance documentation provides specific guidelines for the protection of the audit trail using RACF.

Or

2. A system log stream, which may reside solely in DASD data sets, or in a combination of data sets and a coupling facility structure for better performance, as specified by the administrator. The administrator configures profiles in the LOGSTRM class to control who can access the data while it exists in the managed log stream, and profiles in the DATASET class to control access to any data extracted from the log stream.

Using MVS Data Sets for SMF

When the system is started SMF searches for the first non-full data set in the list of SMF data sets defined. This data set becomes the active SMF data set used to store audit records. Once this data set is full, SMF marks the data set to be processed by the SMF Dump program and takes the next empty data set as the active, searching the list of SMF data sets in a wraparound way {AU.2::AU.2.2}. The operator is also alerted to switch the data set.

SMF data sets that are full need to be processed by the SMF Dump program, IFASMFDP. This program copies the content of a full SMF data set to another data set (the “dump data set”) defined by the installation and marks the SMF data set as empty {AU.2::AU.2.3}. The SMF Dump program itself creates two SMF records (Dump Header and Dump Trailer) that are stored in the beginning and at the end of the dump data set {AU.2::AU.2.4}. Dump data sets must be protected by RACF access control lists.

If no non-full data set is found, SMF stores the records in its buffers until a data set is made available {AU.2::AU.2.5}. If the TOE is configured according to the administrative guidance, the system will halt if no buffer space is left {AU.2::AU.2.6}.

Using a System Log Stream for SMF

In contrast to using MVS data sets directly, when using a log stream for the SMF data only one logical stream exists. Although this stream may reside in multiple MVS data sets as determined by system logger processing, the administrator will view the stream as one logical entity, starting with the earliest available data and ending with the current data, rather than dealing with the individual data sets.

Operators do not need to switch SMF data sets, nor dump them to archive storage, nor clear them. Rather, the data can simply reside in the logger-managed data sets.

z/OS provides the IFASMFDL utility program that can extract an administrator-specified set of SMF data from the log stream, based on time/date, system ID, and/or SMF record type and write that extracted data to a standard MVS data set for later processing {AU.2::AU-R9-SMF-1}.

IFASMFDL can invoke exit routines, just as IFASMFDP can, and so the RACF SMF Unload routine will work with IFASMFDL just as with IFASMFDP, providing an interpreted flat-file of RACF-relevant security records for subsequent analysis {AU.2::AU-R9-RACF-1}.

8.4.4 Cryptographic Functions (FCS)

8.4.4.1 General Cryptography

Several components of the TOE use cryptographic functions as part of their security functions. With the inclusion of the Integrated Cryptographic Services Facility (ICSF) the cryptographic functions may be provided by hardware coprocessors attached to the TOE. ICSF checks for the availability of hardware support for individual cryptographic functions and uses this when appropriate. In the case where no cryptographic coprocessor is attached to the TOE, the components that use ICSF for cryptographic operations (IPSec, System SSL, z/OS Network Authentication Service) will use software implementation of the cryptographic algorithms. IPSec always requires ICSF for AES support, whether using the hardware or software. SSH will either use its own software implementation of the cryptographic algorithms but can also be configured to use CPACF (via calls to ICSF) for AES, SHA-1 and the SHA-2 family of hash functions and will use hardware support for the key generation process. For the RACDCERT command, the command issuer chooses, by the keywords chosen, whether to use ICSF or a software implementation.

Note that CPACF is not considered a cryptographic coprocessor but a native capability of the z/Architecture processor. While the functions provided by CPACF may differ by different processor models, the functions provided by the CPACF instructions may be used by any application.

ICSF provides secure PKCS #11 support. This support requires the Crypto Express4S card to be configured with the EP11 card code and secure keys to be encrypted and stored in the TKDS {CR.2::CR2-V2R1-ICSF-1}.

Support for using secure RSA and ECDSA PKCS #11 CA certificates, generation of secure PKCS#11 key pairs for key generation requests and CMP requests is provided {CR.2::CR2-V2R1-ICSF-2}.

This support function is used by PKI services and RACDCERT for certificate public/private key pair generation using the new secure PKCS#11 support. See the description in the PKI Services section and the RACDCERT command.

Cryptographic Functions for Application Use

The TOE also provides various cryptographic functions via ICSF that are available for use by applications running on the system.

{CR.2::CR-R13-ICSF-1} The CSFPSKE and CSFPSKD services provided by ICSF provide the ability for an application to perform encryption and decryption operations with the following algorithms and key sizes:

- TDES with CBC block chaining mode and 168-bit key size;
- AES with CBC or GCM block chaining modes and 128- or 256-bit keys.

{CR.2::CR-R13-ICSF-2} The CSFPPKV and CSFPPKS services provided by ICSF provide the ability for an application to perform encryption and decryption operations with the following algorithms and key sizes:

- RSA with key sizes up to 4096 as defined by PKCS #1 v1.5

{CR.2::CR-R13-ICSF-3} The CSFPOWH service provided by ICSF provides the ability for an application to perform message digest generation in accordance with the following cryptographic algorithms:

- SHA-1
- SHA-256
- SHA-512

{CR.2::CR-R13-ICSF-4} The CSFPOWH service provided by ICSF provides the ability for an application to generate and verify signatures using the following algorithms and key sizes:

- DSA with:
  - L=1024, N=160 bits,
- RSA with 2048 to 4096-bit keys using PKCS#1 V1.5
- ECDSA with:
  - NIST curve=192
  - NIST curve=224
  - NIST curve=256
  - NIST curve=384
  - NIST curve=521

as defined in FIPS 186-3 , Digital Signature Standard (DSS), and specified in ANSI X9.62-1998, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA).

The hash function can be selected to be either SHA-1, SHA-256, or SHA-512. Note: other hash functions are supported but no claims about those are made in this Security Target.

Note that on z/OS executing on a z114 or z196 processor that also have a CryptoExpress3 coprocessor installed in coprocessor mode, or a zEC12 that have Crypto Express3 or Crypyo Express4s, the coprocessor can provide an implementation of ECDSA .

FIPS 140-2

In addition to providing cryptographic support, several components of z/OS have been designed to meet the Federal Implementation Processing Standard (FIPS) 140-2 Level 1 criteria. FIPS is a standard that has been issued by the National Institute of Standards and Technology and Communications Security Establishment (CSE) of the Government of Canada . This standard specifies security requirements for a cryptographic module which is utilized within a security system to protect sensitive or valuable information. To meet this standard, cryptographic modules are tested against requirements defined in the FIPS PUB 140-2, Security Requirements for Cryptographic Modules. These security requirements cover 11 areas related to the design and implementation of a cryptographic module.

The z/OS components designed to meet FIPS 140-2 Level 1 are ICSF and System SSL. When utilizing these components in FIPS mode, they will restrict cryptographic processing to what is approved or allowed in FIPS mode.

For System SSL, The algorithms/protocols will be restricted to: symmetric TDES (3-key), AES CBC (128 and 256 bit); hashing algorithms SHA-1 (160-bit) and SHA-2 (224, 256, 384, 512 bit); asymmetric algorithms RSA (1024-4096 bit) and DSA (1024 and 2048 bit); Diffie-Hellman key agreement algorithm DH (2048 bit); TLS V1.1, and TLS V1.2 protocols. {CR.1::CR-V2R1-SSL-1}

For ICSF, the algorithms will be restricted to: symmetric algorithms TDES (3-key), AES ECB, CBC, GCM (128, 192 and 256 bit); hashing algorithms SHA-1 (160 bit) and SHA-2 (224, 256, 384 and 512 bit); asymmetric algorithms RSA (1024-4096 bit), DSA (1024 and 2048 bit) and ECDSA (192-521 bit); Diffie-Hellman key agreement algorithms DH (1024-2048 bit) and EC-DH (160-521 bit).{CR.1::CR-V2R1-ICSF-1}

Approved cryptographic algorithms are tested for conformance by inputting defined test vectors and comparing the output results against expected output vectors. {CR.1::CR-V2R1-SSL-2} {CR1::CR-V2R1-ICSF-2} which include the Diffie-Hellman algorithm.

Each component has unique methods to control FIPS mode execution.

System SSL applications that execute in FIPS mode must utilize the gsk_fips_state_set API prior to other System SSL functions {CR.1::CR-R12-SSL-3}. System SSL allows switching from FIPS to non-FIPS mode but not vice versa {CR.1::CR-R12-SSL-4}.

Utilizing the FIPSMODE start up option for the ICSF started task allows to operate ICSF in FIPS mode. FIPSMODE allows ICSF to be started in strict FIPS mode, compatibility mode which allows select applications to execute in FIPS mode or non-FIPS mode {CR.1::CR-R12-ICSF-3}. Compatibility mode applications by default execute in FIPS mode {CR.1::CR-R12-ICSF-4}. FIPSEXEMPT.<token-label> in the cryptoz class allows a user (application) to be exempt from FIPS when using a specified PKCS#11 token {CR.1::CR-R12-ICSF-5} or the calling application has indicated that the specified key must always be used in a FIPS 140-2 compliant fashion {CR.1::CR-R12-ICSF-6}.

ICSF and System SSL modules that execute within FIPS mode have been digitally signed during the bind process using RSA and SHA-256 signatures. Upon validation of the signatures, the modules are verified to ensure they have not been tampered with since being built. Verification is performed during the module load process {CR.1::CR-R12-SSL-5} {CR.1::CR-R12-ICSF-7}.

In the evaluated configuration, Application Transparent TLS (AT-TLS) {CR.1::CR-R12-CS-1}, Network Security Services daemon (NSSD) {CR.1::CR-R12-CS-2}, IKED {CR.1::CR-R12-CS-3} and IPSec support in the Communications Server stack {CR.1::CR-R12-CS-4} allow for execution in either FIPS or non-FIPS mode.

The following hardware support options for cryptographic functions are available:

CPACF

This feature is part of the instruction set of the z/Architecture. Instructions are available for DES encryption and decryption, TDES encryption and decryption and SHA-1 and SHA-2 hashing. In addition a DES based pseudo-random number generator is provided. The instructions for those operations are part of the general instructions of a z/Architecture processor and may therefore be used by programs in any processor state. The instructions are:

- CIPHER MESSAGE (KM)
- CIPHER MESSAGE WITH CHAINING (KMC)
- COMPUTE INTERMEDIATE MESSAGE DIGEST (KIMD)
- COMPUTE LAST MESSAGE DIGEST (KLMD)

The KMC instruction also provides a DES based pseudo random number generator (which is not used by the software to implement any security function of the evaluated configuration). For details of those instructions see [ZARCH].

Specific z/Architecture processor models also support 128-bit, 192-bit, or 256-bit AES encryption/decryption, and SHA-224, SHA-256, SHA-384, or SHA-512 message digests.

CEX3/CEX4S in CEX3C/CEX4C mode

The CEX3 and CEX4S in CEX3C/CEX4C mode are PCI based coprocessor cards with their own main processor (a pSeries processor), a cryptographic hardware coprocessor and their own memory. They contains an operating system (Linux) on top of which application programs implement the functions of IBM's Common Cryptographic Architecture (CCA). Basically CCA commands are passed by the TOE to the coprocessor, processed there and the result is passed back to the TOE. Logical access to the coprocessor functions is controlled by the TSF and unprivileged programs can access those functions only through the ICSF component of the TSF and only for services they are allowed to use.

The coprocessors have the ability to generate RSA key pairs and retain the private key in the coprocessor. When generating such a key pair the coprocessor would only pass back the public key and a key identifier that can be used to request the coprocessor to use a specific private key. The private key will never leave the coprocessor in clear. Only export in encrypted form for backup purposes is possible.

The CEX3C/CEX4C support up to 4096-bit RSA key operations for clear and secure keys.

CEX3/CEX4S in CEX3A/CEX4A mode

The CEX3/CEX4S in CEX3A/CEX4A mode are PCI based cryptographic coprocessor cards that contains the same hardware as in CEX3C/CEX4C mode but which are used only as an accelerator for RSA encryption and decryption, and which have disabled the general purpose processor, memory and operating system. RSA encryption and decryption are the only cryptographic functions the coprocessors can perform. Since the CEX3A/CEX4A provide no own storage, the key has to be provided by the TOE each time it uses the coprocessor. The coprocessor can accept keys both in "normal" format as well as in CRT format (as defined in PKCS#1). The operation code submitted to the card identifies the operation and the key format. Operation code, input data, output data, data length, key length and the key are passed in a block to the coprocessor, which then performs its operation and passed the result back in the output data field. For applications that just need fast RSA encryption and decryption (e. g. a server that allows a lot of TLS based connections), this provides a significantly faster method for RSA operations than using the CEX3/CEX4S in CEX3C/CEX4C mode the overhead associated with the operations on the CEX3C/CEX4C cards. Of course the CEX3A/CEX4A do not provide an option for "secure" private keys.

8.4.4.2 Program Signing and Verification

The TOE supports digital signatures for program objects stored in PDSE data sets. The digital signature for a program object is optionally created during the process of "binding" the object modules into the form of an executable program object and storing it into the PDSE data set. Later, when a user attempts to load or execute that program the system can optionally validate the signature and use the result of the validation to determine whether to allow use of the program or not.

Program Signing

{SP.4::SP.4-R12-Binder-1} Users executing the Binder to create a program object in a PDSE may optionally specify the SIGN=YES option to request that the Binder attempt to digitally sign the program object and save the signature in the PDSE directory when it saves the program object.

{SP.4::SP.4-R12-RACF-1} The Security Administrator authorizes users to perform program signing by

(a) creating FACILITY class profiles of the form IRR.PROGRAM.SIGNING.groupname.userid or IRR.PROGRAM.SIGNING.userid or IRR.PROGRAM.SIGNING.groupname or IRR.PROGRAM.SIGNING (listed here in priority order, and where groupname is the user's current connect group) and

(b) providing APPLDATA information in that profile that specifies a hashing algorithm to use and the key ring (both owning userid and key-ring-name) that contains the code signing digital certificate to use, and

(c) authorizing the user to access that key ring.

{SP.4::SP.4-R12-RACF-2} The specified key ring must contain a digital certificate to use for signing, together with the RSA private key for that certificate, and the CA certificate(s) for the certificate. Certificate requirements:

- The code signing certificate and each CA certificate in the CA chain must be signed using either the sha256WithRSAEncryption or sha1WithRSAEncryption signature algorithms.
- The code signing certificate must have code-signing capability in one of the following ways:
    - Either the certificate has no KeyUsage extension, or
    - the certificate has a KeyUsage extension with at least the digitalSignature and nonRepudiation indicators enabled.
- Each CA certificate in the chain must have certificate-signing capability in both of the following ways:
    - Either the certificate has no BasicConstraints extension, or
      the certificate has a BasicConstraints extension with the cA indicator enabled.
    - Either the certificate has no KeyUsage extension, or
      the certificate has a KeyUsage extension with at least the keyCertSign indicator enabled.

Program Signature Verification

{SP.4::SP.4-R12-RACF-3} The Security Administrator enables program signature verification by:

- Defining the FACILITY class profile IRR.PROGRAM.SIGNATURE.VERIFICATION and specifying in the APPLDATA the owning user ID and key-ring name of the key ring that holds the CA certificates associated with the various code signing certificates, including the IBM-supplied certificate with the label 'STG Code Signing CA', and
- Defining a PROGRAM profile for IRRPVERS, e.g.,

  ```
  RDEFINE PROGRAM IRRPVERS ADDMEM('SYS1.SIEALNKE'//NOPADCHK) UACC(READ)
  SIGVER(SIGREQUIRED(YES) FAILLOAD(ANYBAD) SIGAUDIT(ANYBAD))
  ```
- Defining PROGRAM profiles to protect each signed program that the administrator wants verified during use, and specifying in that profile's SIGVER segment various options that tell RACF how to process the verification for the protected programs:
    - SIGREQUIRED (YES | NO):
        - YES indicates that the program must have a digital signature;
        - NO indicates that it might have one, but is not required to have one.
    - FAILLOAD(ANYBAD | BADSIGONLY | NEVER):
        - ANYBAD indicates that if any failures occur during program signature verification (including administrative setup errors such as missing or incorrectly defined keyrings, signatures by untrusted signers, or incorrect or missing signatures) the system should disallow use of the program.

    - BADSIGONLY indicates that if the signature itself is incorrect or missing the system should disallow use of the program.
    - NEVER (the default) indicates that a problem with signature verification should not prevent use of the program.
  - SIGAUDIT (ALL | SUCCESS | ANYBAD | BADSIGONLY | NONE):
    - ALL indicates that RACF should audit the result of all signature verification operations using an SMF type 80 audit record.
    - SUCCESS indicates that RACF should audit any successful signature verification operations.
    - ANYBAD indicates that RACF should audit any failing signature verification operation.
    - BADSIGONLY indicates that RACF should audit signature verification operations that fail due to an incorrect or missing signature.
    - NONE indicates that RACF should not audit any program verification operations.
- Refreshing the PROGRAM profiles using SETR WHEN(PROGRAM) REFRESH, and running program IRRVERLD. (Note: Running IRRVERLD is required only when the administrator has made signature verification setup changes since the last IPL.)

8.4.4.3 PKI Services

PKI Services allows an installation to establish a Public Key Infrastructure (PKI) and serve as a certificate authority for its internal and external users, issuing and administering digital certificates in accordance with the organization's policies. Users can use a PKI Services application to request and obtain certificates through their own Web browsers {SM.5::SM-R8-PKI-1}, while authorized PKI administrators approve, modify, or reject these requests through their own Web browsers, 32-bit versions of Microsoft Internet Explorer {SM.5::SM-R8-PKI-2}or Mozilla-based browsers such as Mozilla Firefox {SM.5::SM-R8-PKI-3}. The Web applications provided with PKI Services are highly customizable. An installation can allow automatic approval for certificate requests from certain users {SM.5::SM-R8-PKI-4} and, to provide additional authentication, add host IDs, such as RACF user IDs, to certificates issued for certain users {SM.5::SM-R8-PKI-5}. Installations can also issue certificates for browsers, servers, and other purposes, such as virtual private network (VPN) devices, smart cards, and secure e-mail.

PKI Services CA's signing key length can be up to 4096 bits for RSA, up to 1024 bit for DSA, 521 bits for NIST ECC, or 512 bits for Brainpool ECC {SM.5::SM-R12-PKI-6}.

PKI Services can generate RSA keys up to 4096 bits for the certificates if requested {SM.5::SM-R12-PKI-37} and store them as secure keys in the ICSF TKDS {SM.5::SM-V2R1-PKI-3}.

PKI Services can generate NIST ECC keys with maximum length 521 bits for the certificates if requested {SM.5::SM-R12-PKI-41} and store them as secure keys in the ICSF TKDS {SM.5::SM-V2R1-PKI-1}.

PKI Services can generate Brainpool ECC keys with maximum length 512 bits for the certificates if requested {SM.5::SM-R12-PKI-42} and store them as secure keys in the ICSF TKDS {SM.5::SM-V2R1-PKI-2}.

PKI Services may be configured to accept and process certificate requests and revocation requests through the Certificate Management Protocol (CMP). Certificate requests may contain the public key of a public/private key pair, or may be omitted to instruct the PKI Service CMP CGI program to generate the key pair. {SM.5::SM-R12-PKI-43}

Supported Certificate Fields and Extensions

PKI Services certificates support fields and extensions defined in the X.509 version 3 (X.509v3) standard. It can include the following types of extensions:

Standard extensions {SM.5::SM-R8-PKI-7}

The standard X.509v3 certificate extensions:

- authority information access
- authority key identifier
- basic constraints
- certificate policies
- certificate revocation list (CRL) distribution points
    - Distinguish Name format
    - Uniform Resource Identifier format using LDAP or HTTP protocol
- key usage
    - digitalSignature
    - nonRepudiation
    - keyEncipherment
    - dataEncipherment
    - keyAgreement
    - keyCertSign
    - CRLSign
- extended key usage
    - serverauth
    - clientauth
    - codesigning
    - emailprotection
    - timestamping
    - ocspsigning

    - mssmartcardlogon
- subject alternate name
    - email
    - domain
    - IPAddress
    - uniformResourcesIdentifier
    - OtherName
    - subject key identifier

Other extensions

- host identity mapping {SM.5::SM-R8-PKI-8}. This extension associates the subject of a certificate with a corresponding identity on a host system, such as with a RACF user ID.
- Custom extensions {SM.5::SM-R12-PKI-44}. This allows users to create any extensions conformed to the Extension structure: a sequence of OID, critical flag and value.

Supported Certificate Revocation List Fields and Extensions

PKI Services generates CRLs that comply with the X.509 version 3 (X.509v3) standard. The following extensions are included:

CRL extensions: {SM.5::SM-R8-PKI-9}

- AuthorityKeyIdentifier
- CRLNumber
- IssuingDistributionPoint

CRL entry extensions: {SM.5::SM-R8-PKI-10}

- CertificateIssuer
- CRLReason
    - Unspecified
    - keyCompromise
    - cACompromise
    - affiliationChanged
    - superseded
    - cessationOfOperation
    - certificateHold

- InvalidityDate

Certificate Templates

PKI Services will only generate certificates that are consistent with the currently defined Certificate templates. PKI Services shipped with sample certificate templates of the most commonly requested certificate types. You can add, modify, and remove certificate templates to customize the variety of certificate types you offer to your users.

PKI Services templates support generating certificates for the following uses or with the following characteristics:

- TLS Client authentication {SM.5::SM-R8-PKI-11}.
    - key usage: digitalSignature and keyEncipherment
    - extended key usage: clientauth
- TLS Server authentication {SM.5::SM-R8-PKI-12}.
    - Key usage: digitalSignature and keyEncipherment
    - Extended key usage: serverauth
- IPSEC Firewall server {SM.5::SM-R8-PKI-13}.
    - Key usage: digitalSignature, keyEncipherment and dataEncipherment
- Certificate Authority {SM.5::SM-R8-PKI-14}.
    - Key usage: keyCertSign and CRLSign
- z/OS authentication {SM.5::SM-R8-PKI-15}.
    - Key usage: digitalSignature and keyEncipherment
    - Extended key usage: clientauth
    - Host Identity Mapping
- S/MIME email protection {SM.5::SM-R8-PKI-16}.
    - Key usage: digitalSignature and keyEncipherment
    - Subject alternate name: email
- Code signing {SM.5::SM-R8-PKI-17}.
    - Key usage: digitalSignature and docSign
    - Extended key usage: codeSigning
    - Subject alternate name: email
    - Authority Information Access: basic
- Windows logon {SM.5::SM-R8-PKI-18}.
    - Key usage: digitalSignature
    - Extended key usage: clientauth, mssmartcardlogon

- Network device using the Simple Certificate Enrollment Protocol (SCEP) {SM.5::SM-R8-PKI-19}.
- Certificate with key pair generated by PKI Services {SM.5::SM-R11-PKI-38}.
- Extended Validation Server certificate {SM.5::SM-V2R1-PKI-44}

Distribution of certificates

Other than sending the certificate back to the requestor through the browser, PKI Services can also post the issued certificates to LDAP according to the LDAP standard for communications with the Directory {SM.5::SM-R8-PKI-20}.

If the key pair for the certificate was generated by PKI Services, an email with a link embedded with the transaction ID will be sent to the requestor for him to pick up the certificate packaged with the private key {SM.5::SM-R11-PKI-39}.

Providing Certificate status

PKI Services provides certificate status information through Certificate Revocation Lists (CRLs) whose format complies with the X.509 standard and, the Online Certificate Status Protocol (OCSP) standard as defined by RFC 2560 for a “basic” OCSP responder {SM.5::SM-R8-PKI-21}.

The CRLs can be posted to LDAP according to the LDAP standard for communications with the Directory {SM.5::SM-R8-PKI-22}, or posted to an HFS file {SM.5::SM-R8-PKI-23}.

End User Functions

The end user can use the end user web pages to perform the following tasks:

- Install a CA certificate into the browser {SM.5::SM-R8-PKI-24}
- Request a new certificate {SM.5::SM-R8-PKI-25}
- Pick up a previously requested certificate {SM.5::SM-R8-PKI-26}
- Renew or revoke a previously issued browser certificate {SM.5::SM-R8-PKI-27}
- Recover a previously requested certificate and its private key , which requires specification of the email address and passphrase of the original request{SM.5::SM-R11-PKI-40}

8.4.5 Security Management (FMT)

8.4.5.1 RACF User and Group Management

Definition of users and groups

z/OS users and groups are defined in RACF using user and group profiles.

LDAP LDBM users and groups are defined in the LDAP server, but the LDAP users must be mapped one-to-one to RACF z/OS users. See LDAP LDBM Users for info on defining LDAP users.

Local Kerberos users are defined as z/OS users who also have a KERB segment in their RACF USER profile. A remote (foreign) Kerberos user may be defined locally by mapping the foreign principal name to a local z/OS (RACF) user via KERBLINK profiles. See Defining Kerberos Users for more discussion of this topic.

To create a z/OS user, a user profile for the new user has to be created in RACF. Each user profile consists of a base segment and optional segments for the use of specific subsystems. In the evaluated configuration, the base segment, the KERB segment, and the OMVS segment for the specification of attributes for z/OS UNIX System Services contain the information required by the security functions defined in this Security Target. Other segments of the user profile may exist but the effects of any values in those segments do not influence the security policy defined in this Security Target. RACF also supports a special user profile segment, CSDATA, for which the security administrator can specify the format and content of the data fields using other profiles in the CFIELD class, as well as specifying access rules in the FIELD class to determine which users can view or update data in the segment {SM.1::SM.1-R10-RACF-19}.

