

Rocket Model 204 Security Interfaces Manual

Version 7 Release 4.0

May 2012
204-0704-SECU-01



Notices

Edition

Publication date: May 2012

Book number: 204-0704-SECU-01

Product version: Rocket Model 204 Security Interfaces Manual Version 7 Release 4.0

Copyright

© Computer Corporation of America 1989-2012. All Rights Reserved.

Computer Corporation of America is a wholly-owned subsidiary of Rocket Software, Inc.

Trademarks

Rocket is a registered trademark of Rocket Software, Inc. For a list of Rocket registered trademarks go to: www.rocketsoftware.com/about/legal. All other products or services mentioned in this document may be covered by the trademarks, service marks, or product names of their respective owners.

License agreement

This software and the associated documentation are proprietary and confidential to Rocket Software, Inc., are furnished under license, and may be used and copied only in accordance with the terms of such license.

Note

This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulation should be followed when exporting this product.

Contact information

Web Site: www.rocketsoftware.com

Rocket Software, Inc. Headquarters
77 4th Avenue, Suite 100
Waltham, MA 02451-1468
USA
Tel: +1.617.614.4321
Fax: +1.617.630.7100

Contacting Technical Support

If you have current support and maintenance agreements with Rocket Software and CCA, contact Rocket Software Technical support by email or by telephone:

Email: m204support@rocketsoftware.com

Telephone :

North America