To create or modify a user profile, a user must have one of the following authorities:

- the SPECIAL role as a general system administrator {SM.1::SM.1.1}
- the UPDATE authority to the fields in a non-base segment of the profile he wants to modify through field-level access checking {SM.1::SM.1.2}
- to create a new user: is connected to a group that has the group-SPECIAL role and has the CLAUTH attribute for the USER class and is the owner of or has JOIN authority in the new user's default group. Note that the following roles of the ADDUSER command can not be assigned in this case: OPERATIONS, SPECIAL, and AUDITOR {SM.1::SM.1.3}
- to modify the attribute of a user: the CLAUTH attribute for the user class {SM.1::SM.1.4}. Note that only the CLAUTH and NOCLAUTH attribute can be changed {SM.1::SM.1.5}.

To list the contents of a user (user-2) profile using the LISTUSER command, a user (user-1) must have one of the following authorities:

- The SPECIAL role as a general system administrator, or the group-SPECIAL role as a group-administrator for user-2, the AUDITOR role, the group-AUDITOR role as a group auditor for user-2, or user-1 must own user-2 {SM.1::SM.1-R10-RACF-1}
- READ authority to the fields in a non-base segment of the profile he wants to list through field-level access checking {SM.1::SM.1-R10-RACF-2}
- When user-2 does not have the SPECIAL, OPERATIONS, or AUDITOR roles:
    - READ authority to FACILITY resource IRR.LISTUSER {SM.1::SM.1-R10-RACF-3}
    - READ authority to FACILITY resource IRR.LU.OWNER.owner-of-profile to allow use of LISTUSER for any non-excluded user-2 owned by "owner-of-profile" (which specifies a user ID or group name). {SM.1::SM.1-R10-RACF-4}
    - READ authority to FACILITY resource IRR.LU.TREE.owner-of-tree to allow use of LISTUSER for any non-excluded user-2 who would be in the group-SPECIAL scope of "owner-of-tree" (which specifies a user ID or group name). That is, users owned by "owner-of-tree" or owned by groups owned by "owner-of-tree" {SM.1::SM.1-R10-RACF-21}

  - To exclude a user-2 from being listed using IRR.LU.OWNER.owner-of-profile or IRR.LU.TREE.owner-of-tree authority, the administrator can define a profile that protects the resource IRR.LU.EXCLUDE.excluded-user-2 in the FACILITY class. With such a profile defined, a user also needs READ authority to it in order to gain authority via IRR.LU.OWNER.owner-of-profile or IRR.LU.TREE.owner-of-tree {SM.1::SM.1-R10-RACF-5}.

To reset the password for another user to an expired value using the PASSWORD or PHRASE commands:

- The SPECIAL role as a general system administrator, the group-SPECIAL role as a group-administrator for user-2 ,or user-1 must own user-2 {SM.1::SM.1-R10-RACF-6}.

To reset the password or password phrase for another user (user-2) or to resume user-2 using the ALTUSER command, a user (user-1) must have one of the following authorities:

- To specify a new expired or non-expired password/phrase, the SPECIAL role as a general system administrator {SM.1::SM.1-R10-RACF-7}
- To specify a new expired password/phrase, the group-SPECIAL role as a group-administrator for user-2 ,or user-1 must own user-2 {SM.1::SM.1-R10-RACF-8}
- When user-2 does not have the SPECIAL, OPERATIONS, or AUDITOR roles, or the PROTECTED attribute, one of:
  - READ authority to FACILITY resource IRR.PASSWORD.RESET to specify a new expired password/phrase when not within the minimum change window for user-2, or resume user-2 without specifying a resume date. User-1 can not set a phrase for a user-2 who does not have one already {SM.1::SM.1-R10-RACF-9}
  - UPDATE authority to FACILITY resource IRR.PASSWORD.RESET to specify a new non-expired password/phrase when not within the minimum change window for user-2, or resume user-2 without specifying a resume date. User-1 can not set a phrase for a user-2 who does not have one already {SM.1::SM.1-R10-RACF-10}.
  - CONTROL authority allows the same as UPDATE, but also allows changing the password/phrase even when within the minimum change window for user-2 {SM.1::SM.1-R10-RACF-11}.
  - READ authority to FACILITY resource IRR.PWRESET.OWNER.owner-of-profile to specify a new expired password/phrase or resume a user without specifying a resume date, for any non-excluded user-2 owned by “owner-of-profile” (which specifies a user ID or group name) {SM.1::SM.1-R10-RACF-12}
  - UPDATE authority allows the same as READ, and also allows setting a non-expired password or password phrase {SM.1::SM.1-R10-RACF-13}.
  - CONTROL authority allows the same as UPDATE, and also allows setting a new password/phrase even when within the minimum change window for user-2 {SM.1::SM.1-R10-RACF-14}.
  - READ authority to FACILITY resource IRR.PWRESET.TREE.owner-of-tree to specify a new expired password/phrase or resume a user without specifying a

resume date, for any non-excluded user-2 who would be in the group-SPECIAL scope of “owner-of-tree” (which specifies a user ID or group name). That is, users owned by “owner-of-tree” or owned by groups owned by “owner-of-tree” {SM.1::SM.1-R10-RACF-15}.

- UPDATE authority allows the same as READ, and also allows setting a non-expired password or password phrase {SM.1::SM.1-R10-RACF-16}.
- CONTROL authority allows the same as UPDATE, and also allows setting a new password/phrase even when within the minimum change window for user-2 {SM.1::SM.1-R10-RACF-17}.
- To exclude a user-2 from being altered using IRR.PWRESET.OWNER.owner-of-profile or IRR.PWRESET.TREE.owner-of-tree authority, the administrator can define a profile that protects the resource IRR.PWRESET.EXCLUDE.excluded-user-2 in the FACILITY class. With such a profile defined, a user also needs READ authority to it in order to gain authority via IRR.PWRESET.OWNER.owner-of-profile or IRR.PWRESET.TREE.owner-of-tree {SM.1::SM.1-R10-RACF-18}

RACF allows groups of users to be defined, making the management of users and user attributes and roles easier. To create a new group, a group profile must be defined in RACF. A group profile (as a user profile) consists of a base segment and (optional) other segments. As with the user profiles all group attributes related to the Security Policy as defined in this Security Target are contained in the base segment and the OMVS segment of the group profile. Each group defined in RACF must be owned by a RACF-defined user or by its superior group. Ownership of a group is assigned with the ADDGROUP command when a new group profile is created and can be changed with the ALTGROUP command used to change an existing group profile {SM.1::SM.1.6}.

RACF also supports a special group profile segment, CSDATA, for which the security administrator can specify the format and content of the data fields using other profiles in the CFIELD class, as well as specifying access rules in the FIELD class to determine which users can view or update data in the segment {SM.1::SM.1-R10-RACF-20}.

The owner of a group or a user connected to a group that has the group-SPECIAL role can:

- Define new users to RACF (provided he also has the CLAUTH attribute for the USER class) {SM.1::SM.1.7}.
- Connect and remove users from the group {SM.1::SM.1.8}.
- Delegate and change group authorities and set the default UACC for all new resources belonging to members of the group {SM.1::SM.1.9}.
- Modify, list, and delete the group profile {SM.1::SM.1.10}.
- Define, delete, and list the names of the subgroups under the group {SM.1::SM.1.11}.
- Specify the group terminal option {SM.1::SM.1.12}.

Users can be connected to a number of groups and have the group-related authorities of all the groups they are connected to {SM.1::SM.1.13}.

The OMVS segment of a group profile contains the group’s z/OS UNIX group identifier.

Management of z/OS user and group profiles occurs primarily via the RACF commands described later (ADDUSER, ALTUSER, DELUSER, LISTUSER, ADDGROUP, ALTGROUP, DELGROUP, LISTGRP). Administrators enter these commands while running in a TSO session.

Additionally, for administrative convenience, the z/OS LDAP server and RACF provide an administrative back-end to LDAP known as SDBM. RACF administrators can authenticate to LDAP using a RACF identity and password or using a digital certificates over TLS (LDAP SASL bind with EXTERNAL verification) mapped to a RACF USER ID, then make requests to the SDBM back-end via LDAP programming protocols. These requests allow the administrator to view or update RACF USER, GROUP, CONNECT and general resource profile information, and SETROPTS class-related options. LDAP transforms those requests into a tagged format supported by the R_admin() callable service, and uses that service to pass them to RACF for processing, just as though they were entered via TSO. Because the LDAP mechanisms merely provide a transformation of the administrator's LDAP request into a different format (RACF command structure), and RACF performs the authentication, and all security checking and administrative actions occur within RACF just as for the TSO commands, we do not view this LDAP mechanism as relevant to the enforcement of the SFRs. Therefore we do not address it further in this document.

The TOE also provides an interface via Java classes and methods that allows Java programs to perform RACF user and group administration in a manner similar to that used for the LDAP SDBM back-end processing. The Java program invokes the provided Java methods, which transform the provided data into RACF commands and issues them via R_admin(). RACF then processes the commands as though they were entered via TSO, using the identity of the user running the Java program {SM.1::SM.1-R9-JSEC-1}.

User profiles

The base segment of a user profile within RACF contains (among other data not relevant for the security functions defined in this Security Target) the following:

Name	Description
USERID	User's identification (a maximum of 8 characters).
NAME	User's name (not security relevant, because the user is allowed to change his name).
OWNER	Owner of the user's profile.
DFLTGRP	User's default group. (Note: A user may specify, at login time, any group he or she is connected to as the current default group. This does not change the DFLTGRP value in the profile.)
AUTHORITY	User's authority in the default group (use, create, connect, join).
PASSWORD	User's password. The user ID is DES-encrypted using the password (padded with blanks) as a key. Users who have no password and no

Name	Description
	password phrase are said to have the PROTECTED attribute, and can not logon to the system via any mechanism that uses a password, password phrase, or PassTicket.
PHRASE	Optional password phrase. Users who have a phrase must also have a password.
REVOKE	This attribute consists of a flag and a date. The date parameter specifies the date on which the user is revoked. The flag indicates that the user is revoked. The user is revoked, if either the flag is set or the actual date is after the revoke date, if defined.
RESUME	Date on which RACF lets the user have access to the system again.
UACC	Default universal access authority for resource profiles that the user defines. Only applicable to DATASET and a few general resource classes).
WHEN	Days of the week and hours of the day during which the user has access to the system (applies only to login through a terminal, not to other ports-of-entry).
CLAUTH	Classes in which the user can define profiles.
SPECIAL	Gives the user the system-wide SPECIAL attribute.
AUDITOR	Gives the user the system-wide AUDITOR attribute.
OPERATIONS	Gives the user the system-wide OPERATIONS attribute.
MODEL	Name of the data set model profile to be used when creating new data set profiles, either generic or discrete.
CERTNAME	The names of the profiles in the DIGTCERT (digital certificate) class that are related this RACF user ID.
CERTLABL	The certificate labels associated with the profiles in the DIGTCERT class that are related to this RACF user ID.

Table 25: User Profile Structure and Content

The OMVS segment in a user profile contains the following fields (among other information not relevant for the security policy as defined in this Security Target:

HOME

- User’s z/OS UNIX initial directory path name

PROGRAM

- User’s z/OS UNIX program path name, such as a default shell program

UID

- User’s z/OS UNIX user identifier

Administrators have several choices when establishing OMVS information for users:

- They may define the OMVS segment for users completely manually, via ADDUSER or ALTUSER with the OMVS keyword, and explicit specifications for the value of HOME, PROGRAM, and UID {SM.1::SM.1-R11-RACF-1}.
- They may define the OMVS segment via ADDUSER or ALTUSER with the OMVS keyword and explicit specifications for HOME and PROGRAM, but allowing RACF to automatically choose the UID via the AUTOUID keyword, in conjunction with the BPX.NEXT.USER profile in the FACILITY class, where the administrator specifies an APPLDATA field containing the allowable range of automatically-assigned UIDs. RACF will then assign the lowest available unique UID and update the APPLDATA information to indicate the UID it used {SM.1::SM.1-R11-RACF-2}.

The KERB segment in a user profile contains the following fields:

ENCRYPT

Encryption methods allowable for this user: DES, DES3 (TDES), DES with key derivation, AES128, or AES256. For this evaluation only DES3, AES128, or AES256 is allowable.

KERBNAME

The Kerberos principal ID for a locally-defined Kerberos user.

MAXTKTLFE

The maximum lifetime of a Kerberos ticket for this user.

Defining Kerberos Users

z/OS recognizes two kinds of Kerberos users: local and foreign. To define a local Kerberos user, add a KERB segment to the USER profile. Specify the encryption type as DES3 (TDES), NODES, NODESD NOAES128 NOAES256 to ensure that TDES encryption processing is used for this user. Specify the encryption type as AES128 AES256 NODES3 NODES NODESD to ensure use of AES encryption for this user. Specify the user’s Kerberos principal name. When the user next changes his/her password/phrase, the user’s encryption keys will be generated from the new RACF password/phrase {SM.1::SM.1-R10-KERB-1}.

To allow a foreign Kerberos user to authenticate, define a trust relationship between the local Kerberos realm and the foreign realm, using either the peer or transitive trust methods, by defining REALM profiles with passwords in RACF as described in the Network Authentication Service Administration guide {SM.1::SM.1-R8-KERB-2}. Kerberos passwords up to 128 characters in length may be specified in the REALM profiles {SM.1::SM.1-R10-KERB-4}. Then, for each foreign principal you want to accept, define a KERBLINK profile in RACF specifying the name of the local user in the APPLDATA field, as described in the RACF Security Administrator's Guide {SM.1::SM.1-R8-KERB-3}.

Group Profiles

RACF groups are defined by a group profile stored in the RACF database. Also group profiles have a base segment and other segments, of which only the OMVS segment is relevant for the security functionality. The base segment of a group profile within RACF contains (among other data not relevant for the security functions defined in this Security Target) the following:

Name	Description
GROUPNAME	Name of the group
OWNER	Owner of the group profile
SUPGROUP	The profile's superior group
MODEL	Name of a profile to be used as a model
TERMUACC or NOTERMUACC	The group's terminal authorization

Table 26: Group Profile Structure and Content

The OMVS segment of the group profile contains the group's z/OS UNIX group identifier in the GID field.

Administrators have several choices when establishing OMVS information for groups:

- They may define the OMVS segment for groups completely manually, via ADDGROUP or ALTGROUP with the OMVS keyword, and explicit specification of the value of the GID {SM.1::SM.1-R11-RACF-5}.
- They may define the OMVS segment via ADDGROUP or ALTGROUP with the OMVS keyword but allowing RACF to automatically choose the GID via the AUTOGID keyword, in conjunction with the BPX.NEXT.USER profile in the FACILITY class, where the administrator specifies an APPLDATA field containing the allowable range of automatically-assigned GIDs. RACF will then assign the lowest available unique GID and update the APPLDATA information to indicate the GID it used {SM.1::SM.1-R11-RACF-6}.
- They may define the OMVS information automatically, by specifying the BPX.NEXT.USER profile in the FACILITY class to record the allowable range of automatically-assigned GIDs (as above), and the BPX.UNIQUE.USER profile to indicate that whenever a user whose default (or current connect) group has no OMVS segment makes use of UNIX functions, RACF should automatically create a permanent OMVS segment for that group. {SM.1::SM.1-R11-RACF-8}. This process

will also occur if someone inquires about the GID for a group that does not have one using the getgmap() callable service {SM.1::SM.1-R11-RACF-9}.

User roles and attributes

User roles and attributes are extraordinary capabilities, restrictions, or environments that can be assigned to a user, either all of the time or when the user is connected to a specific group or groups. User attributes are stored and managed within the RACF database.

When a role or attribute is to apply only to a specific group or groups, it is specified at the group level and is called a group-related user attribute. For example, user attributes that are specified in an ADDUSER or ALTUSER command are stored in the user's profile and are in effect regardless of the group to which the user is connected {SM.1::SM.1.14}.

RACF maintains the roles and attributes specified in this section in fields in the user profile. The distinction between roles and attributes in this Security Target is artificial and reflects the definition in Chapter 5 for roles and user attributed. RACF does not make this distinction and the IBM guidance describes all of the following as user attributes.

Apart from the explicitly mentioned roles and attributes described below, users are assigned certain roles implicitly:

- Users implicitly are in the "user" role which allows them to change their own authentication data
- Users can be assigned the operator role by authorizing them to issue an operator command in the command's own profile.
- Ownership of objects entitles users to change the object's security attributes. Ownership for non-UNIX objects is identical to ownership of the profile protecting the object.

For LDAP LDBM users, the LDAP server maintains the roles and attributes specified below (in LDAP Roles and LDAP Attributes) in the LDAP LDBM or CDBM back-end, or specified via RACF resource profiles.

RACF Roles

SPECIAL and group-SPECIAL

A user who has the SPECIAL attribute at the system level can issue all RACF commands (but not all operands. There are AUDITOR-only operands related to the configuration of the audit function that only a user with the AUDITOR attribute is allowed to use) {SM.1::SM.1.15}. The SPECIAL attribute gives the user full control over all of the RACF profiles in the RACF database. The SPECIAL attribute can also be assigned at the group level. Such a user with the group-SPECIAL attribute has full control over all of the profiles within the scope of the group.

A user with the SPECIAL role in his user profile is regarded as a system administrator. He can:

- add, delete, list and modify user, group, DATASET and other profiles {SM.1::SM.1.16}
- list and define RACF general options (except options related to auditing) {SM.1::SM.1.17}

A system administrator can delegate administrative activities to users such that they can administer profiles belonging to a defined group. He does this by assigning such users the group-SPECIAL attribute. Those users then have administrative capabilities within the group they were assigned the group SPECIAL attribute {SM.1::SM.1.18}. Users with the attribute group-SPECIAL can not use general RACF options of the SETROPTS command (except for the REFRESH GENERIC and LIST operands) {SM.1::SM.1.19}.

AUDITOR and group-AUDITOR

The AUDITOR attribute is given only to users who are responsible for auditing RACF security controls and functions. To provide a check and balance on RACF security measures, the AUDITOR attribute should be given to security or group administrators other than those who have the SPECIAL attribute. The AUDITOR attribute can also be assigned at the group level. Such a user with the group-AUDITOR attribute can control the audit configuration within the scope of the group where the attribute was assigned {SM.1::SM.1.20}.

A user with the AUDITOR attribute can define and modify the audit related options in user and the auditor related options for resource profiles {SM.1::SM.1.21}. This allows him to define which activities are to be recorded in the audit trail. He can also list the content of any profile and set the system wide audit related options using the SETROPTS command. Those options are:

- AUDIT or NOAUDIT (for each profile class) {SM.1::SM.1.22}
- CMDVIOL or NOCMDVIOL {SM.1::SM.1.23}
- LOGOPTIONS (for each profile class) {SM.1::SM.1.24}
- OPERAUDIT or NOOPERAUDIT {SM.1::SM.1.25}
- SAUDIT or NOSAUDIT {SM.1::SM.1.26}

Audit configuration can also be delegated at the group level by giving the group-AUDITOR attribute to a user.

A user with the group-Auditor attribute can define and modify the audit related options in user, and resource profiles associated with his group {SM.1::SM.1.28}. He can not modify or set audit related attributes that operate system-wide {SM.1::SM.1.29}. Note that a user with SPECIAL controls the activation/deactivation of the OMVS audit related classes (DIRACC, DIRSRCH, FSOBJ, FSSEC, IPOBJ, PROCACT and PROCESS)

OPERATIONS and group-OPERATIONS

A user who has the OPERATIONS attribute has full access authorization to all RACF-protected resources in the DATASET, DASDVOL, GDASDVOL and TAPEVOL classes except when restricted by an access list entry granting less authority {SM.1::SM.1.30}. The OPERATIONS attribute can also be assigned at the group level {SM.1::SM.1.31}.

Operator Role

A user who is allowed to issue operator commands has the role of an operator. To be able to issue operator commands a user must have been authorized to the profiles in the OPERCMDS class protecting the operator commands. Permission to issue operator commands can be given on a per command basis. For the purpose of this Security Target a user who has been authorized to at least one profile in the OPERCMDS class protecting MVS and JES2 operator commands is defined to have the role of an operator.

z/OS UNIX superuser

A user operating with an effective UID of zero or a user that has been authorized to the BPX.SUPERUSER profile in the FACILITY class is defined to have the role of a z/OS UNIX superuser.

Pseudo user

A user defined with the NOPASSWORD, NOPHRASE, and NOOIDCARD parameter in his user profile is defined as having the role of a "pseudo-user". The TOE prohibits that a user with those attributes can log into the TOE. Those IDs can be used by SURROGAT-submitted batch jobs or by started procedures defined in the STARTED class or the started procedures table.

RACF Attributes

CLAUTH

If a user has the CLAUTH attribute in a class, RACF allows the user to define profiles in that class {SM.1::SM.1.32}.

Users receive the CLAUTH attribute on a class-by-class basis. The CLAUTH attribute can be assigned at the user or group level {SM.1::SM.1.33}.

A user with the CLAUTH(USER) attribute can add and modify users except for setting or modifying the following attributes:

- SPECIAL or NOSPECIAL {SM.1::SM.1.34}
- AUDITOR or NOAUDITOR {SM.1::SM.1.35}
- OPERATIONS or NOOPERATIONS {SM.1::SM.1.36}

REVOKE

A user can be prevented from entering the system by assigning the REVOKE attribute {SM.1::SM.1.37}. This attribute is useful when a user needs to be prevented from entering the system, but cannot be deleted using the DELUSER command because the user still owns RACF resource profiles. It is also useful when a user must be temporarily prevented from using the system for some reason.

User accounts can be revoked automatically after a period of inactivity {SM.1::SM.1.38}. This applies also to accounts that have never been active {SM.1::SM.1.39}.

User Revocation

User revocation can take two forms in the TOE:

- Revocation of the RACF user ID associated with a user: As all user authentication occurs via RACF, and all users have a RACF identity, the administrator can revoke a user by using the ALTUSER command with the REVOKE operand {SM.1::SM.1-R8-REV-1}. Note that this will not cover immediate revocation, but it will prevent the user from entering the system in the future.
- Revocation of a user’s digital certificate: For certificates registered in RACF via the RACDCERT command, the administrator can delete the certificate using RACDCERT {SM.1::SM.1-R8-REV-2}. This will prevent the system from recognizing that certificate in the future and associating it with the user’s RACF identity. For certificates supplied by PKI Services, the administrator can publish the certificate on the Certification

Revocation List (CRL) which will signal to applications that support CRLs or the Online Certificate Status Protocol that the certificate is no longer valid and may not be used for authentication {SM.1::SM.1-R8-REV.3}.

For immediate revocation of a user in extreme situations a simple ALTUSER or certificate revocation may not suffice. In that case the administrator may determine which applications the user has access to (e.g., TSO/E, z/OS UNIX System Services, FTP server, HTTP server, LDAP). The administrator can then issue appropriate system or application commands to determine if the user is active in the system, and if so issue the appropriate system or application commands to terminate the user's sessions.

For example, for a TSO/E user the administrator could issue the CANCEL U=user-ID command. For a batch job the administer could issue CANCEL jobname.

As a final resort the administrator could stop servers such as the HTTP server, FTP server, or LDAP server if the administrator is not sure how to locate the user's sessions on the system, as well as stopping all UNIX processing, TSO/E processing, and batch processing.

8.4.5.2 Resource management

RACF makes access decisions based on information stored in profiles or in the metadata associated with z/OS UNIX objects. RACF manages the following resource profiles:

- Data set profiles
- General resource profiles

General resource profiles apply to a number of resources defined as protected resources in this Security Target. The structure of the profiles in RACF used to protect those resources is identical, but the semantics of specific access rights is defined by the manager of the resource and may therefore differ depending on the type of resource.

Profiles consists of a base segment and optionally a set of non-base segments. Fields within non-base segments can be individually protected using the field-level access control possibilities provided by RACF.

Field-level access control allows the control of READ and UPDATE access to individual fields within a segment other than the base segment of a RACF profile. This access is based on RACF profiles in the FIELD class. Profiles in this class have the structure of profile-type.segment-name.field-name where profile-type is either the class name of a general resource profile or one of DATASET, GROUP, or USER. Different profile classes/types can have different segments and the name of the segment that contains the field for which access is controlled is specified as the second part of the profile. Different segments have different fields identified by their name and the name of the field is the third part of the profile controlling access to the field. The access control algorithm for access to general resources is used also for the FIELD class.

Fields in segments are related to operands of RACF commands or the R_admin callable service used to manage profiles and the purpose of field level access control is to provide a mechanism that allows the definition of fine-grained access control to use those command operands or list the content of individual fields. Table 18 in [RACF.SAG] provides a complete list of the segment names and the fields within each of those segments that are subject to field-level access control. The table also maps the field names to the command operands that are used to update the fields. Note that use of those operands requires to have at least

UPDATE authority to the field through field-level access control unless the user has the SPECIAL attribute {SM.4::SM.4-R12-RACFEAL5-FLA-1}. When profiles are listed, the user will only get information about the content of fields in segments protected by field-level access control, if he has at least READ access to the field, unless the user has the SPECIAL or AUDITOR attribute {SM.4::SM.4-R12-RACFEAL5-FLA-2}.

In order use field-level access control, the FIELD class needs to be active and SETROPTS RACLIST needs to be activated for the FIELD class. Otherwise only a user with the SPECIAL or AUDITOR attribute has access to fields {SM.4::SM.4-R12-RACFEAL5-FLA-3}.

When the FIELD class is active and SETROPTS RACLIST is activated for the FIELD class, users with the SPECIAL or AUDITOR attribute can list all fields regardless of the access definition in the profile protecting access to the field {SM.4::SM.4-R12-RACFEAL5-FLA-4.

To allow users to read or update fields in their own user profile protected by field-level access control a userid of &RACUID can be specified in the PERMIT command for the profiles in the FIELD class related to the fields in profiles of type USER {SM.4::SM.4-R12-RACFEAL5-FLA-5}. This does not allow this user access to those fields in the user profiles of other users {SM.4::SM.4-R12-RACFEAL5-FLA-6}.

For information on z/OS UNIX objects see z/OS UNIX file system resources.

Additionally, the LDAP server makes access decisions based on information stored in the LDBM database. For information on LDBM resources see LDAP LDBM Users.

Data set profiles

A data set profile within RACF contains (among other data not relevant for the security functions defined in this Security Target) the following:

Name	Description
Profile name	Name of the data set profile
GENERIC, MODEL, or TAPE	Indicates if it is a generic, a model or a tape data set profile
OWNER	Owner of the data set profile
NOTIFY	The TSO user who is to be notified whenever RACF uses this profile to deny access to a data set
UACC	The universal access authority for the data set or data sets protected by the profile
AUDIT	The type of auditing to be performed for the data set or data sets protected by the profile
ERASE	A setting that indicates whether the data set or data sets protected by the profile are to be erased when they are scratched

Name	Description
UNIT	The unit type on which the data set resides (for discrete profiles only)
VOLUME	The volume on which the data set resides (for discrete profiles only)

Table 27: Data Set Profile Structure and Content

Associated with those profiles is the access control list (ACL) for the profile. Each ACL entry defines the access rights of a user or a group with respect to the resource protected by the profile.

Attributes within an ACL entry are:

- access type (none, execute, read, update, control, alter)
- user IDs and group IDs allowed for the access type
- conditions of access (among other):
    - WHEN(CONSOLE(console-id ...))

      Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing commands originating from the specified system console
    - WHEN(JESINPUT(device-name ...))

      Modifies the access authority. Specifies that the identified users or groups have the specified access authority when entering the system through the specified JES input device
    - WHEN(PROGRAM(program-name...))

      Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing the specified program
    - WHEN(TERMINAL(terminal-id ...))

      Modifies the access authority. Specifies that the identified users or groups have the specified access authority when logged on to the specified terminal

Data set profiles can be created using the ADDSD command. They can be modified using the ALTDSD command and deleted using the DELDSD command. Access control lists for data set profiles can be managed using the PERMIT command. For the conditions that need to be satisfied to use those commands, see the section RACF Commands.

General resource profiles

Other protected resources defined in this Security Target (except the z/OS UNIX file system objects and z/OS UNIX IPC objects) are protected by general resource profiles that contains the resource class and the resource attributes. As with profiles for z/OS data sets, an access control list with entries defining the access types for individual users and / or groups can be defined for each such resource profile. The semantics of the individual access rights are defined by the resource manager responsible for the management of the resources protected by such a profile. Different resource classes may have different resource managers responsible for the protection and management of the resources within the class.

The structure of a general resource profile is defined in the following table (omitting fields that are not relevant for the Security Policy as defined in this Security Target:

Name	**Description**
Class name	Name of the resource class the profile belongs to
Profile name	Name of the generic resource profile
OWNER(user ID or groupname)	The owner of the profile
NOTIFY	The user who is to be notified whenever RACF uses this profile to deny access to a resource
UACC	The universal access authority for the resource or resources protected by the profile
AUDIT	The type of auditing to be performed for the resource or resources protected by the profile
FROM	The name of a profile that is to be used as a model
FCLASS	The class of the model profile
FGENERIC	A setting that indicates that the model profile name is to be treated as a generic name
FVOLUME	The volume that is to be used to locate the model profile
LEVEL	An installation-defined level
SINGLEDSN	The tape volume protected by this profile can contain only one data set (TAPEVOL class only)
TIMEZONE	The time zone in which a terminal resides (TERMINAL class only)
TVTOC	A setting that specifies that RACF is to create a tape volume table of contents (TVTOC) when a user creates the first output data set on the tape volume (TAPEVOL class only)

Name	Description
WHEN	The times when the terminal or terminals protected by the profile can be used to access the system (TERMINAL class only)

Table 28: General Resource Profile Structure and Content

General resource profiles can be created using the RDEFINE command. They can be modified using the RALTER command and deleted using the RDELETE command. Access control lists for general resource profiles can be managed using the PERMIT command. For the conditions that need to be satisfied to use those commands, see the section RACF Commands.

z/OS UNIX file system resources

z/OS UNIX file system resources are not protected by RACF profiles but by permission bits and extended attributes stored in the z/OS UNIX file system. The evaluated configuration supports two different z/OS UNIX file system types: zFS and HFS. A file system for both file system types is always implemented in a single z/OS data set.

See DAC for UNIX objects for details of the access control strategy for z/OS UNIX file system objects.

LDAP LDBM resources

The LDAP administrator can configure some LDAP resources as requiring user authentication prior to access, and others (representing public data which anyone should be able to access) as not requiring authentication.

Additionally, the LDAP server maintains the following attributes for LDBM data objects, using them in making access decisions. The TOE controls access to all directory entry objects based on the following security attributes:

- Entry Owner Information:
    - entryOwner: defines entry owner.
    - ownerPropagate: indicates whether to propagate the ownership of the entry to all descendant entries, until another entry with ownerPropagate is found.
- Access Control Attributes(ACA):
    - aclEntry: defines the access control information.
    - aclPropagate: indicates whether to propagate access control information of the entry to all descendant entries, until another entry with aclPropagate is found.

RACF General Resource classes

For the evaluation the protection of the following classes are considered:

CFIELD

Allows definition of fields in the CSDATA segment of USER and GROUP profiles {SM.1::SM.2-R10-RACF-1}

CONSOLE

Controlling access to operator consoles. Also, conditional access to other resources for commands originating from an operator console. {SM.2::SM.2.1}

CRYPTOZ

Controls access to PKCS#11 cryptographic tokens in the ICSF TKDS. {SM.2::SM.2-R9-CRYPTOZ}

DASDVOL

DASD volumes. See also the GDASDVOL class. {SM.2::SM.2.2}

DEVICES

Used to control access to unit record devices, teleprocessing or communication devices, and graphic devices. {SM.2::SM.2.3}

DIGTCERT

Used to register x.509v3 digital certificates in the RACF database.

DIGTCRIT

Used to define additional mapping criteria for the interpretation of x.509v3 digital certificates presented by clients when the certificates are not specifically registered in the RACF database, and to assign a RACF user ID to the client's session as part of the client authentication process.

DIGTNMAP

Used to define the primary mapping rules for the interpretation of x.509v3 digital certificates presented by clients when the certificates are not specifically registered in the RACF database, and to assign a RACF user ID to the client's session as part of the client authentication process.

DIGTRING

Implements key rings for servers or users in the RACF database, holding information about allowable Certificate Authority (CA) certificates and private keys for locally defined personal certificates and local signing certificates.

FACILITY

This class is used by various components of the TOE to manage specific privileges that could be assigned to users such that they do not need the SPECIAL attribute or the z/OS UNIX superuser privilege. Only a few profiles in this class are relevant for the claims in this Security Target. Access to the relevant profiles in this class is covered by individual claims for those profiles when appropriate..

GDASDVOL

Grouping class for DASDVOL {SM.2::SM.2-R8-RACF-GDASDVOL}

GLOBAL

Global access checking table entry. Provides the ability for fast access check for user that don't have the RESTRICTED attribute. Can be used for defined resource classes only. Must be used to allow READ access to resources classified as SYSLOW only. {SM.2::SM.2.5}

GTERMINL

Resource group class for TERMINAL class. {SM.2::SM.2.6}

GXFACILI

Grouping class for XFACILIT {SM.2::SM.2-R8-RACF-GXFACILI}

IDIDMAP

Class to provide mapping information from a distributed identity to a local RACF user ID {SM.2::SM.2-R12-RACF-IDIDMAP}

JESINPUT

Port of entry class to control which JES2 input devices a user can use to submit batch work to the system. {SM.2::SM.2.7}

JESJOBS

Controlling the submission and cancellation of jobs by job name. {SM.2::SM.2.8}

JESSPOOL

Controlling access to job data sets on the JES spool (that is, SYSIN and SYSOUT data sets). {SM.2::SM.2.9}

KERBLINK

Used to map user identities of local and foreign user IDs {SM.2::SM.2-R8-KERBLINK}

LOGSTRM

Used to control access to system logger resources, such as log streams and the coupling facility structures associated with them {SM.2::SM.2-R9-LOGGER-LOGSTRM}

NODES

Controls the following on MVS systems:

- Whether jobs are allowed to enter the system from other JES2 nodes {SM.2::SM.2.10}
- Whether jobs that enter the system from other nodes have to pass user identification and password verification checks associated with JES/NJE {SM.2::SM.2.11}

OPERCMDS

Controls who can issue operator commands (for example, JES and MVS, and operator commands). {SM.2::SM.2.12}

PKISERV

Used by PKI Services to provide granular administrative access controls for performing administrative actions for a given PKI Services instance and a given certificate or certificate request type {SM.2::V2R1-1}

PROGRAM

Controlled programs (load modules). {SM.2::SM.2.13}

PSFMPL

Used by PSF to perform security functions for printing, such as separator page labeling, data page labeling, and enforcement of the user printable area. {SM.2::SM.2.14}

PTKTDATA

Used to configure PassTicket processing {SM.2::SM.2-R8-PTKTDATA}

RDATALIB

Used to perform authorization checking for the R_datalib callable service {SM.2::SM.2-R9-RDATALIB}

REALM

Used to define local and foreign Kerberos realms {SM.2::SM.2-R8-REALM}

SDSF

Controls the use of authorized commands in the System Display and Search Facility (SDSF). {SM.2::SM.2.15}

SERVAUTH

Contains profiles that are used by servers to check a client's authorization to use the server or to use resources managed by the server. {SM.2::SM.2.18}

SERVER

Controlling the server's ability to register with the daemon. {SM.2::SM.2.19}

SMESSAGE

Controlling to which users a user can send messages (TSO only). {SM.2::SM.2.20}

STARTED

Used in preference to the started procedures table to assign an identity during the processing of an MVS START command. Part of the Identification of STCs. {SM.2::SM.2.21}

TAPEVOL

Tape volumes. {SM.2::SM.2.22}

TERMINAL

Terminals (TSO). {SM.2::SM.2.23}

TSOPROC

TSO logon procedures. {SM.2::SM.2.24}

UNIXPRIV

Contains profiles that are used to grant z/OS UNIX privileges. {SM.2::SM.2.25}

VTAMAPPL

Controlling who can open ACBs from non-APF authorized programs. This prevents programs from counterfeiting login screens. {SM.2::SM.2.26}

WRITER

Controlling the use of JES writers. {SM.2::SM.2.27}

XFACILIT

Analogous to the FACILITY class, but supporting longer resource and profile names (246 characters vs 39 for FACILITY) {SM.2::SM.2-R8-XFACILIT}

8.4.5.3 RACF configuration and management

Configuring RACF with the SETROPTS command

The SPECIAL and AUDITOR roles can define system wide-options of RACF with the SETROPTS command. This command can be used (among other actions) to:

- Choose the resource classes that RACF is to protect. {SM.3::SM.3.1}
- Set the universal access authority (UACC) for otherwise undefined terminals. {SM.3::SM.3.2}
- Specify logging of certain RACF commands and events. {SM.3::SM.3.3}
- Enable or disable list-of-groups access checking. {SM.3::SM.3.4}
- Display options currently in effect. {SM.3::SM.3.5}
- Enable generic profile checking for all active classes. {SM.3::SM.3.6}
- Establish password syntax rules. {SM.3::SM.3.7}
- Activate password processing for checking previous passwords, limit invalid password attempts, and warn of password expiration. {SM.3::SM.3.8}
- Control global access checking for selected individual resources or generic names with selected generalized access rules. {SM.3::SM.3.9}
- Set the passwords for authorizing use of the RVARY command. {SM.3::SM.3.10}
- Initiate refreshing of in-storage generic profile lists and global access checking tables. {SM.3::SM.3.11}
- Enable or disable shared profiles through RACLIST processing for general resources. {SM.3::SM.3.12}
- Activate auditing of access attempts to RACF-protected resources based on installation-defined security levels. {SM.3::SM.3.13}
- Activate enhanced generic naming. {SM.3::SM.3.14}
- Activate profile modeling for GDG, group, and user data sets. {SM.3::SM.3.15}

- Activate protection for data sets with single-level names. {SM.3::SM.3.16}
- Control logging of real data set names. {SM.3::SM.3.17}
- Control the job entry subsystem (JES) options implemented in RACF. {SM.3::SM.3.18}
- Activate tape data set protection. {SM.3::SM.3.19}
- Enable protection of data sets by default (PROTECTALL(FAILURES)). {SM.3::SM.3.20}
- Enable the erasure of scratched DASD data sets. {SM.3::SM.3.21}
- Activate program control. {SM.3::SM.3.22}
- Control whether a profile creator's user ID is automatically added to the profile's access list. {SM.3::SM.3.23}

Some administration activities can be delegated to user with other roles. See the definition of those roles for the administrative options that can be set or defined by those roles.

To operate in correspondence with the requirements in this Security Target, the system administrator needs to configure RACF (using the SETROPTS command) with the following options: CATDSNS(FAILURES), NOCOMPATMODE, ERASE(ALL), GENERIC(*), PROTECTALL(FAILURES), CLASSACT (TEMPDSN), JES(BATCHALLRACF) {SM.3::SM.3.24}. Additional parameter for the PASSWORD operand need to be set to define the password policy. See RACF Passwords and Password Phrases for more information.

8.4.5.4 RACF commands

The administration of RACF is performed by a set of commands. Users need the required authorities or roles to issue those commands or specific parameter of those commands. The main RACF commands are:

ADDGROUP, ALTGROUP, DELGROUP

> Commands to define a new group profile, modify an existing group profile or delete a group profile {SM.3::SM.3.25}

ADDUSER, ALTUSER, DELUSER

> Commands to define a new user profile, modify an existing user profile or delete a user profile {SM.3::SM.3.26}

ADDSD, ALTDSD, DELDSD

> Commands to define a new z/OS data set profile, modify an existing z/OS data set profile or delete an existing z/OS data set profile {SM.3::SM.3.27}

CONNECT, REMOVE

> Command to connect a user to or remove a user from a group {SM.3::SM.3.28}

LISTGROUP, LISTUSER, LISTDSD

> Commands to list user, group or z/OS data set profiles {SM.3::SM.3.29}

RDEFINE, RALTER, RDELETE

> Commands to define, modify or delete a general resource profile {SM.3::SM.3.30}

RACF will provide an ENF 79 signal when an administrator has issued a RDEFINE or a RALTER or a RDELETE command that could change a user's or group's authorization to resources in a resource class that has been defined in the RACF Class Descriptor Table with the SIGNAL=YES option {SM.3::SM.3-V2R1-1}.

RLIST

Command to list a general resource profile {SM.3::SM.3.31}

PASSWORD

Command to specify a user’s password {SM.3::SM.3.32}

PHRASE

Command to specify a user’s password phrase {SM.3::SM.3-R10-RACF-1}

PERMIT

Command to maintain the access list of a resource profile {SM.3::SM.3.33}

RACDCERT

Command to maintain x.509v3 digital certificates, certificate mapping filters, certificate mapping criteria, and key rings in the RACF database.

RACMAP

Command to establish mappings between distributed user identities and local RACF user IDs {SM.3::SM.3-R12-RACF-2}.

SETROPTS

Command to set specific RACF options (see section above for details) {SM.3::SM.3.34}

There are the following options how a RACF command can be issued:

- as a TSO command
- as an operator command
- with command direction (command can be directed to run under the authority of a user ID on a remote node, or the same node using the AT operand. Use of this operand is usually restricted as indicated in the table below). [Note: This method is part of the RACF Remote Sharing Facility (RRSF) which is not a part of this evaluation, and will not be considered further.]
- with automatic command direction (activated using the SET AUTODIRECT command. Use is restricted by RACF profiles as for command direction)[Note: This method is part of the RACF Remote Sharing Facility (RRSF) which is not a part of this evaluation, and will not be considered further.]
- from the RACF parameter library
- via the R_admin callable service

Not all commands can be issued in all options. For example some commands can not be used as TSO commands, and some can not be used as operator commands.

{SM.3::SM.3-R12-RACFEAL5-30} For commands (except RVARY) that can be issued as either a TSO command or as an operator command, the following security requirements apply in addition to any specified by the commands themselves:

- The command issuer must be logged on to the console.
- If the OPERCMDS class is active and SETR RACLISTed then the command issuer must have READ authority to the OPERCMDS profile protecting the command, if one exists:

Command Name	**OPERCMDS Resource name**
ADDGROUP	Subsystem-name.ADDGROUP (Note: for all the RACF TSO commands, when issued as an operator command the OPERCMDS resource name uses the full command name, even if the user issued the command using an abbreviation such as AG)
ADDSD	Subsystem-name.ADDSD
ADDUSER	Subsystem-name.ADDUSER
ALTDSD	Subsystem-name.ALTDSD
ALTGROUP	Subsystem-name.ALTGROUP
ALTUSER	Subsystem-name.ALTUSER
CONNECT	Subsystem-name.CONNECT
DELDSD	Subsystem-name.DELDSD
DELGROUP	Subsystem-name.DELGROUP
DELUSER	Subsystem-name.DELUSER
LISTDSD	Subsystem-name.LISTDSD
LISTGRP	Subsystem-name.LISTGRP
LISTUSER	Subsystem-name.LISTUSER
PASSWORD	Subsystem-name.PASSWORD
PHRASE	Subsystem-name.PASSWORD
PERMIT	Subsystem-name.PERMIT
RALTER	Subsystem-name.RALTER
RDEFINE	Subsystem-name.RDEFINE
RDELETE	Subsystem-name.RDELETE
REMOVE	Subsystem-name.REMOVE
RLIST	Subsystem-name.RLIST
SEARCH	Subsystem-name.SEARCH
SETROPTS	Subsystem-name.SETROPTS (Note: READ authority is required to issue SETROPTS LIST, but UPDATE is required if any other operands are specified.)

Table 29: Profiles protecting RACF Operator Commands

{SM.3::SM.3-R12-RACFEAL5-31} For commands that can be issued as operator commands but not as TSO commands, the following security requirements apply:

- If you are logged on to the console, and the command is protected by an OPERCMDS profile, then you must have READ authority to that profile:

Command Name	**OPERCMDS Resource name**
DISPLAY	Subsystem-name.DISPLAY.SIGNON Note: No OPERCMDS authority check is performed when this command is issued from a RACF parameter library member.
RESTART	Subsystem-name.RESTART
SET	*subsystem-name*.SET.AUTOAPPL *subsystem-name*.SET.AUTODIRECT *subsystem-name*.SET.AUTOPWD *subsystem-name*.SET.INCLUDE *subsystem-name*.SET.JESNODE *subsystem-name*.SET.LIST *subsystem-name*.SET.PWSYNC *subsystem-name*.SET.TRACE Note: No OPERCMDS authority check is performed when this command is issued from a RACF parameter library member.
SIGNOFF	Subsystem-name.SIGNOFF Note: No OPERCMDS authority check is performed when this command is issued from a RACF parameter library member.
STOP	Subsystem-name.STOP
TARGET	*subsystem-name*.TARGET.DESCRIPTION *subsystem-name*.TARGET.LIST *subsystem-name*.TARGET.LOCAL *subsystem-name*.TARGET.NODE *subsystem-name*.TARGET.OPERATIVE **Note:** TARGET.OPERATIVE also protects the DELETE and DORMANT operands. *subsystem-name*.TARGET.PREFIX *subsystem-name*.TARGET.PROTOCOL *subsystem-name*.TARGET.PURGE *subsystem-name*.TARGET.WDSQUAL *subsystem-name*.TARGET.WORKSPACE Note: No OPERCMDS authority check is performed when this command is issued from a RACF parameter library member.

Table 30: Profiles protecting Commands that are not TSO Commands

If you are not logged on to the console, or the command is not protected by an OPERCMDS profile, then (except for the DISPLAY command) you must issue the command from a console with MASTER or SYSTEM authority.

Administrators can also use the LDAP SDBM backend {SM.3::SM.3-R9-LDAP-1} or the Java JSEC interfaces {SM.3::SM.3-R9-JSEC-1} to issue the RACF commands ADDUSER, ALTUSER, DELUSER, LISTUSER, ADDGROUP, ALTGROUP, DELGROUP, LISTGRP, CONNECT, and REMOVE. Additionally, administrators can use the LDAP SDBM backend to issue RDEFINE, RDELETE, RALTER, RLIST, and SETROPTS commands for class-related options {SM.3::SM.3-R11-LDAP-2}.

The following table defines the required to use the individual RACF commands and command options/parameter:

Command	Authorities required for use
ADDGROUP	{SM.3::SM.3-R12-RACFEAL5-1} General: • SPECIAL or • group-SPECIAL and the superior group is within the scope of group-SPECIAL authority, or • owner of the superior group, or • JOIN authority to the superior group Special parameter authorization: • SPECIAL or UPDATE authority via field-level access control to add segments to a group’s profile • For use of the SHARED keyword: SPECIAL or READ authority to the SHARED.IDS resource in the UNIXPRIV class
ADDSD	{SM.3::SM.3-R12-RACFEAL5-2} The level of authority required to use the ADDSD command and the types of profiles the caller can define are: To protect a user data set with RACF, one of the following must be true: • The high-level qualifier of the data set name (or the qualifier supplied by the RACF naming conventions table) must match the caller’s user ID, or • SPECIAL, or • The user ID for the data set profile must be within the scope of a group in which the caller has the group-SPECIAL attribute. To protect a group data set with RACF, one of the following must be true: • CREATE authority in the group, or • SPECIAL, or • OPERATIONS attribute and not be connected to the group, or • The data set profile must be within the scope of the group in which the caller has the group-SPECIAL attribute.

Command	Authorities required for use
	• The data set profile must be within the scope of the group in which the caller has the group-OPERATIONS attribute, and he must not be connected to the group. Special parameter authorization: • • To add non-base segments, SPECIAL or UPDATE authority via field-level access control. • To add a discrete profile for a VSAM data set already RACF-protected by a generic profile, you must have ALTER access authority to the catalog or to the data set through the generic profile. • To specify a model data set profile (using, as required, FROM, FCLASS, FGENERIC, and FVOLUME), you must have sufficient authority over the model profile (the *from* profile). RACF makes the following checks until one of the conditions is met: • You have the SPECIAL attribute. • The *from* profile is within the scope of a group in which you have the group-SPECIAL attribute. • You are the owner of the *from* profile. • The high-level qualifier of the profile name (or the qualifier supplied by the RACF naming conventions table) is your user ID. • For a discrete profile, you are on the access list in the *from* profile with ALTER authority. (If you have any lower level of authority, you cannot use the profile as a model.) • For a discrete profile, your current connect group (or, if list-of-groups checking is active, any group to which you are connected) is on the access list in the *from* profile with ALTER authority. • For a discrete profile, the UACC is ALTER.
ADDUSER	{SM.3::SM.3-R12-RACFEAL5-3} General: • SPECIAL or • CLAUTH for the USER class and one of the following is true: ○ The caller is the owner of the default group specified ○ The caller has JOIN authority in the default group specified ○ The default group specified is within the scope of a group where the caller has group-SPECIAL Special parameter authorization:

Command	Authorities required for use
	• SPECIAL to give the new user the OPERATIONS, SPECIAL or AUDITOR attribute • SPECIAL to assign a security category to a profile or the category must be in the caller's user profile • SPECIAL to assign a security level to a profile or the security level in the caller's profile is equal or greater than the security level assigned to the new user • When defining information within a segment other than the base segment: ◦ SPECIAL, or ◦ UPDATE authority to the desired field through field-level access control • For use of the SHARED keyword: SPECIAL or READ authority to the SHARED.IDS resource in the UNIXPRIV class
ALTDSD	{SM.3::SM.3-R12-RACFEAL5-4} The level of authority required to use the ALTDSD command and the types of profiles the caller can define are: To protect a user data set with RACF, one of the following must be true: • The high-level qualifier of the data set name (or the qualifier supplied by the RACF naming conventions table) must match the caller's user ID, or • SPECIAL, or • The user ID for the data set profile must be within the scope of a group in which the caller has the group-SPECIAL attribute. • The caller is the owner of the profile • For a discrete profile: the caller has ALTER authority to the profile or the caller's current connect group (or, if list-of-groups checking is active) has ALTER authority Special parameter authorization: • For use of the GLOBALAUDIT keyword: AUDITOR or the data set profile is within the scope of a group in which the caller has the group-AUDITOR attribute • For access to the DFP or TME segment: field-level access authority to those segments
ALTGROUP	{SM.3::SM.3-R12-RACFEAL5-5} General: • To issue the command,, except as specified below: • SPECIAL • The group profile is within the scope of a group in which you have the group-SPECIAL attribute • You are the current owner of the group. Special parameter authorization:

Command	Authorities required for use
	• To change the superior group of a group: • SPECIAL or • group-SPECIAL and the current and new superior group is within the scope of group-SPECIAL authority, or • owner of the current and new superior group, or • JOIN authority to the current and new superior group • A combination of group-SPECIAL, owner or JOIN authority where one of those applies for the current and one of those applies for the new superior group • SPECIAL or permission through field-level access checking in order to add, delete, or alter segments of a group's profile • For use of the SHARED keyword: SPECIAL or READ authority to the SHARED.IDS resource in the UNIXPRIV class
ALTUSER	{SM.3::SM.3-R12-RACFEAL5-6} General: • SPECIAL (allows use of all operands except UAUDIT/NOUAUDIT) Special parameter authorization: • If the owner of the user profile is within the scope of a group in which the group-SPECIAL attribute, he can use all of the operands except SPECIAL, AUDITOR, OPERATIONS, NOEXPIRED and UAUDIT/NOUAUDIT. • The owner of the user's profile can use any of the following operands for user-related attributes: ○ ADSP | NOADSP ○ DATA | NODATA ○ DFLTGRP ○ GRPACC | NOGRPACC ○ MODEL | NOMODEL ○ NAME ○ OIDCARD | NOOIDCARD ○ OWNER ○ PASSWORD | NOPASSWORD ○ PHRASE | NOPHRASE ○ RESTRICTED | NORESTRICTED ○ RESUME | NORESUME ○ REVOKE | NOREVOKE ○ WHEN • Each user can change his or her name field, default group or model data set profile name (using the NAME, DFLTGRP, or MODEL operand, respectively).

Command	Authorities required for use
	• You can use the GROUP, AUTHORITY, and UACC operands for group-related user attributes if you have JOIN or CONNECT authority, if the group profile is within the scope of a group in which you have the group-SPECIAL attribute, or if you are the owner of the specified group. • To specify the AUDITOR/NOAUDITOR, SPECIAL/NOSPECIAL, and OPERATIONS/NOOPERATIONS operands as system-wide user attributes, the caller must have the SPECIAL attribute. • To specify the UAUDIT/NOUAUDIT operand, either the caller must have the AUDITOR attribute, or the user profile must be within the scope of a group in which the caller has the group-AUDITOR attribute. • The CLAUTH and NOCLAUTH operands can be specified if the caller is the owner of the user’s profile and has the CLAUTH attribute for the class to be added or deleted. • • To change information within a segment other than the base segment, the caller must have one of the following: ○ The SPECIAL attribute ○ At least UPDATE authority to the desired field within the segment through field-level access control. • To reset passwords and password phrases or to resume user IDs, the caller must have at least one of the following authorizations: ○ SPECIAL. ○ group-SPECIAL authority over the user profile. ○ The caller is the OWNER of the user profile. ○ The caller has sufficient access to the IRR.PASSWORD.RESET resource in the FACILITY class. ○ The caller has sufficient access to an appropriate resource in the FACILITY class (IRR.PWRESET.OWNER.owner or IRR.PWRESET.TREE.owner), and both of the following conditions are also true: ▪ The other user does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. ▪ The caller is not excluded from altering the user by the IRR.PWRESET.EXCLUDE.excluded-user resource in the FACILITY class. • When the caller’s reset and resume authority is through access to the IRR.PASSWORD.RESET resource, the IRR.PWRESET.OWNER.owner resource, or the IRR.PWRESET.TREE.owner resource, the following requirements apply: ○ If the caller has READ access, he can: ▪ Use the PASSWORD operand to reset a password

Command	Authorities required for use
	(to an expired password) for a user who does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. ▪ Use the PHRASE operand to reset a password phrase (to an expired password phrase) for a user with an assigned password phrase who does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. Note: The caller cannot use the PHRASE operand to add a password phrase for a user who does not have one. ▪ Use the RESUME operand, without specifying a date, for a user who does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. ○ If the caller has UPDATE access, he can: ▪ Use the PASSWORD, PHRASE, and RESUME operands as noted for READ access. ▪ Use the NOEXPIRED operand (with PASSWORD or PHRASE) for a user who does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. ○ If the caller has CONTROL access, he can: ▪ Use the PASSWORD, PHRASE, RESUME, and NOEXPIRED operands as noted for READ and UPDATE access. ▪ Reset the password or password phrase within the minimum change interval for a user who does not have the SPECIAL, OPERATIONS, AUDITOR, or PROTECTED attribute. • For use of the SHARED keyword: SPECIAL or READ authority to the SHARED.IDS resource in the UNIXPRIV class
CONNECT	{SM.3::SM.3-R12-RACFEAL5-7} General: • SPECIAL, or • Group-SPECIAL in the group • Ownership of the group • JOIN authority in the group • CONNECT authority in the group • A caller can not give a higher level of authority in the group than he has himself. •
DELDSD	{SM.3::SM.3-R12-RACFEAL5-8} General: • SPECIAL, or • The data set profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • The high-level qualifier of the profile name (or the qualifier supplied by the naming convention table) is the caller’s user ID, or

Command	Authorities required for use
	• The caller is the owner of the profile • For a discrete profile: the caller is on the access list with ALTER authority, or • For a discrete profile, the caller’s group or one of the caller’s groups (if checking list of groups is active) is on the access list and has ALTER authority. • For a discrete profile, the universal access authority is ALTER.
DELGROUP	{SM.3::SM.3-R12-RACFEAL5-9} General: • SPECIAL, or • The group to be deleted is within the scope of a group where the caller has the group-SPECIAL in the group, or • The caller is the Owner of the group to be deleted, or • The caller is the Owner of a superior group of the group to be deleted • JOIN authority in the superior group
DELUSER	{SM.3::SM.3-R12-RACFEAL5-10} General: • SPECIAL, or • Group-SPECIAL for the group where the user profile to be deleted is within the scope of the group, or • Ownership of the user profile Other: • A user profile that has a distributed identity filter associated with it can only be deleted after the distributed identity filter has been deleted.
DISPLAY	{SM.3::SM.3-R12-RACFEAL5-25} None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).
LISTDSD	{SM.3::SM.3-R12-RACFEAL5-11} For listing the RACF segment of a data set profile: • SPECIAL, or • The data set profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • OPERATIONS, or • The data set profile is within the scope of a group in which the caller has the group-OPERATIONS attribute, or • AUDITOR, or • The data set profile is within the scope of a group in which the caller has the group-AUDITOR attribute, or • The high-level qualifier of the profile name (or the qualifier supplied by the naming convention table) is the caller’s user ID, or • The caller is the owner of the profile • The caller is on the access list with READ authority, or • The caller’s group or one of the caller’s groups (if checking list of groups is active) is on the access list and

Command	Authorities required for use
	has READ authority, or • The universal access authority is READ, or • The caller has READ access for the profile name from the GLOBAL ENTRY TABLE (if the table contains an entry for the profile) For listing the DFP or TME segment of a data set profile: • SPECIAL, or • AUDITOR, or • READ authority to the desired field within the segment through field-level access control. Special parameter authorization: • To specify the AUTHUSER operand to display the access list for a profile, one of the following conditions must be met for each profile to be listed: ○ SPECIAL, or ○ The data set profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or ○ OPERATIONS, or ○ The data set profile is within the scope of a group in which the caller has the group-OPERATIONS attribute, or ○ AUDITOR, or ○ The data set profile is within the scope of a group in which the caller has the group-AUDITOR attribute, or ○ The high-level qualifier of the profile name (or the qualifier supplied by the naming convention table) is the caller's user ID, or ○ The caller is the owner of the profile ○ The caller has ALTER access for the profile name from the GLOBAL ENTRY TABLE (if this table contains an entry for the profile). ○ For a discrete profile, the caller is on the profile's access list with ALTER authority. ○ For a discrete profile, the caller's current connect group (or, if list-of-groups checking is active, any group to which the caller is connected) is in the access list and has ALTER authority. ○ For a discrete profile, the universal access authority is ALTER. Other: • To display the type of access attempts (as specified by the GLOBALAUDIT operand on the ALTDSD command) that are being logged on the SMF data set, either the caller must have the AUDITOR attribute or the profile must be within the scope of a group in which the caller has the group-AUDITOR attribute.

Command	Authorities required for use
LISTGRP	{SM.3::SM.3-R12-RACFEAL5-12} For listing the RACF segment of a group profile: • SPECIAL, or • AUDITOR, or • For each group to be listed at least one of the following is true: ○ The caller has group-SPECIAL the group or the group profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or ○ The caller has group-AUDITOR for the group or the group profile is within the scope of a group in which the caller has the group-AUDITOR attribute, or ○ The caller is the owner of the group, or ○ The caller has JOIN or CONNECT authority for the group For listing the other segments of a group profile: • SPECIAL, or • AUDITOR, or • READ access to the desired field through field-level access control
LISTUSER	{SM.3::SM.3-R12-RACFEAL5-13} For listing the RACF segment of a user profile: • SPECIAL, or • AUDITOR, or • For each user to be listed at least one of the following is true: ○ the user profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or ○ the user profile is within the scope of a group in which the caller has the group-AUDITOR attribute, or ○ The caller is the owner of the user profile, or ○ The caller has READ access to the IRR.LISTUSER resource in the FACILITY class and the user does not have the SPECIAL, AUDITOR, or OPERATIONS attribute. ○ The caller has READ access to an appropriate resource (IRR.LU.OWNER.owner or IRR.LU.TREE.owner) in the FACILITY class, and both of the following conditions are also true: ▪ The user does not have the SPECIAL, AUDITOR, or OPERATIONS attribute. ▪ The caller is not excluded from listing the user by the IRR.LU.EXCLUDE.excluded-user resource in the FACILITY class.

Command	**Authorities required for use**
	For listing other segments of a user profile: • SPECIAL, or • AUDITOR, or • READ authority to the desired field within the segment through field-level access checking Other: • • The value of the UAUDIT/NOUAUDIT operand is only listed if the authorization is given by the AUDITOR or group-AUDITOR authorities.
PASSWORD	{SM.3::SM.3-R12-RACFEAL5-14} General: • To change a user's password or password phrase or change interval: ○ The caller is the user himself, or ○ SPECIAL, or ○ the user's profile is within the scope of a group in which the caller has group-SPECIAL • To reset another user's password to the default value: ○ The caller is the owner of the user's profile ○ SPECIAL, or ○ the user's profile is within the scope of a group in which the caller has group-SPECIAL
PERMIT	{SM.3::SM.3-R12-RACFEAL5-15} General: • SPECIAL, or • The profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • The caller is the owner of the resource, or • For a discrete profile: the caller has ALTER authority (either via the standard access list, or via group access or via UACC) • For profiles in the DATASET class: the high-level qualifier of the profile name (or the qualifier supplied by the RACF naming conventions table) is the caller's user ID. • For profiles in the FILE or DIRECTRY class: the second qualifier of the profile name is the caller's user ID.
PHRASE	See PASSWORD
RACDCERT	See separate table
RACMAP	{SM.3::SM.3-R13-RACFEAL5-16} General: • SPECIAL, or • Sufficient authority to the IRR.IDIDMAP.*function* resource in the FACILITY class, where *function* is MAP, DELMAP, QUERY, or LISTMAP. READ authority is required for the caller to create, delete, or list a distributed identity filter

Command	Authorities required for use
	for his own RACF user ID. UPDATE authority is required for the caller to create, delete, or list a distributed identity filter for another RACF user ID.
RACPRIV	{SM.3::SM.3-R12-RACFEAL5-17} General: • READ access to the profile IRR.WRITEDOWN.BYUSER in the FACILITY class.
RALTER	{SM.3::SM.3-R12-RACFEAL5-18} General (with the exception of the GLOBALAUDIT parameter): • SPECIAL, or • The profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • The caller is the owner of the profile, or • For a discrete profile: the caller has ALTER authority (either via the standard access list, or via group access or via UACC) • For profiles in the FILE or DIRECTRY class: the second qualifier of the profile name is the caller’s user ID. (Note: the FILE and DIRECTRY classes apply only in z/VM systems, and will not be considered in this evaluation even though the classes exist in RACF for z/OS.) • Special parameter authorization: • • For modifying information in segments other than the base segment: ○ SPECIAL, or ○ UPDATE Access allowed by field-level access control for the fields to be modified • For use of the GLOBALAUDIT parameter: ○ AUDITOR, or ○ The profile is within the scope of a group for which the caller has the group-AUDITOR attribute. • For use of the ADDMEM parameter: ○ For classes other than PROGRAM, GLOBAL, RACFVARS, and NODES, if the member resources are already RACF-protected by a member class profile or as a member of a profile in the same grouping class, one of the following must be true: ▪ The caller has ALTER access authority to the member, or ▪ The caller is the owner of the member resource, or ▪ The member resource is within the scope of a group in which the caller has the group-SPECIAL attribute, or ▪ The caller has the SPECIAL attribute. ○ For classes other than PROGRAM, GLOBAL, RACFVARS,

Command	Authorities required for use
	and NODES, if the member resources are not RACF-protected (that is, there is no profile defined for that member), one of the following must be true: ▪ The caller has CLAUTH authority to define resources in the member resource class. ▪ The caller has the SPECIAL attribute. ○ To add a member to a profile in the RACFVARS or NODES class, one of the following must be true: ▪ The caller has CLAUTH authority to define resources in the specified class ▪ The caller has the SPECIAL attribute. ▪ The caller is the owner of the profile indicated by profile-name. ▪ The caller has ALTER access authority to the profile indicated by profile-name. ○ To add a member to a profile in the PROGRAM class, one of the following must be true: ▪ You have CLAUTH authority to define resources in the specified class (for example, PROGRAM). ▪ You have the SPECIAL attribute. ○ To add a member to a profile in the GLOBAL class (other than the GLOBAL DATASET, GLOBAL DIRECTRY, or GLOBAL FILE profile) ▪ If the profile resource-name is already RACF-protected by a profile in class class-name: • ALTER access authority to the profile resource-name in class class-name, or • The caller is the OWNER of the profile resource-name, or • The profile resource-name in class class-name is within the scope of a group in which the caller has the group-special attribute, or • The caller has the SPECIAL attribute. ▪ If the profile resource-name is not already RACF-protected (that is, there is no profile defined for that member in class class-name): • The caller has CLAUTH authority to define resources in the class class-name, or • The caller has the SPECIAL attribute. ○ To add a member to the GLOBAL DATASET profile, one of the following must be true: ▪ The caller is the owner of the DATASET profile in the GLOBAL class. ▪ The member is within the scope of a group in which the caller has the group-SPECIAL attribute, or the high-level qualifier of the member name is the caller's user ID.

Command	Authorities required for use
	▪ The caller has the SPECIAL attribute. ○ To add a member to the GLOBAL DIRECTRY or GLOBAL FILE profile, the caller must have the SPECIAL attribute. • For use of the DELMEM operand: ○ If class-name is specified as GLOBAL or PROGRAM, the rules for member are the same as given for ADDMEM. • For use of the ADDVOL operand: ○ SPECIAL, or ○ CLAUTH attribute for the TAPEVOL resource class in addition to the other authorization requirements for using the RALTER command. • For use of the GLOBALAUDIT operand: ○ AUDITOR, or ○ The resource profile must be within the scope of a group in which the caller has the group-AUDITOR attribute.
RDEFINE	{SM.3::SM.3-R12-RACFEAL5-19} General (with the exception of the GLOBALAUDIT parameter): • SPECIAL, or • If the caller has CLAUTH authority for the GLOBAL class, and group-SPECIAL authority in a group, he can add members whose high-level qualifier is the group name or a user ID in the scope of the group. This applies only to classes that are sensitive to high-level qualifiers, such as DATASET. • If the name to be defined is not already defined to RACF as a member of a resource group and the caller is defining a profile in a normal (non-member, non-grouping) class, a member class, or a member of a grouping class, he must have CLAUTH authority for the specified class. • If the resource to be defined is a discrete name already defined to RACF as a member of a resource group, the caller can define it as a resource to RACF if he has ALTER authority, or if the resource group profile is within the scope of a group in which he has the group-SPECIAL attribute, or if he is the owner of the resource group profile. If authority conflicts arise because the resource is a member of more than one group and the user's authority in those groups differs, RACF resolves the conflict by using the least restrictive authority. • If the caller defines a profile in the FILE or DIRECTRY class, one of the following must be true: ○ The second qualifier of the profile name must match the caller's user ID ○ The caller has the SPECIAL attribute ○ The profile name must be within the scope of a group

Command	Authorities required for use
	in which the caller has the group-SPECIAL attribute. (Note: the FILE and DIRECTRY classes apply only in z/VM systems, and will not be considered in this evaluation even though the classes exist in RACF for z/OS.) • If the caller does not have the SPECIAL attribute and the SETROPTS GENERICOWNER option is in effect, and if an existing generic profile protects the profile name the caller is defining, he needs to own the less specific profile. GENERICOWNER does not apply to the PROGRAM class. • • To define segments other than the base segment, the caller must have the SPECIAL attribute or he must be permitted to do so through field-level access checking. Other: • • For Model profiles: To specify a model profile (using, as required, FROM, FCLASS, FGENERIC, and FVOLUME), the caller must have sufficient authority over the model profile (the from profile). RACF makes the following checks until one of the conditions is met: ◦ The caller has the SPECIAL attribute. ◦ The from profile is within the scope of a group in which the caller has the group-SPECIAL attribute. ◦ The caller the owner of the from profile. ◦ If the FCLASS operand is DATASET, the high-level qualifier of the profile name (or the qualifier supplied by the RACF naming conventions table) is the caller's user ID. ◦ For a discrete profile, the caller is on the access list in the from profile with ALTER authority. ◦ For a discrete profile, the caller's current connect group (or, if list-of-groups checking is active, any group to which the caller is connected) is in the access list in the from profile with ALTER authority. ◦ For a discrete profile, the universal access authority (UACC) is ALTER. Special parameter authorization: • For use of the ADDMEM operand: In addition to the authority needed to issue the RDEFINE command, the caller needs one of the following authorities to add members using the RDEFINE command: ◦ See the requirements specified for the use of the ADDMEM operand for the RALTER command.
RDELETE	{SM.3::SM.3-R12-RACFEAL5-20} General:

Command	Authorities required for use
	• SPECIAL, or • The profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • The caller is the owner of the profile, or • For a discrete profile: the caller has ALTER authority (either via the standard access list, or via group access or via UACC) • For profiles in the FILE or DIRECTRY class: the second qualifier of the profile name is the caller’s user ID.
REMOVE	{SM.3::SM.3-R12-RACFEAL5-21} General: • SPECIAL, or • The profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • The caller is the owner of the group, or • The caller has JOIN or CONNECT authority in the group.
RESTART	None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).
RLIST	{SM.3::SM.3-R12-RACFEAL5-22} General: • SPECIAL, or • The resource profile is within the scope of a group in which the caller has the group-SPECIAL attribute, or • OPERATIONS, or • The resource profile is within the scope of a group in which the caller has the group-OPERATIONS attribute, or • AUDITOR, or • The resource profile is within the scope of a group in which the caller has the group-AUDITOR attribute, or • The caller is the owner of the resource. • If the profile is in the FILE or DIRECTRY class and the second qualifier of the profile name is the caller’s user ID. (Note: the FILE and DIRECTRY classes apply only in z/VM systems, and will not be considered in this evaluation even though the classes exist in RACF for z/OS.) • To list the contents of segments other than the base segment, the caller must have the SPECIAL or AUDITOR attribute or the installation must permit the caller to do so through field-level access checking. • If the caller is on the access list for the resource and has at least READ authority. If you specify ALL, RACF lists only information pertinent to the caller’s user ID. • If the caller’s current connect group (or, if list-of-groups checking is active, any group to which the caller is connected) is in the access list and has at least READ

Command	Authorities required for use
	authority. • The universal access authority of the resource is at least READ. • The caller has at least read access for the profile name from the GLOBAL ENTRY TABLE (if this table contains an entry for the profile). Other: • The caller will see the type of access attempts, as specified by the GLOBALAUDIT operand, only if he has the AUDITOR attribute or if the resource profile is within the scope of a group in which he has the group-AUDITOR attribute.
RVARY	General: none {SM.3::SM.3-R12-RACFEAL5-32} Other: • No special authority is needed to issue the RVARY command. However, the operator (at the operator console or security console) must approve a change in RACF status or the RACF data sets - or a change in the operational mode if RACF is enabled for sysplex communication - before RACF allows the command to complete. • If the RVARY command changes RACF or database status (ACTIVE/INACTIVE), RACF issues an informational message and the operator is required to enter the password defined by RVARYPW STATUS(*status-pw*) to authorize the change. If the RVARY command switches the RACF data sets (SWITCH) or changes the RACF operating mode (DATASHARE/NODATASHARE), RACF issues an informational message and the operator is required to enter the password defined by RVARYPW SWITCH(*switch-pw*). When RVARY is issued as a RACF operator command from a console with master authority, the default password YES is also accepted for RVARY ACTIVE, RVARY NODATASHARE or RVARY SWITCH commands.
SEARCH	{SM.3::SM.3-R12-RACFEAL5-23} General: • SPECIAL, or • AUDITOR, or • The profile is within the scope of a group in which the caller has the group-SPECIAL or group-AUDITOR attribute, or • If the profile is for a DASD data set, the high-level qualifier of the data set name (or the qualifier supplied by the naming convention table) is the caller's user ID, or • If the profile is in the FILE or DIRECTRY class, the second qualifier of the profile name is the caller's user ID (note: the FILE and DIRECTRY classes apply only in z/VM

Command	Authorities required for use
	systems, and will not be considered in this evaluation even though the classes exist in RACF for z/OS.), or • The caller is on the access list for the profile and has at least READ authority, or • The caller’s current connect group (or, if list-of-groups checking is active, any group to which the caller is connected) is on the access list and has at least READ authority, or • The caller has the OPERATIONS attribute, or the profile is within the scope of a group in which the caller has the group-OPERATIONS attribute, and the class is DATASET or a general resource class that specifies OPER=YES in the static class descriptor table or OPERATIONS(YES) in the dynamic class descriptor table, or • The universal access authority is at least READ (or GLOBAL when listing discrete profiles). Special parameter authorization: • In order to use the USER operand, one of the following must be true: ○ SPECIAL, or ○ AUDITOR, or ○ The caller is the owner of the user profile, or ○ The parameter entered for the USER operand is the caller’s user ID ○ The caller has group-SPECIAL or group-AUDITOR authority in a group that owns the user profile. Other: • No authorization is required to the user or group profiles that are listed when the UID or GID keyword is specified.
SET	{SM.3::SM.3-R12-RACFEAL5-26} None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).
SETROPTS	{SM.3::SM.3-R12-RACFEAL5-24} General: • SPECIAL, for all operands except: ○ APPLAUDIT | NOAPPLAUDIT ○ AUDIT | NOAUDIT ○ CMDVIOL | NOCMDVIOL ○ LOGOPTIONS ○ OPERAUDIT | NOOPERAUDIT ○ SAUDIT | NOSAUDIT ○ • AUDITOR, for use of the following operands: ○ APPLAUDIT | NOAPPLAUDIT ○ AUDIT | NOAUDIT ○ CMDVIOL | NOCMDVIOL ○ LOGOPTIONS

Command	Authorities required for use
	◦ OPERAUDIT | NOOPERAUDIT ◦ SAUDIT | NOSAUDIT ◦ Special parameter authorization: • For using the LIST operand: SPECIAL or AUDITOR Other: In some situations, a caller can use SETROPTS even if he does not have the SPECIAL or AUDITOR attributes. These situations are: • The LIST operand can be used if the caller has the group-SPECIAL or group-AUDITOR attribute in the current connect group or (if GRPLIST is active) in any group that the caller is connected to. • The REFRESH operand together with GENERIC can be used if the caller has the group-SPECIAL, AUDITOR, group-AUDITOR, OPERATIONS, group-OPERATIONS attribute, or CLAUTH authority for the classes specified. • The REFRESH operand together with GLOBAL can be used if the caller has the OPERATIONS attribute or CLAUTH authority for the classes specified. • The REFRESH operand together with RACLIST can be used if the caller has CLAUTH authority to the specified class. • The REFRESH operand together with WHEN(PROGRAM) can be used if the caller has CLAUTH authority for the program class or the OPERATIONS attribute.
SIGNOFF	{SM.3::SM.3-R12-RACFEAL5-27} None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).
STOP	{SM.3::SM.3-R12-RACFEAL5-28} None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).
TARGET	{SM.3::SM.3-R12-RACFEAL5-29} None (can be used an operator command only. RACF allows restricting the use of specific operator commands to defined users).

Table 31: Authorization required for RACF Commands and Command Parameters

8.4.5.5 Management of z/OS UNIX file systems

z/OS UNIX file systems considered in this evaluation include HFS, zFS, and NFS file systems. A file system is contained in an MVS data set and must be mounted before it can be used. Mounting of any of these file systems can occur via the auto-mount functions provided by z/OS UNIX System Services, or via an administrator-issued “mount” command.

- To mount a file system an administrator must have either UID(0) or at least READ authority to UNIXPRIV resource SUPERUSER.FILESYS.MOUNT {SM.3::SM.3-R13-UNIX-1}.

- {SM.3::SM.3-R13-UNIX-2} Additionally, a non-administrator may mount an HFS, zFS, or NFS file system if all of the following are true:
- The user has at least READ authority to UNIXPRIV resource SUPERUSER.FILESYS.USERMOUNT.
- The mount point directory must be empty.
- The user must have RWX permissions to the mount point, and if the mount point has the sticky bit set, the user must be the owner of the mount point.
- If the mount point directory resides in a remote file system (e.g., NFS), and if the mount point has the sticky bit set, then owner UID of the mount point directory must match the user's UID.
- The user must have RWX permissions to the root directory in the file system to be mounted, and if that root directory has the sticky bit set the user must be the owner of the root directory.
- The mount command must specify NOSETUID and must not specify NOSECURITY.
- For remote file systems (e.g., NFS), if the file system root has the sticky bit set then the owner UID of the file system root must match the user's UID.

Management of z/OS UNIX file system objects and IPC objects

Access permissions to z/OS UNIX file system objects and IPC objects are managed by functions in the z/OS UNIX System Services environment {SM.3::SM.3.35}. The standard functions to set or modify permission bits to file system objects and IPC objects also exist in the z/OS UNIX environment and allow users with the required permission to perform those actions {SM.3::SM.3.36}. In addition functions exist that allow the owner of a file system object to set or modify the access control list entries of this file system object {SM.3::SM.3.37}.

Note: The following three bullets only apply when an auto-mounter defines the file system and do not apply when an existing file system is otherwise mounted.

- In z/OS V2R1, the auto-mount capabilities for mounting a new zfs file system are being augmented to allow access permissions to be assigned. Prior when a newly created zfs file system was auto mounted, the root permissions were set to 750. In V2R1 through the allocation_spec field with the allocany and allocuser, the pathperm setting will allow the default 750 setting to be set to a different permission set {SM.3::V2R1-ZFS-1}.
- The auto-mounter initially allocates a file system (a one-time operation for a file system) when the profile for the managed directory specifies the "allocuser" or "allocany" keys. A new associated key is "pathperm" which specifies the root permissions for the newly created (zFS) file system. This field does not apply if the file system is already mounted. Likewise. for the "allocany" and "allocuser" keys, a new "EUID" key now is provided that indicated that the UID of the newly allocated zFS file system should be the EUID of the process rather than the RUID of the process (which is the default behavior) {SM.3::V2R1-ZFS-2}.

- In addition in V2R1, if the automount policy indicates “EUID” parm for the allocany/allocuser keys then the file system owner, when defined, is the EUID instead of the default RUID. {SM.3::V2R1-ZFS-3}.

When a non-privileged user unmounts a file system previously mounted using the USER MOUNT authorization, the user must have READ access to the SUPERUSER.FILESYS.USERMOUNT} profile and the file system must have been mounted by that user {SM.3::V2R1-ZFS-4}. In addition the user must still have access to the mount point and if the sticky bit is on, must still be the owner {SM.3::V2R1-ZFS-5}.

8.4.5.6 LDAP User and Group Management

LDAP LDBM Users

LDAP has the ability to authenticate to RACF through LDBM by supplying a RACF password/phrase on a simple bind to the LDBM back-end or by a digital certificates over TLS (LDAP SASL bind with EXTERNAL verification) mapped to a RACF USER ID. Authorization information is still gathered by the LDAP server back-end based on the DN that performed the bind operation. The LDAP administrator defines the authorized LDAP LDBM users by defining “subject distinguished names” DNs in the LDBM directory. Additionally, for the evaluated configuration, the administrator must define the DN as using what LDAP calls native authentication (i.e., RACF authentication) rather than LDAP authentication, and must provide the RACF user ID that represents this LDAP subject. During the bind operation, the client user will provide his/her subject DN and the RACF password/phrase for the RACF user ID that corresponds to that subject DN. The LDAP server will then use z/OS authentication functions to validate the specified password/phrase against the configured RACF user ID. (Note: Security claims appear earlier under Identification and Authentication functions.)

LDAP LDBM Groups

LDBM supports group definitions. These group definitions allow for a collection of names to be easily associated for access control checking. LDBM supports static (where the members are defined individually {SM.2::SM.2-R8-LDAP-1}), dynamic (where membership is determined using one or more LDAP search expressions {SM.2::SM.2-R8-LDAP-2}), and nested (a group that references other group entries that can be static, dynamic or nested groups {SM.2::SM.2-R8-LDAP-3}) group entries.

LDAP also supports, via the CDBM, an LDAP administrative group whose members have specific LDAP administrative capabilities based on the LDAP administrative roles assigned to them. When authenticating an LDAP user who is a member of the LDAP administrative group, LDAP does not make use of (gather) additional group information for that user {SM.2::SM.2-R13-LDAP-5}.

When configured to search SDBM for a user's groups, LDAP will convert those groups into SDBM DNs and add them to the LDBM user's set of groups, making them available for use in LDBM ACLs for authorization checking {SM.2::SM.2-R10-LDAP-4}.

LDAP Roles

The TOE supports the LDAP roles:

1. administrator {SM.1::SM.1-R8-LDAP-1},
2. member of the administrative group {SM.1::SM.1-R13-LDAP-13}

3. for basic replication: masterServer {SM.1::SM.1-R8-LDAP-2} (used as the master in LDAP replication processing), and peerServer {SM.1::SM.1-R8-LDAP-3} (used as a peer in LDAP replication processing),
4. for advanced replication: supplier (a server that sends changes to another (consumer) server, consumer (a server that receives changes via replication from another (supplier) server; {SM.1::SM.1-R12-LDAP-9}
5. and (by default) “end user”.{SM.1::SM.1-R12-LDAP-10}

Three non-default roles (administrator, and for basic replication, masterServer and peerServer) are defined within the LDAP configuration file.

End users have no pre-defined administrative rights, though under the control of access lists in the LDAP directory they may be allowed to create, or delete objects, or even manipulate the access lists for objects.

For advanced replication, a replication context is created as an entry in the directory with the auxiliary objectclass ibm-replicationContext that identifies the root of a replicated subtree. It can be added to any entry. The replication configuration information is maintained in a set of entries created below the base of a replication context. If there is more than one replication context present in the same subtree, the replication configuration information under the child replication context entry is used while the replication configuration under the parent entry is ignored. If the ibm-replicationContext auxiliary objectclass is added to a non-suffix level entry in the directory, explicit aclEntry and entryOwner attribute values are required. For advanced replication, supplier credentials are created in the directory associated with the replica subtree that identify the servers that are supplied (replicated to) by each server, and on a consumer server a credentials entry is required for each supplier and verified using a simple or SASL EXTERNAL bind. {SM.1::SM.1-R12-LDAP-11}

When configuring LDAP LDBM basic replication, replicas may be read-write, or read-only {SM.1::SM.1-R8-LDAP-5}. A peerServer can replicate its changes to other read-write replicas, and has the ability to update all data, bypassing all access list (ACL) controls {SM.1::SM.1-R8-LDAP-6}. A masterServer can replicate its changes to other read-write or read-only replicas {SM.1::SM.1-R8-LDAP-7}. A particular server may be both a peerServer to other read-write replicas, and a masterServer to read-only replicas {SM.1::SM.1-R8-LDAP-8}.

For advanced replication, a read-only replica is a consumer replica. If the replica can both read and write, the replica is both a consumer and supplier replica. {SM.1::SM.1-R12-LDAP-12}

The LDAP root Administrator has the ability to define LDAP groups to assist in the management of access rights and privileges {SM.1::SM.1-R13-LDAP-4}. Those administrator defined groups are not considered to be roles in the sense of the CC requirement FMT_SMR.1 but are just ways to manage access rights more easily.

The LDAP root Administrator also has complete access rights to all data in the LDAP LDBM and CDBM databases {SM.1::SM.1-R13-LDAP-14}.

{SM.1::SM.1-V2R1-LDAP-15} Additionally, the LDAP root Administrator can define members of the LDAP administrative group via entries in the CDBM database. The LDAP root Administrator may define those members (via DN or RACF group) within the directory subtree under cn=admingroup,cn=configuration or within the directory subtree under cn=safadmingroup,cn=configuration. With either method, the LDAP root Administrator may assign LDAP administrative roles to the administrative group members. The difference between the methods is that for members defined under cn=safadmingroup,cn=configuration the roles are derived by querying the user's access to RACF resources in the LDAP class, while for members defined under cn=admingroup,cn=configuration the user's administrative roles are specified directly within the CDBM entry for the user.

Briefly, The administrative roles supported in the z/OS LDAP server are:

- Directory data administrator – Intended to administer all DIT data (LDBM and cn=ibmpolicies in CDBM)
- No administrator – Intended to quickly revoke an administrative group member's administrative privileges.
- Operational administrator – Intended to request persistent search via control
- Password administrator – Intended to administer user passwords, for example Help Desk resetting a password of users in the LDBM backend.
- Replication administrator – Intended to administer advanced replication configurations.
- Root administrator – Super user (sum of all administrative roles, excluding NoAdmin)
- Schema administrator – Intended to administer the schema.
- Server configuration group member – Intended to administer cn=configuration.

In more detail, these are the characteristics of each of the LDAP administrative roles:

- {SM.1::SM.1-R13-LDAP-16} Directory Data Administrator – Users assigned this role have unrestricted access to the entries in the LDBM backend and the cn=ibmpolicies tree in the CDBM backend. aclEntry values are ignored if the bound user has this administrative role. Filtered ACLs will still be applied, but cannot augment any permissions explicitly denied by this role definition. The Directory Data Administrator has the following authority in the directory:
    - add, modify and delete access to Master Server DN and replication supplier entries of a consumer replica when also assigned the Schema Administrator and Server Configuration Group Member administrative roles. Note that any configurations in ds.conf are not included.
    - complete access to the cn=replication, cn=configuration entry in the CDBM backend
    - modify access to the ibm-slapdAdminPW attribute in their own configuration entry
    - complete access to entries in the LDBM backend. However, for setting the password of any entry, the Directory Data Administrator must abide by the normal password policy rules (if there are any). Users in this role cannot change the password of other administrative group members that exist in the directory.

  - read, search, and compare access to all entries in and under the cn=configuration suffix in the CDBM backend
    - ibm-slapdAdminDN, ibm-slapdAdminPW, and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs).
  - read access to the cn=schema entry
  - cannot add or delete entries under cn=replication, cn=configuration
- {SM.1::SM.1-R13-LDAP-17} No administrator – Users assigned this role have no administrative privileges. By defining this role the LDAP root administrator revokes all the administrative privileges of an administrative group member. The No Administrator has the following authority in the directory:
  - modify access to the ibm-slapdAdminPW attribute in their own configuration entry
  - read, search, and compare access to all entries in and under the cn=configuration suffix in the CDBM backend except for
    - ibm-slapdAdminDN, ibm-slapdAdminPW and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs).
  - read access to cn=schema entries
  - access to the LDBM backend through normal ACL evaluation. Filtered ACLs will still be applied, but cannot augment any permissions explicitly DENIED by this role definition. See <ref to ACL chapter> for more information.
- {SM.1::SM.1-R13-LDAP-18 Operational Administrator - Users assigned this role can perform persistent searches. aclEntry values are ignored when processing this role. ACL Filters do not apply. The operational administrator has the following authority in the directory:
  - can include the PersistentSearch control on searches.
  - has modify access to ibm-slapdAdminPW attribute in their own configuration entry
  - has read access to all entries in and under the cn=configuration suffix in the CDBM backend except for
    - ibm-slapdAdminDN, ibm-slapdAdminPW, and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry

    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs)
  - has read access to the cn=schema entry
  - has read access to the LDBM backend and cn=ibmpolicies in CDBM
- {SM.1::SM.1-R13-LDAP-19} Password administrator – Users assigned this role can unlock other user's accounts or change passwords of users in the LDBM backend. Password policy constraints set by the server do not apply to users assigned the password administrator role. This role has access to the LDBM backend through normal ACL evaluation. Filtered ACLs will still be applied, but cannot augment any permissions explicitly DENIED by this role definition. See <ref to ACL chapter> for more information. The password administrator has the following authority in the directory:
  - has modify access to the ibm-slapdAdminPW attribute in their own configuration entry
  - has read, search, and compare access to all the entries in and under cn=configuration except
    - ibm-slapdAdminDN, ibm-slapdAdminPW and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs)
  - has read, search, and compare access to the cn=schema entry
  - can unlock the user accounts or set the password without following normal password policy rules for all the users in backends except global administrative group members. This means users with the password administrator role:
    - have read, write and search access to the userpassword attribute and read, write, search and compare access to the pwdChangedTime attribute.
    - can add, delete, and modify the ibm-pwdAccountLocked attribute only if the new value of the attribute is false.
    - can delete pwdFailureTime, pwdAccountLockedTime, pwdExpirationWarned, and pwdGraceUseTime attribute values for all users.
- {SM.1::SM.1-R13-LDAP-20} Replication administrator – Users assigned the Replication Administrator role can update advanced replication topology objects. This role has access to the LDBM backend through normal ACL evaluation. Filtered ACLs

will still be applied, but cannot augment any permissions explicitly DENIED by this role definition. See <ref to ACL chapter> for more information. The Replication administrator has the following authority in the directory

- has no access to Master DN and replication supplier entries in a consumer replica
- has read, write, search, and compare access to cn=replication, cn=configuration entry in the CDBM backend
- has modify access to the ibm-slapdAdminPW attribute in their own configuration entry
- has read access to all the entries in and under cn=configuration   except
    - ibm-slapdAdminDN, ibm-slapdAdminPW and ibm-slapdAdminGroupEnabled attributes under the cn=configuration b entry
    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs).
- can issue the following extended operations: Cascading control replication, Control replication queue, Control replication, Control replication error log, Quiesce or Unquiesce replication context, and Replication topology .
- has read access to the cn=schema entry
- has read, write, search, compare, add and delete access to all the replication topology objects in the backends. The determination of replication objects is based on the presence of following objectclasses in the entry - ibm-replicationcontext, ibm-replicagroup, ibm-replicasubentry, ibm-replicationAgreement, ibm-replicationCredentials, ibm-replicationCredentialsSimple, ibm-replicationCredentialsExternal, ibm-replicationWeeklySchedule, ibm-replicationDailySchedule, and ibm-replicationfilter.

- {SM.1::SM.1-R13-LDAP-21} Root administrator – Users assigned this role has the same access to data in the LDAP server as the LDAP root administrator (adminDN in ds.conf). This role has permissions of all other roles except No administrator.
- {SM.1::SM.1-R13-LDAP-22} Schema administrator – Users assigned this role have unrestricted access to the schema back-end only. This role has access to the LDBM backend through normal ACL evaluation. Filtered ACLs will still be applied, but cannot augment any permissions explicitly DENIED by this role definition. See <ref to ACL chapter> for more information. The Schema administrator has the following authority in the directory:
    - modify access to the cn=schema entry
    - add, modify, and delete access to the Master Server DN and replication supplier entries (cn=configuration in the CDBM backend) of a consumer

replica when the user also has the directory data administrator and server configuration group member roles.

- modify access to the ibm-slapdAdminPW attribute in their own configuration entry
- read access to all the entries in and under cn=configuration except
    - ibm-slapdAdminDN, ibm-slapdAdminPW and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
    - ibm-slapdAdminPW attribute under other local administrative group member entries, and
    - passwords of advanced replication supplier credential entries (Master Server DNs).
- read, write, search and compare access to the cn=schema entry

- {SM.1::SM.1-R13-LDAP-23} Server configuration group member – Users assigned this role have restricted update access to the cn=configuration entries in the CDBM backend. This role has access to the LDBM backend through normal ACL evaluation. Filtered ACLs will still be applied, but cannot augment any permissions explicitly DENIED by this role definition. See <ref to ACL chapter> for more information. The Server configuration group member has following authority in the directory:
    - add, modify and delete access to all the entries in the CDBM cn=configuration backend except the following
        - ibm-slapdAdminDN, , and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
        - entries under the cn=AdminGroup, cn=configuration entry
    - add, modify, and delete access to the Master Server DN and replication supplier entries in the cn=configuration entry in the CDBM backend of a consumer replica when the user is also assigned directory data administrator and schema administrator roles.
    - modify access to the ibm-slapdAdminPW attribute in its own configuration entry
    - read access to all the entries in and under cn=configuration except
        - ibm-slapdAdminDN, ibm-slapdAdminPW and ibm-slapdAdminGroupEnabled attributes under the cn=configuration entry
        - ibm-slapdAdminPW attribute under other local administrative group member entries, and
        - passwords of advanced replication supplier credential entries (Master Server DNs).
    - read access to the cn=schema entry

{SM.1::SM.1-R13-LDAP-24} When the administrator assigns LDAP administrative roles via cn=safadmingroup, cn=configuration the DN specified must eventually lead to a RACF user

ID that has READ authorization to one or more of the RACF resources specified below. DNs satisfying that requirement might be for SDBM entries, a DNs from a TLS client certificate, a DN of an LDBM entry participating in native authentication, or a Kerberos mapped DN. The resource names used have the form domain-name.ADMINROLE.role where:

- domain-name is the value specified in the ibm-slapdSAFSecurityDomain attribute of the cn=configuration entry. The LDAP server uses the value in this attribute as the first part of the RACF defined administration role when it does SAF authorization checking on security related profiles for the administration roles.
- role is one of the following values:
    - CONFIG (for Server configuration group member)
    - DIRDATA (for Directory Data Administrator)
    - NOADMIN (for No Administrator)
    - OPER (for Operational Administrator)
    - PASSWD (for Password Administrator)
    - REPL (for Replication Administrator)
    - ROOT (for Root Administrator)
    - SCHEMA (for Schema Administrator)

LDAP Attributes

The ibm-nativeId LDAP attribute specifies the RACF user ID associated with an LDAP user authenticating to LDAP to access LDAP LDBM data.

Several attributes and object classes determine group membership for LDAP groups:

- For static groups in the accessGroup, groupOfNames, ibm-staticGroup object classes, the values of the member attribute determine group membership.
- For static groups in the groupOfUniquenames object class the values of the uniqueMember attribute determine group membership.
- For dynamic groups the scope and search filters contained in the values of the memberURL attribute determine group membership.
- For nested groups the values of the ibm-memberGroup attribute determine the groups that are members of the nested group.

8.4.5.7 RACF Certificate and Key Management

Digital Certificates, Key Rings, and Certificate Mappings in RACF and PKCS#11 Cryptographic Tokens

RACF provides the RACDCERT command which can be used to

- create certificate requests to send to a Certifying Authority {SM.1::SM.1-R8-RACF-RACDCERT-1}

- generate public/private key pairs and certificates (DIGTCERT class) {SM.1::SM.1-R8-RACF-RACDCERT-2}
- export a certificate or certificate packages to a data set, optionally with the private key {SM.1::SM.1-R8-RACF-RACDCERT-3}
- install certificates into the RACF database and register them as belonging to a user or to a certifying authority {SM.1::SM.1-R8-RACF-RACDCERT-4}. The __certificate() and InitACEE() services can also register/deregister certificates {SM.1::SM.1-R8-RACF-RACDCERT-5}, and administrators an allow users to register their own certificates by granting them READ access to FACILITY resource IRR.DIGTCERT.ADD {SM.1::SM.1-R8-RACF-RACDCERT-6}.
- delete or list certificates in the RACF database {SM.1::SM.1-R8-RACF-RACDCERT-7}
- maintain (create, list, delete) key rings containing certificates (DIGTRING class) {SM.1::SM.1-R8-RACF-RACDCERT-8}
- add certificates to or delete them from key rings {SM.1::SM.1-R8-RACF-RACDCERT-9}
- create mapping rules (certificate name filters) that can map client certificates that are not installed/registered in the database to specified user IDs based on subject or issuer information (DIGTNMAP class) {SM.1::SM.1-R8-RACF-RACDCERT-10}. This can allow a many-to-one mapping for applications that do not need to have each user run under his own ID. In this case, accountability can be maintained for auditing purposes by having the application provide the subject's distinguished name via the X500Name parameter when creating the security environment (ACEE) for the user {SM.1::SM.1-R8-RACF-RACDCERT-11}. The mapping process can also make use of mapping criteria specified by the DIGTCRIT class when it is necessary to map a client certificate into different IDs depending on characteristics of the user's session (such as the application name, or system name where the application is running) {SM.1::SM.1-R8-RACF-RACDCERT-12}.
- create and manage the contents of PKCS#11 cryptographic tokens contained in the ICSF TKDS {SM.1::SM-1.R9-RACF-RACDCERT-13}

{SM.1::SM-1.R12-RACF-RACDCERT-14} RACDCERT supports installing or generating certificates that have the following key characteristics, subject to US export regulations and the available cryptographic hardware present on the system:

- RSA keys up to 4096 bits
- DSA keys up to 1024 bits;
- NIST ECC keys up to 521 bits;
- Brainpool ECC keys up to 512 bits (non-twisted curves only).

{SM.1::SM-1.V2R1-RACF-RACDCERT-15} RSA keys can be generated through

- RACF software and stored in the RACF DB (clear key)
- Crypto Express3 or Crypto Express4S coprocessor cards (z114, z196 or EC12 processor) and stored in the ICSF PKDS or TKDS (secure key)

{SM1.::SM-1.V2R1-RACF-RACDCERT-16} NIST/Brainpool keys can be generated through

- ICSF software and stored in the RACF database (clear key)
- Crypto Express3 or Crypto Express4S coprocessor cards (z114, z196 or EC12 processor) and stored in the ICSF PKDS or TKDS (secure key)

z/OS also provides the PKI Services component which provides a full-function Certificate Authority and certificate life-cycle management process. Certificates that PKI Services issues are not (by default) placed in the RACF database, but may be put there manually by users or administrators. See PKI Services for additional details.

The rest of this section describes processing in RACF.

Profiles in the DIGTCERT class contain information about digital certificates contained in the RACF database, as well as the certificate itself and optionally the certificate's private key. Additionally, the user's USER profile will have information about a certificate associated with the user.

Profiles in the DIGTRING class contain information about key rings and the certificates contained in a key ring. Each key ring is a named collection of the personal, site, and CA certificates associated with a user. When the user represents a server, the key ring has the allowable CA certificates that must be used to sign certificates presented by clients of the server during TLS handshaking.

Profiles in the DIGTNMAP and DIGTCRIT classes contain profiles used during certificate name filtering, a process during client authentication that can derive a user ID to use for the session from a certificate that is not specifically registered in the RACF database.

Note that only the RACDCERT command may be used to administer profiles in the DIGTCERT, DIGTRING, and DIGTNMAP classes.

Management for RACF Digital Certificates, Key Rings, Certificate Mappings, and Criteria

Administrators can use the RACDCERT command to generate or delete digital certificates, generate certificate requests, maintain key rings, and maintain certificate mappings. RACF maintains certificates in the DIGTCERT class, key rings in the DIGTRING class, and certificate mappings in the DIGTNMAP class.

Additionally RACF provides programming interfaces to allow applications to maintain RACF key rings.

Management for RACF digital certificates, key rings, certificate mappings, and certificate mapping criteria occurs during processing of the Authority checking for RACDCERT Processing or the use of the associated programming interfaces as described above. It also occurs during TLS processing, Communications Server Network Security Server processing, or other processing using the R_datalib programming interfaces to read or update RACF key ring information.

The authority to perform the individual management operations is determined by checking the user's access to specific RACF profiles. This access check processing generally follows the normal MVS DAC algorithm for general resources described above in the section on discretionary access control, using specific resource names in the FACILITY class that depend on the function requested. It also allows users with SPECIAL to perform certain of the functions, as explained below.

Authority checking for RACDCERT Processing

Note: Since the check for sufficient authority to perform one of the management functions of RACDCERT is performed by checking the user's authority to specific profiles using the standard RACF access check algorithm, the claims in this section start with "AC" instead of "SM".

In general to use RACDCERT users need either the SPECIAL attribute (AC.4-R9-RACF-1) or

- READ access to FACILITY resource IRR.DIGTCERT.function to issue RACDCERT commands for themselves {SM.7::AC.4-R9-RACF-2};
- UPDATE access to FACILITY resource IRR.DIGTCERT.function to issue RACDCERT commands for other users {SM.7::AC.4-R9-RACF-3};
- CONTROL access to FACILITY resource IRR.DIGTCERT.function to issue RACDCERT commands for SITE and CERTAUTH certificates {SM.7::AC.4-R9-RACF-4}.

Note that access to the LISTCHAIN function is controlled by the IRR.DIGTCERT.LIST profile {SM.7::AC.4-V2R1-RACF-30}.

Note that if the CSFSERV class is active, a number of RACDCERT functions require additional authorizations to resources in this class, caused by the ICSF functions called as part of the RACDCERT processing. For details of the access rights and resources required see the description of the RACDCERT subcommands in [RACF.CMD].

The following tables describe the basic functions and the authorities used for each RACDCERT function in more detail {SM.7::AC.4-V2R1-RACF-29}:

FUNCTION	**READ**	**UPDATE**	**CONTROL**
ADD	Add a certificate to one's own ID	Add a certificate to another user’s ID	Add a site or certificate authority certificate
ADDRING	Create a key ring for one’s own ID	Create a key ring for another user’s ID	n/a
ADDTOKEN (controlled only via CRYPTOZ class)	n/a	n/a	n/a
ALTER	Change the trust status or label of one’s own certificate	Change the trust status or label of another user’s certificate	Change the trust status or label of a site or certificate authority certificate

FUNCTION	READ	UPDATE	CONTROL
ALTMAP	Alter a mapping associated with one's own ID	Alter a mapping associated with another user's ID or with MULTIID	n/a
BIND (Also see CRYPTOZ class)	See BIND table	See BIND table	See BIND table
CHECKCERT (Note: uses LIST as the function in the DAC check)	Check one's own certificate	Check another user's certificate	Check a site or certificate authority certificate
CONNECT	See Connect tables	See Connect tables	See Connect tables
DELETE	Delete one's own certificate	Delete another user's certificate	Delete a site or certificate authority certificate
DELMAP	Delete a mapping associated with one's own ID	Delete a mapping associated with another user's ID or with MULTIID	n/a
DELRING	Delete one's own key ring	Delete another user's key ring	n/a
DELTOKEN (controlled only via CRYPTOZ class)	n/a	n/a	n/a
EXPORT	See Export table	See Export table	See Export table
GENCERT	See Gencert table	See Gencert table	See Gencert table
GENREQ	Generate a request based on one's own certificate	Generate a request based on another user's certificate	Generate a request based on a site or certificate authority certificate
IMPORT (also see CRYPTOZ class)	See ADD above.	See ADD above.	See ADD above.

FUNCTION	READ	UPDATE	CONTROL
LIST	List one's own certificate	List another user's certificate	List a site or certificate authority certificate
LISTCHAIN (Note: use of this subcommand is controlled by the IRR.DIGTCERT.LIST profile in the FACILITY class)	No access, command will terminate with an error	No access, command will terminate with an error	Display information about a digital certificate and its issuer chain of certificates in the RACF database.
LISTMAP	List mapping information associated with one's own ID	List mapping information associated with another user's ID or MULTIID	n/a
LISTTOKEN (also see CRYPTOZ class)	See LIST above	See LIST above	See LIST above
MAP	Create a mapping associated with one's own ID	Create a mapping associated with another user's ID or MULTIID	n/a
REMOVE	Remove a certificate from one's own key ring	Remove a site or certificate authority certificate from one's own key ring	Remove a certificate from another user's key ring
REKEY	Rekey one's own certificate	Rekey another user's certificate	Rekey a site or certificate authority certificate
ROLLOVER	Rollover one's own certificate	Rollover another user's certificate	Rollover a site or certificate authority certificate
UNBIND (controlled only via CRYPTOZ class)	n/a	n/a	n/a

Table 32: Authorizations required for RACDCERT functions

This table describes the authorities needed to perform the BIND function to bind a certificate to a PKCS#11 token:

USAGE	One's own certificate	Another user's certificate	A site or certificate authority certificate
PERSONAL	READ authority to IRR.DIGTCERT.BIND	UPDATE authority to IRR.DIGTCERT.BIND	CONTROL authority to IRR.DIGTCERT.BIND
SITE CERTAUTH	CONTROL authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.BIND	CONTROL authority to IRR.DIGTCERT.ADD and UPDATE authority to IRR.DIGTCERT.BIND	UPDATE authority to IRR.DIGTCERT.BIND

Table 33: Authorizations required for RACDCER BIND

This table describes the authorities needed to perform the CONNECT function to connect a certificate to one's own key ring:

USAGE	One's own certificate	Another user's certificate	A site or certificate authority certificate
PERSONAL	READ authority to IRR.DIGTCERT.CONNECT	UPDATE authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.CONNECT
SITE CERTAUTH	CONTROL authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.ADD and UPDATE authority to IRR.DIGTCERT.CONNECT	UPDATE authority to IRR.DIGTCERT.CONNECT

Table 34: Authorizations required for RACDCERT CONNECT (own key ring)

This table describes the authorities needed to perform the CONNECT function to connect a certificate to another user's key ring:

USAGE	One’s own certificate	Another user’s certificate	A site or certificate authority certificate
PERSONAL	CONTROL authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.CONNECT
SITE CERTAUTH	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.CONNECT	CONTROL authority to IRR.DIGTCERT.CONNECT

Table 35: Authorizations required for RACDCERT CONNECT (not own key ring)

This table describes the authorities needed to perform the EXPORT function:

Function	READ	UPDATE	CONTROL
EXPORT (in CERT format)	Export one’s own certificate	Export another user’s certificate	Export a site or certificate authority certificate
EXPORT (in PKCS#7 format)	Export one’s own certificate but not the parent CA chain	Export another user’s certificate but not the parent CA chain	Export site or certificate authority certificates or the entire parent CA chain for oneself or another user.
EXPORT (in PKCS#12 format. Note: uses EXPORTKEY as the function in the DAC check)	Export one’s own certificate and the private key	Export another user’s certificate and the private key	Export a site or certificate authority certificate and the private key

Table 36: Authorizations required for RACDCERT EXPORT

This table describes the authorities needed to perform the GENCERT function:

SIGNWITH option chosen	To generate one's own certificate	To generate another user's certificate	To generate a site or certificate authority certificate
SIGNWITH one's own certificate	READ authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT
SIGNWITH a SITE or CERTAUTH certificate	READ authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT
SIGNWITH not specified	READ authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.GENCERT	UPDATE authority to IRR.DIGTCERT.ADD and UPDATE authority to IRR.DIGTCERT.GENCERT	CONTROL authority to IRR.DIGTCERT.ADD and CONTROL authority to IRR.DIGTCERT.GENCERT

Table 37: Authorizations required for RACDCERT GENCERT

Authority Checking for R_datalib Processing
The R_datalib callable services provides access to some fields of certificates and key rings, including when appropriate the private keys when stored in RACF. R_datalib allows reading, creation, or modification of key rings As with RACDCERT functions, the SPECIAL attribute authorizes some functions. In addition, profiles in the RDATALIB class or in the FACILITY class can authorize various R_datalib functions.
When using the FACILITY class, RACF will use resource names of the form IRR.DIGTCERT.function to authorize the processing, where the descriptions below will describe the applicable function values.
When using the RDATALIB class, RACF will use resource names of the form <ringOwner>.<ringName>.function, where the descriptions below will the describe the applicable function values.
The ringOwner must be in upper case. The ringName will be folded into upper cases during profile checking. Rings differ only in case will be using the same profile {SM.7::AC.4-R9-RACF-26}.
In the case the owner ID and the ring name are of their maximum limits, and you want to create a discrete profile, it can be done by truncating the ring name from the end so that the whole profile name length is 246 characters {SM.7::AC.4-R9-RACF-27}.

If the input Ring_name is of the virtual keyring form, a single ‘*’, the ring name part in the resource will be IRR_VIRTUAL_KEYRING so that different profiles can be set up to control access on real and virtual keyrings {SM.7::AC.4-R9-RACF-28}.
If the caller of R_datalib provides an owner ID of *TOKEN*, then the request specifies use of a PKCS#11 cryptographic token in the ICSF TKDS, and all security checking occurs in ICSF using the CRYPTOZ class. R_datalib does not do any checking in the FACILITY or RDATALIB classes for these cases {SM.7::AC.4-R9-RACF-30}. For more information on this case see Authority Checking for PKCS11 Cryptographic Tokens in the ICSF TKDS.
For the DatagetFirst, DataGetNext, and GetUpdateCode functions:
Using RDATALIB Checking for a Real Keyring {SM.7::AC.4-R9-RACF-5}:

Access to <ringOwner>.<ringName>.LST in the RDATALIB class, e.g. SERVER1.FTPRING1.LST	**Action able to perform**
READ	DataGetFirst, DataGetNext: list Server1’s ring named FTPring1, and returns one’s own private key if the usage is PERSONAL GetUpdateCode: return the sequence number of Server1’s ring named FTPring1
UPDATE	DataGetFirst, DataGetNext: list Server1’s ring named FTPring1, and returns another's private key if the usage is PERSONAL
CONTROL (or caller is RACF SPECIAL)	DataGetFirst, DataGetNext: list Server1’s ring named FTPring1, and returns SITE/CA’s private key if the usage is PERSONAL

Table 38: Authorizations for RDATALIB - real key ring

Using RDATALIB Checking for a Virtual Keyring {SM.7::AC.4-R9-RACF-6}:

Virtual keyring owner	**Resource Name**	**Access**	**Action able to perform**
Ordinary ID, e.g. USER1	USER1.IRR_VIRTUAL_KEYRING.LST	READ	DataGetFirst, DataGetNext: list USER1’s virtual keyring, and returns the private keys if the caller is USER1, i.e. the

Virtual keyring owner	Resource Name	Access	Action able to perform
			owner of the virtual keyring GetUpdateCode: return the sequence number
		UPDATE	DataGetFirst, DataGetNext: list USER1’s virtual keyring, and returns the private key GetUpdateCode: return the sequence number
CERTAUTH	CERTIFAUTH.IRR_VIRTUAL_KEYRING.LST	READ	DataGetFirst, DataGetNext: list CERTAUTH’s virtual keyring GetUpdateCode: return the sequence number
SITE	SITECERTIF.IRR_VIRTUAL_KEYRING.LST	READ	DataGetFirst, DataGetNext: list SITE’s virtual keyring GetUpdateCode: return the sequence number

Table 39: Authorizations for RDATALIB - virtual key ring

Using FACILITY Checking {SM.7::AC.4-R9-RACF-7}:

Access to IRR.DIGTCERT. LISTRING in the FACILITY class	Action able to perform
READ	DataGetFirst, DataGetNext: list one’s own real or virtual ring, and returns one’s own private key if the usage is PERSONAL list one’s own real or virtual ring, and returns SITE/CA’s private key if the usage is PERSONAL, if caller is SPECIAL or has CONTROL

Access to IRR.DIGTCERT. LISTRING in the FACILITY class	**Action able to perform**
	to IRR.DIGTCERT.GENCERT in the FACILITY class GetUpdateCode: return the sequence number of one’s own real or virtual ring
UPDATE	DataGetFirst, DataGetNext: list another's real or virtual ring, and returns SITE/CA’s private key if the usage is PERSONAL if caller is SPECIAL or has CONTROL to IRR.DIGTCERT.GENCERT in the FACILITY class GetUpdateCode: return the sequence number of another's real or virtual ring

Table 40: Authorizations for RDATALIB - FACILITY checking - LISTRING

For the CheckStatus function:

- The call requires READ authority to resource IRR.DIGTCERT.LIST in the FACILITY class {SM.7::AC.4-R9-RACF-8}.

For the IncSerialNum function:

- The call requires either the SPECIAL attribute {SM.7::AC.4-R9-RACF-9} or
  - READ authority to resource IRR.DIGTCERT.GENCERT in the FACILITY class if the caller owns the certificate {SM.7::AC.4-R9-RACF-10};
  - CONTROL authority to resource IRR.DIGTCERT.GENCERT in the FACILITY class for a site or certificate authority certificate {SM.7::AC.4-R9-RACF-11}.

For the NewRing function:

- No checking will be performed if the caller has the RACF SPECIAL attribute {SM.7::AC.4-R9-RACF-12}.

Using RDATALIB Profile Checking: {SM.7::AC.4-R9-RACF-13}:

Access to <ringOwner>.<ringName>. UPD in the RDATALIB class, e.g. SERVER1.FTPRING1.UPD	**Action able to perform**
READ	• add a new ring for Server1 named FTPring1 • remove all certificates from the the existing ring named FTPring1 owned by Server1

Table 41: Authorizations for RDATALIB - Profile checking - NewRing

Using FACILITY Profile Checking: {SM.7::AC.4-R9-RACF-14}:

Access to IRR.DIGTCERT. ADDRING in the FACILITY class	**Access to IRR.DIGTCERT.REMOVE in the FACILITY class**	**Action able to perform**
READ	n/a	create one’s own new ring
UPDATE	n/a	create another's new ring
n/a	READ	remove certificates from one’s ring
n/a	UPDATE	remove certificates from another's ring

Table 42: Authorizations for RDATALIB - FACILITY checking - NewRing

For the Del Ring Function:

- No checking will be performed if the caller has the RACF SPECIAL attribute {SM.7::AC.4-R9-RACF-15}.

Using RDATALIB Profile Checking {SM.7::AC.4-R9-RACF-16}:

Access to <ringOwner>.<ringName>.UPD in the RDATALIB class, e.g. SERVER1.FTPRING1.UPD	**Action able to perform**
READ	delete a ring owned by Server1 named FTPring1

Table 43: Authorizations for RDATALIB - Profile Checking - DelRing

Using FACILITY Profile Checking {SM.7::AC.4-R9-RACF-17}:

Access to IRR.DIGTCERT.DELRING in the FACILITY class	**Action able to perform**
READ	delete one’s own ring
UPDATE	delete another's ring

Table 44: Authorizations for RDATALIB - FACILITY checking - DelRing

For the DataRemove Function:

- No checking will be performed if the caller has the RACF SPECIAL attribute {SM.7::AC.4-R9-RACF-18}.

Using RDATALIB Profile Checking {SM.7::AC.4-R9-RACF-19}:

Access to <ringOwner>.<ringName>. UPD in the RDATALIB class, E.g. SERVER1.FTPRING1.UPD	**Action able to perform**
READ	remove one’s own cert from Server1’s ring named FTPring1
UPDATE	remove one’s own or another's cert from Server1’s ring named FTPring1
CONTROL	remove any type cert from Server1’s ring named FTPring1

Table 45: Authorizations for RDATALIB - Profile Checking - DataRemove

Using FACILITY Profile Checking {SM.7::AC.4-R9-RACF-20}:

Access to IRR.DIGTCERT.REMOVE in the FACILITY class	**Action able to perform**
READ	remove one’s own cert from one’s ring
UPDATE	remove any type cert from one’s ring
CONTROL	remove any type cert from another's ring

Table 46: Authorizations for RDATALIB - FACILITY checking - DataRemove

In addition, if the DataRemove operation specifies CDDL_ATT_DEL_CERT_TOO, then RACF will also check, IRR.DIGTCERT.DELETE whether using RDATALIB or FACILITY profiles {SM.7::AC.4-R9-RACF-21}:

Access to IRR.DIGTCERT.DELETE in the FACILITY class	**Action able to perform**
READ	delete one’s own cert from RACF if it is not

Access to IRR.DIGTCERT.DELETE in the FACILITY class	Action able to perform
	connected to other rings
UPDATE	delete one’s or another's cert from RACF if it is not connected to other rings
CONTROL	delete any type cert from RACF if it is not connected to other rings

Table 47: Authorizations for RDATALIB - FACILITY checking - DataRemove - specific parameters

For the DataPut Function:

- No checking will be performed if the caller has the RACF SPECIAL attribute {SM.7::AC.4-R9-RACF-22}.

Note: In the following tables,

- Any usage = PERSONAL, CERTAUTH or SITE
- Any type cert = certificate is owned by any regular ID, or by the site or a certificate authority.

Using RDATALIB Profile Checking {SM.7::AC.4-R9-RACF-23}:

- With READ Access to <ringOwner>.<ringName>.UPD, e.g. SERVER1.FTPRING1.UPD

	Input cert is not in RACF	Input cert is already in RACF		Input cert is in RACF and already connected to the ring	
		with no private key	with private key	with no private key	with private key
Input cert only Input cert and private key	• add one’s own cert • connect to Server1’s ring named FTPring1 one’s own cert with usage PERSONAL only	if cert owned by caller • connect to Server1’s ring named FTPring1 with usage PERSONAL only, other usages cause error • change the NOTRUST status to TRUST if trust flag turns on if cert is not owned by caller, error		if cert owned by caller • re-connect to Server1’s ring named FTPring1 with usage PERSONAL only, other usages cause error, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	

	Input cert is not in RACF	**Input cert is already in RACF**		**Input cert is in RACF and already connected to the ring**	
		with no private key	**with private key**	**with no private key**	**with private key**
				if cert is not owned by caller, error	
		if cert owned by caller • re-add cert with private key • connect to Server1's ring named FTPring1 with usage PERSONAL only, other usages cause error • change the NOTRUST status to TRUST if trust flag turns on if cert is not owned by caller, error	if cert owned by caller • connect to Server1's ring named FTPring1 with usage PERSONAL only, other usages cause error • change the NOTRUST status to TRUST if trust flag turns on if cert is not owned by caller, error	if cert owned by caller • re-add cert with private key • re-connect to Server1's ring named FTPring1 with usage PERSONAL only, other usages cause error, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	if cert owned by caller • re-connect to Server1's ring named FTPring1 with usage PERSONAL only, other usages cause error, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on if cert is not owned by caller, error

	Input cert is not in RACF	**Input cert is already in RACF**		**Input cert is in RACF and already connected to the ring**	
		with no private key	**with private key**	**with no private key**	**with private key**
				if cert is not owned by caller, error	

Table 48: Authorizations for RDATALIB - Profile checking - DataPut - Part 1

With UPDATE Access to <ringOwner>.<ringName>.UPD, e.g. SERVER1.FTPRING1.UPD

	Input cert is not in RACF	**Input cert is already in RACF**		**Input cert is in RACF and already connected to the ring**	
		with no private key	**with private key**	**with no private key**	**with private key**
Input cert only	• add any type cert • connect to Server1's ring named FTPring1 one's own cert with any usage or • connect another's or SITE/CA's cert with usage SITE or CERTAUTH only, PERSONAL usage causes error	• connect to Server1's ring named FTPring1 one's own cert with any usage or • connect to Server1's ring named FTPring1 another's or SITE/CA's cert with usage SITE or CERTAUTH only, PERSONAL usage causes error • change the NOTRUST status to TRUST if trust flag turns on		• re-connect to Server1's ring named FTPring1 one's own cert with any usage or • re-connect to Server1's ring named FTPring1 another's or SITE/CA's cert with usage SITE or CERTAUTH only, PERSONAL usage causes error, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	

	Input cert is not in RACF	**Input cert is already in RACF**		**Input cert is in RACF and already connected to the ring**	
		with no private key	**with private key**	**with no private key**	**with private key**
Input cert and private key	• add any type cert • connect to Server1's ring named FTPring1 any type cert with any usage	• re-add any type cert with private key under original ID • connect to Server1's ring named FTPring1 any type cert with any usage • change the NOTRUST status to TRUST if trust flag turns on	• connect to Server1's ring named FTPring1 any type cert with any usage • change the NOTRUST status to TRUST if trust flag turns on	• re-add any type cert with private key under original ID • re-connect to Server1's ring named FTPring1 any type cert with any usage, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	• re-connect to Server1's ring named FTPring1 any type cert with any usage, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on

Table 49: Authorizations for RDATALIB - Profile checking - DataPut - Part 2

With CONTROL Access to <ringOwner>.<ringName>.UPD, e.g. SERVER1.FTPRING1.UPD

	Input cert is not in RACF	**Input cert is already in RACF**		**Input cert is in RACF and already connected to the ring**	
		with no private key	**with private key**	**with no private key**	**with private key**
Input cert only	• add any type cert • connect to Server1's ring named FTPring1 any type cert with any usage	• connect to Server1's ring named FTPring1 any type cert with any usage • change the NOTRUST status to TRUST if trust flag turns on		• re-connect to Server1's ring named FTPring1 any type cert with any usage, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	
Input cert and private key		• re-add cert with private key under original ID • connect to Server1's ring named FTPring1 any type cert with any usage • change the NOTRUST status to TRUST if trust flag turns on	• connect to Server1's ring named FTPring1 any type cert with any usage • change the NOTRUST status to TRUST if trust flag turns on	• re-add cert with private key under original ID • re-connect to Server1's ring named FTPring1 any type cert with any usage, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on	• re-connect to Server1's ring named FTPring1 any type cert with any usage, with new specified default value • change the NOTRUST status to TRUST if trust flag turns on

Table 50: Authorizations for RDATALIB - Profile checking - DataPut - Part 3

Using FACILITY Profile Checking {SM.7::AC.4-R9-RACF-24}:

Certificate does not exist in RACF Database

Access to IRR.DIGTCERT.ADD in the FACILITY class	**Access to IRR.DIGTCERT.CONNECT in the FACILITY class**	**Action able to perform**
READ	READ	• add one’s own cert • connect one’s own cert with usage PERSONAL to one’s own ring
CONTROL	READ	• add one’s own cert • connect one’s own cert with any usage to one’s own ring
UPDATE	UPDATE	• add one’s own or another's cert • connect one’s own or another's cert with usage PERSONAL to one’s ring or • connect SITE/CA’s cert with SITE/CERTAUTH usage to one’s own ring
CONTROL	UPDATE	• add any type cert • connect one’s own or another's cert with usage PERSONAL to one’s ring or • connect any type cert with usage SITE/CERTAUTH to one’s ring
UPDATE	CONTROL	• add one’s own or another's cert • connect any type cert with usage PERSONAL to any ring or • connect SITE/CA’s cert with any usage to any ring
CONTROL	CONTROL	• add any type cert • connect any type cert with any usage to any ring

Table 51: Authorizations for RDATALIB - FACILITY checking - DataPut - Part 1

Certificate exists in RACF Database with no private key but private key is specified

Access to IRR.DIGTCERT.ADD in the FACILITY class	Access to IRR.DIGTCERT.CONNECT in the FACILITY class	Action able to perform
READ	READ	• re-add one's own cert with private key • change the NOTRUST status of the connected cert to TRUST if trust flag turns on • connect one's own cert with usage PERSONAL to one's own ring
CONTROL	READ	• re-add one's own cert with private key • change the NOTRUST status of the connected cert to TRUST if trust flag turns on • connect one's own cert with any usage to one's own ring
UPDATE	UPDATE	• re-add one's own or another's cert with private key • change the NOTRUST status of the connected cert to TRUST if trust flag turns on • connect one's own or other's cert with usage PERSONAL to one's ring or • connect SITE/CA's cert with SITE/CERTAUTH usage to one's own ring
CONTROL	UPDATE	• re-add any type cert with private key • change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on • connect one's own or other's

Access to IRR.DIGTCERT.ADD in the FACILITY class	Access to IRR.DIGTCERT.CONNECT in the FACILITY class	Action able to perform
		cert with usage PERSONAL to one's ring or • connect any type cert with usage SITE/CERTAUTH to one's ring
UPDATE	CONTROL	• re-add one's own or other's cert with private key • change the NOTRUST status of the connected cert to TRUST if trust flag turns on • connect any type cert with usage PERSONAL to any ring or • connect SITE/CA's cert with any usage to any ring
CONTROL	CONTROL	• re-add any type cert with private key • change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on • connect any type cert with any usage to any ring

Table 52: Authorizations for RDATALIB - FACILITY checking - DataPut - Part 2

Certificate already exists in RACF Database and no private key is input

Access to IRR.DIGTCERT.ADD in the FACILITY class	Access to IRR.DIGTCERT.CONNECT in the FACILITY class	Access to IRR.DIGTCERT.ALTER in the FACILITY class (will be checked if changing status from NOTRUST to TRUST/HIGHTRUST is requested)	Action able to perform
n/a	READ	READ	• connect one's own cert with usage PERSONAL to one's own ring

Access to IRR.DIGTCERT.ADD in the FACILITY class	**Access to IRR.DIGTCERT.CONNECT in the FACILITY class**	**Access to IRR.DIGTCERT.ALTER in the FACILITY class (will be checked if changing status from NOTRUST to TRUST/HIGHTRUST is requested)**	**Action able to perform**
			• change the NOTRUST status of the connected cert to TRUST if trust flag turns on
CONTROL	READ	READ	• connect one's own cert with any usage to one's own ring • change the NOTRUST status of the connected cert to TRUST if trust flag turns on
n/a	UPATE	READ – one's own cert UPDATE – other's cert CONTROL – SITE/CA's cert	• connect one's own or other's cert with usage PERSONAL to one's ring or • connect SITE/CA's cert with SITE/CERTAUTH usage to one's own ring • change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on
CONTROL	UPDATE	READ – one's own cert UPDATE – other's cert CONTROL – SITE/CA's cert	• connect one's own or other's cert with usage PERSONAL to one's own ring or • connect SITE/CA's cert with SITE/CERTAUTH usage to one's own ring

Access to IRR.DIGTCERT.ADD in the FACILITY class	**Access to IRR.DIGTCERT.CONNECT in the FACILITY class**	**Access to IRR.DIGTCERT.ALTER in the FACILITY class (will be checked if changing status from NOTRUST to TRUST/HIGHTRUST is requested)**	**Action able to perform**
			• change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on
n/a	CONTROL	READ – one's own cert UPDATE – other's cert CONTROL – SITE/CA's cert	• connect any type cert with usage PERSONAL to any ring or • connect SITE/CA's cert with any usage to any ring • change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on
CONTROL	CONTROL	READ – one's own cert UPDATE – other's cert CONTROL – SITE/CA's cert	• connect any type cert with any usage to any ring • change the NOTRUST status of the connected cert to TRUST/HIGHTRUST if trust flag turns on

Table 53: Authorizations for RDATALIB - FACILITY checking - DataPut - Part 3

For the DataRefresh Function:

No checking will be performed if the caller has the RACF SPECIAL attribute, otherwise if the DIGTCERT class is SETR RACLISTed then the caller needs class authority (CLAUTH) to the DIGTCERT class {SM.7::AC.4-R9-RACF-25}.

8.4.6

8.4.6.1 Network configuration and management

z/OS provides some basic configuration data sets for TCP/IP and TCP/IP based protocols. Those configuration data sets that are also related to security are:

- PROFILE.TCPIP

  Provides TCP/IP initialization parameters and specifications for network interfaces and routing.
- TCPIP.DATA

  Provides parameters for TCP/IP based client and server programs.
- Additional Communications Server configuration information (e.g., IPSec and AT-TLS) exists in policy files accessed via the Communications Server Policy Agent.
- The IKE daemon, NSS server, Defense Manager daemon, FTP server, TN3270 server, and Policy Agent (as well as other servers) also have their own configuration files.
- The HTTP server configuration file (default: httpd.conf)

Configuration statements in those data sets define the properties (including security properties) of the TCP/IP protocol itself as well as the main protocol server.

Communications Server ipsec Command Interface

The Communications Server provides a command named ipsec that allows authorized users to query information about IP filters, defensive filters and IPSec security associations. It also allows authorized users to activate or deactivate IPSec functions, affect which IP filters are loaded, and create, update and delete defensive filters. The administrator can control access to this command by granting READ access to the following resources in the SERVAUTH class:

- EZB.IPSECCMD.sysname.tcpname.DISPLAY allows clients to display information about IP filters, per-stack defensive filters and IPSec security associations {SM.4::SM-R10-CS-IPSECCMD-1}.
- EZB.IPSECCMD.sysname.tcpname.CONTROL allows clients to reload or refresh IP filters, create, update or delete per-stack defensive filters and activate or deactivate IPSec security associations {SM.4::SM-R10-CS-IPSECCMD-3}.
- EZB.IPSECCMD.sysname.DMD_GLOBAL.DISPLAY allows clients to display information about global defensive filters {SM.4::SM-R10-CS-IPSECCMD-4}.
- EZB.IPSECCMD.sysname.DMD_GLOBAL.CONTROL allows clients to create, update or delete global defensive filters {SM.4::SM-R10-CS-IPSECCMD-6}.

Communications Server Network Management Interface

The Communications Server provides, via the IKE daemon, a network management interface (NMI) that allows local applications to query information about IP filters and IPSec security associations. It also allows applications to activate or deactivate IPSec functions. The IKE daemon provides this information via a UNIX (not TCP/IP) socket. The administrator can control access to this interface by granting READ access to the following resources in the SERVAUTH class:

- EZB.NETMGMT.sysname.tcpname.IPSEC.DISPLAY allows clients to display information about IPSec filtering and security associations. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface {SM.4::SM-R10-CS-SECMON-1}.
- EZB.NETMGMT.sysname.tcpname.IPSEC.CONTROL allows clients to issue management requests to activate, deactivate, or modify IPSec security associations. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface. {SM.4::SM-R10-CS-SECMON-3}.
- EZB.NETMGMT.sysname.sysname.IKED.DISPLAY allows clients to display information about IKE daemon usage of the Network Security Services (NSS) client functions via the NMI or the ipsec command with the -w option. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface. {SM.4::SM-R10-CS-SECMON-4}.

Additionally, the Network Security Services (NSS) server provides a network management interface that allows a central administrator to monitor and control NSS and IPSec information in a manner similar to that provided by the IKE daemon. For these network management requests, the administrator can use the following SERVAUTH resources to provide protection:

- EZB.NSS.sysname.clientname.IPSEC.NETMGMT allows clients to register with the NSS server for IPSec network management services {SM.4::SM-R9-CS-NSS-1}.
- EZB.NETMGMT.sysname.clientname.IPSEC.DISPLAY allows clients to display IPSec-related information via the NSS NMI or the ipsec command with the –z option {SM.4::SM-R9-CS-NSS-2}.
- EZB.NETMGMT.sysname.clientname.IPSEC.CONTROL allows clients to issue management requests to activate, deactivate, or modify IPSec security associations via the NSS NMI or the ipsec command with the –z option {SM.4::SM-R9-CS-NSS-3}.
- EZB.NETMGMT.sysname.sysname.NSS.DISPLAY allows clients to display information about current NSS client connections to the NSS server via the NSS NMI or the ipsec command with the –x option {SM.4::SM-R9-CS-NSS-4}.

Communications Server is providing security controls over what type of trace information can be provided through their network management interfaces (NMI) for ipsec and AT-TLS data. The administrator can use the following SERVAUTH resources to control the generation of and access to trace information by applications intended to perform network management functions:

- EZB.TRCCTL.sysname.tcpname.OPEN allows applications to invoke the NMI to open a trace {SM.4::SM-V2R1-CS-TRS-1};
- EZB.TRCCTL.sysname.tcpname.PKTTRACE allows an application to invoke the NMI to set filters for packet trace {SM.4::SM-V2R1-CS-TRS-2};
- EZB.TRCCTL.sysname.tcpname.DATTRACE allows an application to invoke the NMI to set filters for data trace {SM.4::SM-V2R1-CS-TRS-3};

- EZB.TRCSEC.sysname.tcpname.IPSEC allows an application to request IPSec cleartext data on a packet trace filter {SM.4::SM-V2R1-CS-TRS-4};
- EZB.TRCSEC.sysname.tcpname.ATTLS allows an application to request AT-TLS cleartext data on a data trace filter {SM.4::SM-V2R1-CS-TRS-5}.

Communications Server Policy Agent

The Communications Server provides a Policy Agent that can act in any of several roles, depending on configuration options:

- The Policy Agent may act as the Policy Definition Point (PDP) on a single system, installing policies in one or more z/OS Communications Server stacks {SM.4::SM-R9-CS-POLCEN-1}.
- The Policy Agent may act as a centralized policy server, providing PDP services for one or more remote policy clients {SM.4::SM-R9-CS-POLCEN-2}.
- The Policy Agent may act as a policy client, retrieving remote policies from the policy server. Each stack in a Common INET (CINET) environment acts as a separate policy client {SM.4::SM-R9-CS-POLCEN-3}. Communications between the policy client and the policy server may optionally be secured by AT-TLS {SM.4::SM-R10-CS-POLCEN-11}.

A single Policy Agent may act as a policy client or a policy server, but not both {SM.4::SM-R9-CS-POLCEN-11}.

Policies may be defined in several different ways. When acting as the PDP for a single system, Policy Agent can read policy definitions from local configuration files, a central repository that uses the Lightweight Directory Access Protocol (LDAP), or both {SM.4::SM-R9-CS-POLCEN-4}.

The Policy Agent also installs policies in one or more z/OS Communications Server stacks. It can be used to replace existing policies or update them as necessary {SM.4::SM-R9-CS-POLCEN-5}.

When acting as a policy server, Policy Agent also acts as a PDP for the local system, and so can read policies from local configuration files or an LDAP server, and install them in local stacks {SM.4::SM-R9-CS-POLCEN-6}. But it also reads policies from local configuration files on behalf of policy clients. These policies are retrieved by policy clients, but are not installed in the local stacks on the policy server {SM.4::SM-R9-CS-POLCEN-7}.

When acting as a policy client, Policy Agent retrieves remote policies from the policy server, and can also use local policies from configuration files or an LDAP server {SM.4::SM-R9-CS-POLCEN-8}.

The choice of local or remote policies can be made separately for each type of supported policy: Quality of Service (QoS), Intrusion Detection (IDS), Policy-Based Routing (PBR), IPSec, or AT-TLS {SM.4::SM-R9-CS-POLCEN-9}. For a given policy type, all policies are obtained either locally or remotely {SM.4::SM-R9-CS-POLCEN-10}.

When acting as a policy server, the policy agent will:

- First, authenticate its clients using a RACF user ID and password or PassTicket {SM.4::IA-R9-CS-POLCEN-1}.

- Then, authorize retrieval of policy data, requiring READ access to policy agent resources in the SERVAUTH class. These resources must be protected or retrieval will fail {SM.4::AC-R9-CS-POLCEN-1}. They have the form EZB.PAGENT.sysname.image.ptype {SM.4::AC-R9-CS-POLCEN-2} where
    - Sysname is the system name defined in the sysplex
    - Image is the TCP name or policy client name
    - Ptype is either QOS, IDS, IPSEC, or (for AT-TLS) TTLS.

8.4.6.2 PKI Services Management

Administrator Functions

The administrator can use the administration web pages to perform the following tasks:

- Process a certificate request
    - Approve a request without making changes {SM.5::SM-R8-PKI-28}
    - Approve a request with changes {SM.5::SM-R8-PKI-29}
    - Reject a request {SM.5::SM-R8-PKI-30}
    - Delete a request {SM.5::SM-R8-PKI-31}
- Process a certificate
    - Revoke a certificate {SM.5::SM-R8-PKI-32}
    - Suspend a certificate {SM.5::SM-R8-PKI-33}
    - Resume a certificate {SM.5::SM-R8-PKI-34}
    - Delete a certificate {SM.5::SM-R8-PKI-35}
- Perform searches for certificate requests and certificates {SM.5::SM-R8-PKI-36}

Security Administration for PKI Services

PKI Services security administration comprises the following tasks:

- Authorizing users for the PKI Services administration group (connecting and deleting members)
- Authorizing users for inquiry access
- granular administrator authorization controls may be configured to scope the capabilities of the PKI Services Administrators as defined below.

These granular controls are enabled via a new keyword value pair in the PKI Services configuration file; **AdminGranularControl={T | F}** {SM.5::SM-V2R1-PKI-37}. When enabled, each administrative action is authority checked against the appropriate resource name in the **PKISERV** class from with in the PKI Services address space. These authority checks are in addition to the **IRR.RPKISERV.PKIADMIN*[.ca-domain]*** resource of the **FACILITY** class made with in the R_PKIServ callable service {SM.5::SM-V2R1-PKI-38}. Each administrative function has a unique corresponding resource name in the **PKISERV** class.

For most administrative actions, the resource name is based on the CA domain, the action to be performed, and the certificate template name under which the certificate was requested. The **PKISERV** class resource names are in the following form:

<CA-domain-name>.<administrative-action>.<certificate-template-name>

There are two special administrative actions that are associated with the PKI Services utilities **createcrls** and **postcerts**. These utilities are not directly related to a certificate template, therefore the **PKISERV** class resource names are in the following form:

<CA-domain-name>.<administrative-action>

The CA-domain-name is the 1 to eight character name that represents the specific instance of PKI Services. When a CA domain name is not in use by the PKI Services instance (an un-named domain), the CA-domain-name used in the resource name is: **NOCADOMAIN**

The certificate-template-name is the 1 to eight character name that represents the specific certificate template nickname defined for use by the PKI Services Web Application. When a certificate or certificate request is created without a certificate template, the certificate-template-name used in the resource name is: **NONICKNAME**

The following are the list of administrative-action names and their corresponding R_PKIServ function and action codes:

QUERYREQS	R_PKIServ QUERYREQS function
QUERYREQDETAILS	R_PKIServ REQDETAILS function
QUERYCERTS	R_PKIServ QUERYCERTS function
QUERYCERTDETAILS	R_PKIServ CERTDETAILS function
PREREGISTER	R_PKIServ PREREGISTER function
APPROVE	R_PKIServ MODIFYREQS function with action code “approve”
APPROVEWITHMODS	R_PKIServ MODIFYREQS function with action code “approve with modifications”
REJECT	R_PKIServ MODIFYREQS function with action code “reject”
DELETEREQS	R_PKIServ MODIFYREQS function with action code “delete”
REVOKE	R_PKIServ MODIFYCERTS function with action code “revoke”
DELETECERTS	R_PKIServ MODIFYCERTS function with action code “delete certificate from issued certificate

	list”
RESUME	R_PKIServ MODIFYCERTS function with action code “resume suspended certificate”
AUTORENEWENABLE	R_PKIServ MODIFYCERTS function with action code “Enable automatic certificate renewal”
AUTORENEWDISABLE	R_PKIServ MODIFYCERTS function with action code “Disable automatic certificate renewal”
CHANGEMAIL	R_PKIServ MODIFYCERTS function with action code “Change requestor email”
CREATECRL	R_PKIServ MODIFYCERTS function with action code “Create CRLs”
POSTCERT	R_PKIServ MODIFYCERTS function with action code “Post Certificates”

Resource name examples:

NOCADOMAIN.QUERYREQS.1YBSSL - Covers query for certificate requests that used the “1-Year Browser SSL certificate” template with nickname 1YBSSL on an un-named CA domain

MASTERCA.RESUME.NONICKNAME - Covers the MODIFYCERTS resume administrative action for a certificate that was created without a certificate template name on the MASTERCA CA domain.

SUBCA1.CREATECRL - Covers the MODIFYCERTS create CRLs administrative action on the SUBCA1 CA domain.

When the granular administrative controls are enabled, the PKI Services daemon will perform a third party authorization check using the userid of the administrator for each administrative action against the appropriate resource name {SM.5::SM-V2R1-PKI-39}. The following describes the results for administrative actions for the different access levels:

No protection profile	Treated the same as an access of **NONE**
NONE	For administrative actions QUERYREQS and QUERYCERTS, no results or authorization failures are returned. For all other administrative actions, an authorization failure is returned {SM.5::SM-V2R1-PKI-40}
READ	For administrative actions QUERYREQDETAILS and QUERYCERTDETAILS, all result information is returned with exception of the passphrase (if present). If a passphrase is present in the request or certificate record, no authorization

	failure is returned {SM.5::SM-V2R1-PKI-41}. For all other administrative actions, the administrator has full authorization to perform the action and receives all available results {SM.5::SM-V2R1-PKI-42}.
UPDATE and higher	The administrator has full authorization to perform the action and receives all available results {SM.5::SM-V2R1-PKI-43}.

8.4.6.3 Security Management for System Logger Log Streams

Applications can read and write to defined log streams as explained in the DAC section of this document. However, before they can do this an administrator or an application must define the log stream and the policies that apply to it.

The system policy for log streams exists in an MVS data set known as the “LOGR couple data set”. Administrators who need to define or view the logger policy information use the IXCMIAPU utility program to do so. They require:

READ authority to the MVSADMIN.LOGR resource in the FACILITY class in order to generate reports about the logger policy {SM.6::SM-R9-LOGGER-1}.

Additionally, logger administrators who need to define, in the CFRM policy, coupling facility structures that will be utilized by log streams will also need UPDATE authority to the MVSADMIN.XCF.CFRM resource in the FACILITY class {SM.6::SM-R9-LOGGER-3}.

Additionally, logger administrators who need to define, delete, or modify the definitions of log streams will need:

- ALTER authority to resource log_stream_name in class LOGSTRM to define, delete, or update the stream {SM.6::SM-R9-LOGGER-4}
- ALTER authority to resource MVSADMIN.LOGR in class FACILITY to define or delete a coupling facility structure for use by a log stream {SM.6::SM-R9-LOGGER-6}.
- UPDATE authority to resource IXLSTR.structure_name in class FACILITY to associate the named coupling facility structure with a log stream {SM.6::SM-R9-LOGGER-7}.
- **UPDATE authority to resource IXGZAWARE_CLIENT in class FACILITY when specifying ZAI(YES) to indicate the log stream is to be treated as a z/OS IBM zAware log stream client {SM.6::SM-V2R1-LOGGER-12}.**

Applications wishing to administer log streams using the programming interfaces will need:

ALTER authority to resource log_stream_name in class LOGSTRM to define, update the definition of, or delete a log stream {SM.6::SM-R9-LOGGER-8}.

Additionally, UPDATE to resource name IXLSTR.structure_name in class FACILITY to define a log stream that uses a coupling facility structure {SM.6::SM-R9-LOGGER-9}

Additionally, when defining a log stream modeled upon the definition of another log stream, UPDATE access to resource IXLSTR.model_structure_name in class FACILITY (when the model stream has a structure) {SM.6::SM-R9-LOGGER-10}.

ALTER to resource MVSADMIN.LOGR in class FACILITY if they wish to use logger interfaces to define coupling facility structures {SM.6::SM-R9-LOGGER-11}

8.4.6.4 Audit configuration and management

Within the system configuration it needs to be decided, which SMF records shall be generated by z/OS. Three record types (type 80, 81, and 83) are dedicated to RACF and are the most important ones for security. Which events are actually recorded with those records can be configured by a user with the AUDITOR attribute in his RACF user profile {AU.3::AU.3.1}. In addition record type 30 is generated for a number of security related events.

Because a set of mandatory events is always audited, not all audit records (such as unauthorized attempts to access the system or changes to the status of the RACF database) can be configured.

In addition, resource profiles can define which events related to this resource are audited {AU.3::AU.3.2}. The owner of a resource profile as well as a user in the AUDITOR role are able to change the entries related to auditing within the resource profile {AU.3::AU.3.3}.

The system can be configured to send certain audit messages to the security console to immediately alert operators of detected policy violations {AU.3::AU.3.4}.

8.4.7 Object Re-Use (FDP_RIP)

z/OS provides explicit object reuse functionality for the following objects, and z/OS ensures that these objects are prepared for reuse before they are allocated to another subject:

- Memory objects are filled with zeros before they are allocated for the first time to a subject {OR.1::OR.1.1}.
- z/OS data sets are erased when the data is released when the erase-on-scratch option is active {OR.1::OR.1.2}.
- z/OS system log streams that reside in z/OS data sets are cleared by the system logger before it writes any data into them. Similarly, for z/OS log stream data residing in a coupling facility the system logger clears the structure data in the coupling facility before writing any data into the structure {OR.1::OR.1-R9-LOGGER-1}.
- z/OS tape volumes are erased when they are returned to the scratch pool by appropriately configuring the SECCLS parmlib option for the parmlib member EDGRMMxx {OR.1::OR.1-R8-RMM-1} or under control of the appropriate data set profile's ERASE option when TAPEAUTHDSN=YES is specified in SYS1.PARMLIB(DEVSUPxx) {OR.1::OR.1-R8-RMM-2}.
- z/OS UNIX file system objects and z/OS UNIX IPC objects are cleared before they are made accessible to a new subject {OR.1::OR.1.3-R13}.
- LDAP LDBM objects are not specifically cleared when they are deleted, but LDAP does ensure that any data returned from an object is not residual data from some previous object that may have occupied the same physical space in the LDBM database {OR.1::OR.1-R8-LDAP-1}.

8.4.8 Self Protection and Time Management

8.4.8.1 Event Notifications generated by RACF

RACF can send an ENF type 62 signal to listeners when a SETROPTS RACLIST command affects in-storage profiles used for authorization checking. RACF sends a signal when a SETROPTS RACLIST, SETROPTS NORACLIST, or SETROPTS RACLIST REFRESH command is issued for a class, activating, deactivating, or updating the profiles. Signals are sent for a class in the static class descriptor table if SIGNAL=YES was specified on the ICHERCDE macro that defined the class. Signals are sent for a class in the dynamic class descriptor table if SIGNAL(YES) was specified on the CDTINFO keyword of the RDEFINE or RALTER command that defined the class.{SM.3::SM.3-R12-RACFEAL5-33}

These signals enable an application or resource manager that uses those classes of resources to react appropriately after an administrator implements a change that may affect subsequent security decisions by the application or resource manager.

8.4.8.2 Time Management

An operator with appropriate authorization can use the SET command to set the time (using the CLOCK parameter) and the date (using the DATE parameter). Time within a Parallel Sysplex can be synchronized using a Sysplex Timer (when supported by the hardware) or the Server Time Protocol.

8.4.8.3 Session Locking

The hardware available for the TOE (see section on hardware configuration) does not provide means for direct connections of synchronous terminals any more. Since the requirements stated in the SFRs FTA_SSL.1 and FTA_SSL.2 are targeted towards such direct connections, they are irrelevant to this TOE and therefore trivially met.

8.4.9 Communication Security

8.4.9.1 Communications Server

z/OS provides basic networking functions with the Communications Server component. This subsystem provides support for network communication using the IBM SNA protocols as well as the TCP/IP protocol suite. APIs for both protocol stacks are provided. For IP, both IPv4 and IPv6 are supported.

The Communications Server uses RACF to protect access of users to the following resources:

- the TCP/IP stack in general {CS.1::CS.1.1}
- TCP and UDP ports {CS.1::CS.1.2}
- IP addresses {CS.1::CS.1.3}
- Centralized policy information for QoS (Qualities of Service), PBR (Policy-Based Routing), IPSec, IDS (Intrusion Detection Services), and AT-TLS policy {CS.1::CS.1-R9-CS-POLCEN-1}.

- Network management information related to IP Filters and IPSec security associations {CS.1::CS.1-R9-CS-SECMON-1}
- Network Security Services, which provides centralized services for clients allowing:
  - IKE daemons to perform RSA and ECDSA signature generation and verification as well as RSA and ECDSA certificate management/validation functions (ECDSA is only supported with IKEv2) through the System SSL gsk_validate_certificate_mode API{CS.1::CS.1-R12-CS-NSS-5} and through additional certificate revocation checking. {CS.1::CS.1-R13-CS-NSS-6}
  - XMLAppliance clients to perform remote RACF authentication and access control functions {CS.1::CS.1-R10-CS-NSS-1}, certificate management functions {CS.1::CS.1-R11-CS-NSS-2}, private key retrieval {CS.1::CS.1-R11-CS-NSS-3}, and RSA signature creation and RSA decryption {CS.1::CS.1-R11-CS-NSS-4}.

z/OS provides the following security functions as part of the Communications Server:

- Access Control for the IP stack and access control to ports and port ranges:

  The IP stack as well as TCP/UDP ports and port ranges can be protected with RACF. Users can be granted or denied access to the IP stack in general as well as to individual ports and port ranges. See TCP/IP connections for the associated security claims.
- z/OS Communications Server provides packet filtering functionality that can control information flow into or out of the system based on security characteristics of the packets or of the network interface they use.
- IPSec security associations:

  The Communications Server can be configured to establish IPSec security associations at the IP layer. All packets transmitted between security association endpoints will be authenticated, encrypted, or both using the configured algorithms. The Communications Server provides support for IPSec-protected communication in accordance with RFCs 4301 through 4305, 4308, and 4835 {CS.1::CS.1-R12-IPSec-1} and IKEv2 in accordance with RFCs 5996, 4307 through 4308, 4718, 4753, 4754, 4809, 4868, 4869 and 4945 {CS.1::CS.1-R13-IPsec-6}. It also provides the IKE application that negotiates IPSec security association parameters with communication peers {CS.1::CS.1-R8-IPSec-2}. IPSec is configured through the PROFILE.TCPIP configuration and the Policy Agent (see section Network configuration and management).

  IPSec, when authenticating using certificates, will obtain the subject alternate extension present in the client's certificate and compare it contents to the identity defined in the IKE security policy {CS.1::CS.1-R12-IPSec-7}.

  In the evaluated configuration the following encryption algorithms are supported:
  - AES-CBC-128, AES-CBC-256 (both specified by RFC 3602), AES-GCM-128 as specified in RFC 4106, AES-GCM-256 as specified in RFC 4106 for ESP encryption {CS.1::CS.1-V2R1-IPSec-8};

  - HMAC-SHA1-96, AES-XCBC-MAC-96 for ESP authentication and authentication header protection CS.1::CS.1-V2R1-IPSec-9};
  - IKEv1 as defined in RFCs 2407, 2408, 2409, RFC 4109, and no other RFCs for hash functions, IKEv2 as defined in RFCs 5996, 4307 and no other RFCs for hash functions, for key negotiation and SA establishment CS.1::CS.1-V2R1-IPSec-10};
  - DH Groups 14 (2048-bit MODP), and 24 (2048-bit MODP with 256-bit POS), 19 (256-bit Random ECP), 20 (384-bit Random ECP), DH Group 21 (521-bit), DH Group 5 (1536-bit) (not usable when configured for FIPS mode) for use in IKE key establishment; (Note: also DH Groups 1 and 2 are supported, but not in FIPS mode) CS.1::CS.1-V2R1-IPSec-11};
  - ECDSA and RSA algorithm for Peer Authentication {CS.1::CS.1-V2R1-IPSec-12}.
- Note: When hardware crypto has been activated, the cryptographic operations performed by IPSec {CS.1::CS.1-R8-IPSec-3} and System SSL {CS.1::CS.1-R8-SSL-1} will make use of the hardware crypto when appropriate, either through ICSF or the CPACF processor instructions. In the absence of hardware crypto support, IPSec {CS.1::CS.1-R9-IPSec-4} and System SSL {CS.1::CS.1-R9-SSL-2} will use software algorithms for cryptographic operations, although in the case of AES (CBC, GCM, and GMAC) encryption and SHA-2 digests IPSec will still make use of ICSF {CS.1::CS.1-R13-IPSec-5}. For symmetric (TDES, AES) and hashing (SHA-1, SHA-2) functions, ICSF will invoke the CPACF if it supports the function or will provide the functions via software within ICSF {CS.1::CS.1-R13-ICSF-1}.
- A Network Security Services (NSS) server that can be used by:
  - IKE daemons to perform RSA signature generation and verification and certificate management/validation from a centralized location, minimizing the number of systems on which digital certificates for the IKE daemons must be installed
  - Network management applications to monitor and manage IPSec on NSS client nodes (see the Network Management section)
  - XMLAppliance applications to remotely perform RACF user authentication and access control calls for RACF resources that the application specifies, as well as certificate management operations, retrieval of private keys from RACF certificates, and RSA signature generation and RSA decryption using ICSF-protected keys.

  The Network Security Server will authenticate its clients using the RACF user ID and password or PassTicket that they provide and will ensure that the connection is protected by AT-TLS {IA.1::IA-R9-CS-NSS-1}.

  For the IKE certificate-based processing, the Network Security Server will authorize use of its services via resources in the SERVAUTH class:

  - EZB.NSS.sysname.clientname.IPSEC.CERT to control whether the client can request certificate services {AC.4::AC-R9-CS-NSS-1}

    - EZB.NSSCERT.sysname.mappedlabelname.CERTAUTH to control whether the client can access a CERTAUTH certificate on the NSS server’s key ring {AC.4::AC-R9-CS-NSS-2}.
    - EZB.NSSCERT.sysname.mappedlabelname.HOST to control whether the client can access a personal or SITE certificate on the NSS server’s key ring {AC.4::AC-R9-CS-NSS-3}.

  For the XMLAppliance certificate processing, the Network Security Server will authorize use of certificates and private keys via resources in the SERVAUTH class:

    - EZB.NSSCERT.sysname.mappedlabelname.CERTAUTH or EZB.NSSCERT.sysname.mappedlabelname.HOST to control whether the client can access a certificate on the NSS server's keyring.{AC.4::AC-R11-CS-NSS-9}
    - EZB.NSSCERT.sysname.mappedlabelname.PRIVKEY to control whether the client can access a private key, either through direct retrieval or for usage in RSA signature generation or RSA decryption.{AC.4::AC-R11-CS-NSS-10}

  The Network Security Server will authorize use of the network management service via the EZB.NSS.sysname.clientname.IPSEC.NETMGMT resource in the SERVAUTH class. NSS IPSec clients that a connect with a user ID permitted to this resource are allowed to utilize the IPSec monitoring and management service provided by the NSS server {AC.4::AC-R12-CS-NSS-4}.

  The Network Security Server will authorize the use of XMLAppliance services as follows:

    - RACF user authentication and access control calls require READ access to the EZB.NSS.sysname.clientname.XMLAPPLIANCE.SAFACCESS resource in the SERVAUTH class. {AC.4::AC-R10-CS-NSS-5}.
    - Certificate management calls require READ access to the EZB.NSS.sysname.clientname.XMLAPPLIANCE.CERT resource in the SERVAUTH class {AC.4::AC-R11-CS-NSS-6}
    - Private key retrieval calls require READ access to the EZB.NSS.sysname.clientname.XMLAPPLIANCE.PRIVKEY resource in the SERVAUTH class {AC.4::AC-R11-CS-NSS-7}
    - RSA signature generation and RSA decryption calls from XMLAppliance clients require READ access to the EZB.NSS.sysname.clientname.XMLAPPLIANCE.PRIVKEY resource in the SERVAUTH class {AC.4::AC-R11-CS-NSS-8}

- TLS layer to set up a trusted channel to another trusted IT product, in a way transparent to the application (called Application Transparent TLS, or AT-TLS). The selectable algorithms can be limited by configuring a subset of allowable algorithms at the server. The TLS protocol can be used to set up a trusted channel to another system through a potentially insecure network. TLS protects the data against disclosure and attacks related to integrity like undetectable modifications or replay. Servers can support encryption using TDES with 168-bit key length or AES with either 128- or 256-bit key length. Application Transparent Transport Layer Security (AT-TLS) supports the use of all cipher suites supported by System SSL {CS.1::CS.1.4-R13}. The TN3270 and FTP protocols are enabled to use AT-TLS and can be tunneled through TLS to establish a trusted channel to another trusted IT product that also implements this protocol {CS.1::CS.1.5}. Applications that AT-TLS has been configured to support, can be tunneled through TLS to establish a trusted channel to another trusted IT product that also implements this protocol {CS.1::CS.1-V1R7.1}.

AT-TLS is configured through the PROFILE.TCPIP configuration file and the Policy Agent. This configuration may also specify a list of LDAP servers for certificate revocation information (see Section Network configuration and management).

- An rpcbind application with the following characteristics:
    - Control over which users can register or deregister application port information, which prevents unauthorized users from directing application RPC requests to the wrong TCP/IP port. To implement this security control, administrators define a RACF SERVAUTH profile to protect EZB.RPCBIND.<system-name>.<rpc-bind-name>.REGISTRY and give appropriate users READ access. This control protects the registerrpc(), svc_register(), pmap_set(), and pmap_unset() services {CS.1::CS.1-R10-CS-1}. In a multilevel secure environment no application can register or deregister with rpcbind unless this profile exists and grants access {CS.1::CS.1-R10-CS-2}.
- Packet filtering functionality that can control information flow into or out of the system based on security characteristics of the packets or of the network interface they use, as follows:
    - {CS.1::CS.1-R12-PF-1} Filter rules can apply to a packet based on information within the packet or information external to the packet.
        - Internal information: source address, destination address, protocol, source port, or destination port, ICMP type and code, OSPF type, and mobility header type.
        - External information: the direction of packet flow routing attribute, security class (determined by the network interface used by the packet).
    - {CS.1::CS.1-R12-PF-2} When a filter rule applies to a packet, it can discard the packet silently, discard the packet with ICMP notification to the sender, permit the packet flow, or permit the packet flow and enforce IPSec processing.
    - {CS.1::CS.1-R11-PF-3} A z/OS TCP/IP stack configured for IP security implements a default “deny” policy in the absence of any configured filter rules.

In addition, the Communications Server provides the following application protocols that include user authentication using RACF:

- FTP (user authentication is optional) {CS.1::CS.1.6}
- telnet {CS.1::CS.1.7}

- rlogin, rsh, and rexec {CS.1::CS.1.8}
- TN3270 {CS.1::CS.1.9}
- Network Security Services Server {CS.1::CS.1-R10-NSS-6}
- Policy Agent Server {CS.1::CS.1-R10-CS-POLCEN-12}
- Load Balancing Advisor {CS.1::CS.1-R10-CS-LBA-3}

z/OS also provides an HTTP server that uses RACF for authentication, (though the administrator can also configure anonymous access if necessary) {CS.1::CS.1.V1R7.2}

Access control to resources used within a FTP, HTTP, or telnet session is also performed using RACF {CS.1::CS.1.10}.

Import of certificates and key pairs used for authentication and key exchange for the TLS and IPSec protocols is restricted to authorized administrators {CS.1::CS.1.11}.

The FTP and TN3270 Server applications can use AT-TLS services to provide end-to-end data channels that are authenticated and encrypted {CS.1::CS.1-R8-CS-1}. AT-TLS (Application Transparent Transport Layer Security) uses System SSL services to provide end-to-end data channels that are authenticated and encrypted for most TCP applications.

8.4.9.2 System SSL

z/OS provides TLS functions via the System SSL component for applications wishing to use TLS directly (without taking advantage of the AT-TLS functions of the Communications Server). The selectable algorithms can be limited by configuring a subset of allowable algorithms at the server {CS.2::CS.1-R8-SSL-2}. The TLS protocol can be used to set up a trusted channel to another system through a potentially insecure network. TLS protects the data against disclosure and attacks related to integrity like undetectable modifications or replay. Servers can support encryption using TDES with 168-bit key length {CS.2::CS.1-R8-SSL-3}, AES with either 128- or 256-bit key length {CS.2::CS.1-R8-SSL-4} utilizing session keys generated with information provided through an RSA key exchange method. These same encryption types can utilize session keys generated with information provided through an EC Diffie-Hellman key exchange {CS.2::CS.1-R13-SSL-6}.

The following cipher suites are supported:

- TLS_RSA_WITH_AES_128_CBC_SHA {CS.2::CS.1-V2R1-SSL-7}
- TLS_RSA_WITH_AES_256_CBC_SHA {CS.2::CS.1-V2R1-SSL-8}
- TLS_RSA_WITH_AES_128_CBC_SHA256 {CS.2::CS.1-V2R1-SSL-9}
- TLS_RSA_WITH_AES_256_CBC_SHA256 {CS.2::CS.1-V2R1-SSL-10}
- TLS_DH_DSS_WITH_AES_128_CBC_SHA {CS.2::CS.1-V2R1-SSL-11}
- TLS_DH_RSA_WITH_AES_128_CBC_SHA {CS.2::CS.1-V2R1-SSL-12}
- TLS_DHE_DSS_WITH_AES_128_CBC_SHA {CS.2::CS.1-V2R1-SSL-13}
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA {CS.2::CS.1-V2R1-SSL-14}
- TLS_DH_DSS_WITH_AES_256_CBC_SHA {CS.2::CS.1-V2R1-SSL-15}
- TLS_DH_RSA_WITH_AES_256_CBC_SHA {CS.2::CS.1-V2R1-SSL-16}
- TLS_DHE_DSS_WITH_AES_256_CBC_SHA {CS.2::CS.1-V2R1-SSL-17}

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA {CS.2::CS.1-V2R1-SSL-18}
- TLS_DH_DSS_WITH_AES_128_CBC_SHA256 {CS.2::CS.1-V2R1-SSL-19}
- TLS_DH_RSA_WITH_AES_128_CBC_SHA256 {CS.2::CS.1-V2R1-SSL-20}
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

8.4.9.3 Network Authentication Service

The z/OS Network Authentication Service provides communication security via the Kerberos and GSS-API protocols, which use one of the supported encryption protocols (TDES, AES-128, AES-256) to encrypt application messages when requested by applications that support Kerberos and GSS-API functions {CS.3::CS.1-R13-KERB-1}.

The z/OS Network Authentication Service will utilize ICSF to drive encryption requests to CPACF for DES, TDES, AES-128, and AES-256).{CS.3::CS.1-R13-KERB-2}

If requested by the applications using the appropriate AP_REQ extension the z/OS Network Authentication Service will allow the applications to negotiate and agree upon encryption types not understood by the KDC {CS.3::CS.1-R13-KERB-3}.

8.4.9.4 NFS Client and Server

The z/OS NFS client and server support the use of Kerberos (via the Network Authentication Service) to provide integrity and confidentiality for authentication credentials and data as they flow over the network. NFS server configuration parameters allow the administrator to configure use of Kerberos for network traffic and the z/OS NFS client and server support the following Kerberos V5 security mechanisms {CS.4::CS.1-R10-NFS-1}:

- krb5, which provides Kerberos V5 based integrity on the RPC credentials (but not data) using the DES_MAC_MD5 integrity algorithm and uses the RPCSEC_GSS service of rpc_gss_svc_none.

- krb5i, which provides Kerberos V5 based integrity on both the RPC credentials and data using the DES_MAC_MD5 integrity algorithm and uses the RPCSEC_GSS service of rpc_gss_svc_integrity.

- krb5p, which provides Kerberos V5 based integrity and privacy on both the RPC credentials and data using the DES_MAC_MD5 algorithm for integrity and 56 bit DES for privacy. It uses the RPCSEC_GSS service of rpc_gss_svc_privacy.

When acquiring Kerberos tickets the z/OS NFS client supports the following encryption options {CS.4::CS-1-R10-NFS-5}:

- ENCTYPE_DES_CBC_MD5

8.4.9.5 IBM Ported Tools for z/OS (OpenSSH)

Additionally, the IBM Ported Tools for z/OS provide OpenSSH functionality, with an sshd daemon that supports the SSHv2 protocol {CS.5::CS.1-R8-SSH-1} and these commands to allow remote users to perform work on the z/OS system:

- ssh, to establish a UNIX shell environment {CS.5::CS.1-R8-SSH-2}
- scp to perform remote file copying operations {CS.5::CS.1-R8-SSH-3}
- sftp to perform file transfer operations (similar to ftp) {CS.5::CS.1-R8-SSH-4}
- ssh-keygen to generate the host key files and the RSA or DSA key pairs {CS.5::CS.1-R8-SSH-7}

The SSH protocol can be used to set up a trusted channel to another system through a potentially insecure network. SSH protects the data against disclosure and attacks related to integrity like undetectable modifications or replay. SSH supports encryption using TDES with 168-bit key length {CS.5::CS.1-R8-SSH-5} and AES with 128-, 192-, or 256-bit key length {CS.5::CS.1-R8-SSH-6}, {CS.5::CS.1-R9-OpenSSH-1}.

8.4.9.6 IPSec

Communications Server supports IPSec, and Internet Key Exchange (IKE). IP security for z/OS Communications Server supports two versions of the IKE protocol: IKEv1 and IKEv2.

These features are implemented in the IP layer on a per packet basis, and thus are available to any network application without requiring any special modifications.

Applications can also implement their own additional security features as necessary, on top of the underlying IP security.

IP security policy is enabled, enforced, managed, and monitored through a coordinated effort of several z/OS Communications Server components:

- Policy Agent

  The Policy Agent is used to configure IP security on a z/OS system. It reads the configuration files that contain the IP security policy configuration statements, checks them for errors, and installs them into the IKE daemon and the TCP/IP stack.

- Internet Key Exchange daemon (IKED)

  The Internet Key Exchange daemon is responsible for retrieving IP security policy from Policy Agent, and dynamically managing keys that are associated with dynamic IPSec VPNs. This daemon also provides network management capabilities for IP security aspects of local TCP/IP stacks.

- Network Security Services daemon (NSSD)

  The Network Security Services daemon provides the NSS server functionality. It provides the NSS IPSec certificate service to perform digital signature creation and verification operations on behalf of NSS IPSec clients. The NSS IPSec certificate service is used by the IKED during phase 1 negotiations when digital signature authentication is required.

- TCP/IP stack

  The stack maintains a list of currently active IP filters and IPSec Security Associations, actively filters network traffic, controls encryption and decryption of network data, and maintains counters that are associated with an IPSec Security Association lifetime.

8.4.9.7 Management of Communications Server Functions

z/OS provides some basic configuration data sets for TCP/IP and TCP/IP based protocols. Those configuration data sets that are also related to security are:

- PROFILE.TCPIP

  Provides TCP/IP initialization parameters and specifications for network interfaces and routing.

- TCPIP.DATA

  Provides parameters for TCP/IP based client and server programs.

- Additional Communications Server configuration information (e.g., IPSec and AT-TLS) exists in policy files accessed via the Communications Server Policy Agent.
- The IKE daemon, NSS server, Defense Manager daemon, and Policy Agent also have their own configuration files.

Configuration statements in those data sets define the properties (including security properties) of the TCP/IP protocol itself as well as the main protocol server.

Manual IPSec Configuration

the IP security policy configuration files can manually created by coding all of the required statements in a z/OS UNIX file or MVS data set. There are a large number of configuration options provided by IP security policy statements that permit advanced users to carefully fine-tune IP security policy on a per-stack basis. For a dynamic tunnel, the choice of IPSec protocol is configured using the IpDataOffer statement in an IP security policy configuration file. For a manual tunnel, the choice of IPSec protocol is configured using the IpManVpnAction statement in an IP security policy configuration file.

For more details about the IpDataOffer statement and the IpManVpnAction statement, see z/OS Communications Server: IP Configuration Reference.

Communications Server ipsec Command Interface

The Communications Server provides a command named ipsec that allows authorized users to query information about IP filters, defensive filters and IPSec security associations. It also allows authorized users to activate or deactivate IPSec functions, affect which IP filters are loaded, and create, update and delete defensive filters. The administrator can control access to this command by granting READ access to the following resources in the SERVAUTH class:

- EZB.IPSECCMD.sysname.tcpname.DISPLAY allows clients to display information about IP filters, per-stack defensive filters and IPSec security associations {SM.4::SM-R10-CS-IPSECCMD-1}.

- EZB.IPSECCMD.sysname.tcpname.CONTROL allows clients to reload or refresh IP filters, create, update or delete per-stack defensive filters and activate or deactivate IPSec security associations {SM.4::SM-R10-CS-IPSECCMD-3}.
- EZB.IPSECCMD.sysname.DMD_GLOBAL.DISPLAY allows clients to display information about global defensive filters {SM.4::SM-R10-CS-IPSECCMD-4}.
- EZB.IPSECCMD.sysname.DMD_GLOBAL.CONTROL allows clients to create, update or delete global defensive filters {SM.4::SM-R10-CS-IPSECCMD-6}.

Communications Server Network Management Interface

The Communications Server provides, via the IKE daemon, a network management interface (NMI) that allows local applications to query information about IP filters and IPSec security associations. It also allows applications to activate or deactivate IPSec functions. The IKE daemon provides this information via a UNIX (not TCP/IP) socket. The administrator can control access to this interface by granting READ access to the following resources in the SERVAUTH class:

- EZB.NETMGMT.sysname.tcpname.IPSEC.DISPLAY allows clients to display information about IPSec filtering and security associations. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface {SM.4::SM-R10-CS-SECMON-1}.
- EZB.NETMGMT.sysname.tcpname.IPSEC.CONTROL allows clients to issue management requests to activate, deactivate, or modify IPSec security associations. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface. {SM.4::SM-R10-CS-SECMON-3}.
- EZB.NETMGMT.sysname.sysname.IKED.DISPLAY allows clients to display information about IKE daemon usage of the Network Security Services (NSS) client functions via the NMI or the ipsec command with the -w option. If not defined, applications must run with UID(0) or access to BPX.SUPERUSER in order to use the interface. {SM.4::SM-R10-CS-SECMON-4}.

Additionally, the Network Security Services (NSS) server provides a network management interface that allows a central administrator to monitor and control NSS and IPSec information in a manner similar to that provided by the IKE daemon. For these network management requests, the administrator can use the following SERVAUTH resources to provide protection:

- EZB.NSS.sysname.clientname.IPSEC.NETMGMT allows clients to register with the NSS server for IPSec network management services {SM.4::SM-R9-CS-NSS-1}.
- EZB.NETMGMT.sysname.clientname.IPSEC.DISPLAY allows clients to display IPSec-related information via the NSS NMI or the ipsec command with the –z option {SM.4::SM-R9-CS-NSS-2}.
- EZB.NETMGMT.sysname.clientname.IPSEC.CONTROL allows clients to issue management requests to activate, deactivate, or modify IPSec security associations via the NSS NMI or the ipsec command with the –z option {SM.4::SM-R9-CS-NSS-3}.

- EZB.NETMGMT.sysname.sysname.NSS.DISPLAY allows clients to display information about current NSS client connections to the NSS server via the NSS NMI or the ipsec command with the –x option {SM.4::SM-R9-CS-NSS-4}.

Communications Server Policy Agent

The Communications Server provides a Policy Agent that can act in any of several roles, depending on configuration options:

- The Policy Agent may act as the Policy Definition Point (PDP) on a single system, installing policies in one or more z/OS Communications Server stacks {SM.4::SM-R9-CS-POLCEN-1}.
- The Policy Agent may act as a centralized policy server, providing PDP services for one or more remote policy clients {SM.4::SM-R9-CS-POLCEN-2}.
- The Policy Agent may act as a policy client, retrieving remote policies from the policy server. Each stack in a Common INET (CINET) environment acts as a separate policy client {SM.4::SM-R9-CS-POLCEN-3}. Communications between the policy client and the policy server may optionally be secured by AT-TLS {SM.4::SM-R10-CS-POLCEN-11}.

A single Policy Agent may act as a policy client or a policy server, but not both {SM.4::SM-R9-CS-POLCEN-11}.

Policies may be defined in several different ways. When acting as the PDP for a single system, Policy Agent can read policy definitions from local configuration files, a central repository that uses the Lightweight Directory Access Protocol (LDAP), or both {SM.4::SM-R9-CS-POLCEN-4}.

When acting as a policy server, Policy Agent also acts as a PDP for the local system, and so can read policies from local configuration files or an LDAP server, and install them in local stacks {SM.4::SM-R9-CS-POLCEN-6}. But it also reads policies from local configuration files on behalf of policy clients. These policies are retrieved by policy clients, but are not installed in the local stacks on the policy server {SM.4::SM-R9-CS-POLCEN-7}.

When acting as a policy client, Policy Agent retrieves remote policies from the policy server, and can also use local policies from configuration files or an LDAP server {SM.4::SM-R9-CS-POLCEN-8}.

The choice of local or remote policies can be made separately for each type of supported policy: Quality of Service (QoS), Intrusion Detection (IDS), Policy-Based Routing (PBR), IPSec, or AT-TLS {SM.4::SM-R9-CS-POLCEN-9}. For a given policy type, all policies are obtained either locally or remotely {SM.4::SM-R9-CS-POLCEN-10}.

When acting as a policy server, the policy agent will:

- First, authenticate its clients using a RACF user ID and password or PassTicket {SM.4::IA-R9-CS-POLCEN-1}.
- Then, authorize retrieval of policy data, requiring READ access to policy agent resources in the SERVAUTH class. These resources must be protected or retrieval will fail {SM.4::AC-R9-CS-POLCEN-1}. They have the form EZB.PAGENT.sysname.image.ptype {SM.4::AC-R9-CS-POLCEN-2} where
    - Sysname is the system name defined in the sysplex

- Image is the TCP name or policy client name
- Ptype is either QOS, IDS, IPSEC, or (for AT-TLS) TTLS.

8.5 TSFI

IBM has documented most of the functional specification in their public documentation. This documentation consists of a large number of documents that serve as guidance documentation, functional specification and design documentation. Often all those functions are combined in a single document.

The following interface types exist:

- Supervisor Calls (including extended services SVCs): They are invoked using the SVC instruction described in [ZARCH]. They represent a legacy form of entering a privileged processor state from a user program. While in theory a program could directly issue the SVC instructions, the normal "method of use" is to use a macro that generates the code that constructs the parameter list and calls the SVC. The public documentation describes the macros and how to use them in an assembler language program. The format of the parameter lists when invoking the SVC instruction is defined in [MVS.DIAG]. Extended services SVCs are just regular SVCs that have a number of subfunctions where the subfunction to call is one of the parameter. The structure of the rest of the parameter list itself usually depends on the subfunction and therefore there is usually one separate macro for each subfunction. The following are security enforcing TSFI relevant for the functions described that are implemented as SVCs with the pointer to the manual describing the related programming interfaces:
    - SVC 4: GETMAIN described in 'DFSMS Macro Instructions for Data Sets'
    - SVC 13: OPEN described in 'DFSMS Macro Instructions for Data Sets'
    - SVC 16: OPEN (TYPE=J), described in 'DFSMS Macro Instructions for Data Sets'
    - SVC 29: SCRATCH described in 'DFSMSdfp Advanced Services'
    - SVC 30: RENAME described in 'DFSMSdfp Advanced Services'
    - SVC 130: RACHECK described in 'Security Server RACROUTE Macro Reference'
    - SVC 131: RACINIT described in 'Security Server RACROUTE Macro Reference'
    - SVC 132: RACLIST described in 'Security Server RACROUTE Macro Reference'
    - SVC 133: RACDEF described in 'Security Server RACROUTE Macro Reference'

  Note that the RACF SVCs (SVCs 130-133) can not be invoked by unauthorized programs. They are therefore not TSFI in the usual sense, since they are restricted to be used by trusted resource managers to define a resource and its protection to RACF, verify a user's access to a resource, or to authenticate a user. Also note that those services should be invoked using the (newer) RACROUTE macro instead of the macro names listed above. A mapping of those services to the RACROUTE macro can be found in Appendix A of 'Security Server RACROUTE Macro Reference'.

- Program Calls (PCs): PCs are invoked using the Program Call instruction. The Program Call (PC) instruction is the preferred way to have a user level program call functions of the TSF and switching state. The PC instruction is defined in [ZARCH]. As with SVCs there is usually a macro that generates the code that constructs the parameter list and then calls the PC. Since there are functions that have a "legacy" version as well as a "regular" version, the same macro may generate code calling either a SVC (if used with "legacy" parameter) or a PC (if used with parameter not supported by the legacy version).
  There are many PCs related to security services without a programmer using the related programming interfaces knowing this. In addition there are many services (e. g. those for the UNIX System Services) that are mapped to a single PC where the parameters the PC is called with determine the service to be provided. One should also keep in mind that the PC is the mechanism used to call services from a trusted address space (which will 'register' the PCs it exposes upon start-up of the address space. In addition, as described in [ZARCH] the Access Key Mask defined when registering the PC determines the hardware state (User or Supervisor state, storage key mask) the caller needs to have. This is a mechanism that allows the hardware to perform some checks if the caller is authorized to call the service. In addition the function called can perform additional checks, e. g. if the calling application executes with APF authorization or if the caller has specific privileges managed by RACF.
  The documents describing security-related TSFI that may be relevant are:
    - for UNIX System Services: UNIX System Services Programming: Assembler Callable Services Reference
    - for Communications Server: IP Programmer's Guide and Reference
    - for the ICSF cryptographic services: Cryptographic Services - Integrated Cryptographic Service Facility - Application Programmer's Guide
    - for BCP: MVS Programming: Assembler Services Reference, Volume 1 and Volume 2
    - for PKI Services: Cryptographic Services - PKI Services Guide and Reference
    - for Kerberos: Integrated Security Services Network Authentication Service Programming
- Trusted Programs / Aliases: They can be called as any program from a library but will only execute as authorized programs when they are called as the first program in a job step. Therefore their "method of use" (when running authorized) is to have them called by a job in the first EXEC statement.
  There are a number of trusted programs that are designed be started as their own address space by an authorized operator. Those trusted programs will fail if started otherwise. This for example also applies for all network daemons. Then there are a number of trusted programs that perform non-security related services. The remaining trusted programs that perform security related services are those for audit trail evaluation. Those programs and their usage are described in the document Security Server RACF Auditor's Guide.
- Operator and JES2 Commands: They can be called by a user in the role of an operator provided he has the required privileges to call the command (operator commands are subject to RACF access control). They can be entered at a system console or can be

entered in the JCL // COMMAND statement.
The general operator commands are described in the document MVS System Commands.
The JES2 operator commands are described in the document JES2 commands.
In addition to those documents there are component specific documents describing administrative commands related to the component. Those are:

    - for LDAP: IBM Tivoli Directory Server Administration and Use for z/OS
    - for Print Services: Infoprint Server Operation and Administration
    - for Communications Server: IP System Administrator's Commands
    - for ICSF: Cryptographic Services Integrated Cryptographic Service Facility Administrator's Guide
    - for PKI Services: Cryptographic Services - PKI Services Guide and Reference
    - for DFSMS: multiple documents titled DFSMSxxx Storage Administration where 'xxx' identifies the DFSMS subcomponent (like 'dss' or 'rmm')
    - for zFS: Distributed File Service zFS Administration
    - for Kerberos: Integrated Security Services Network Authentication Service Administration
    - for RACF: RACF Command Language Reference
    - for TSO: two manuals: TSO/E Command Reference and TSO/E Administration
    - for UNIX System Services: UNIX System Services Command Reference

- JCL statements: They are used to define the Job Control. They are written down in a data set that is then used to start job by either using the START operator command or by submitting the job from a terminal session using the TSO SUBMIT command. The specification of the JCL statements are defined in the document MVS JCL Reference.
- Configuration files and data sets: as the name states they are data sets (usually with a predefined name) that are read by the component they belong to when it starts up. The required protection for those configuration files and data sets is described in the documentation. For most components the configuration is defined in the data sets SYS1.PARMLIB. The members of this data set and how they configure the system is defined in the document MVS Initialization and Tuning Reference. Configuration files for network services are not defined in SYS1.PARMLIB but in separate files which are described in the document Communications Server: IP Configuration Reference
- Network services: those are communication services the TOE offers to external systems. On the network side the "method of use" is the communication protocol. For the services described in the document Communications Server: IP Configuration Guide (with the specifics how to configure those in the document Communications Server: IP Configurations Reference).

Using the services described in those documents the evaluator is able to develop programs or scripts or to execute commands that test the claims made in the SFRs.

For additional information the evaluator may just read the other public documents IBM has published for z/OS V2R1. For example, if the evaluator needs to know the reason for an error message he receives during his testing, he may just consult one of the following publications:

- Communication Server: (4 books that describe "Communication Server IP Messages", one book that describes "Communication Server SNA Messages")
- Cryptographic Services: one book titled "ICSF Messages"
- Distributed File Services: one book titled "Distributed File Service Messages and Codes".
- DFSORT: one book titled "DFSORT Messages, Codes and Diagnosis Guide"
- HCD/HCM: one book titled "Hardware Configuration Definition Messages"
- ISPF: one book titled "ISPF Messages and Codes"
- JES2: one book titled "JES2 Messages"
- MVS: 10 (ten!) books titled "MVS System Messages Vol x" (where x is a number between 1 and ten). The total number of pages of just those ten books is 8662.
- RACF: one book titled "Security Server RACF Messages and Codes"
- SMP/E: one book titled "SMP/E for z/OS Messages, Codes, and Diagnosis"
- TSO/E: one book titled "TSO/E Messages"
- UNIX System Services: one book titled "UNIX System Services Messages and Codes".
- Directory Server: Error messages are described in a separate document "IBM Tivoli Directory Server Messages and Codes".

Happy reading!

References

[CC]	Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model Part 2: Security functional components Part 3: Security assurance components September 2012, Version 3.1 Revision 4
[CEM]	Common Methodology for Information Technology Security Evaluation Evaluation Methodology September 2012, Version 3.1 Revision 4
[GPOSPP]	General-Purpose Operating System Protection Profile. Version 3.9 as of 2012-12-06; strict conformance.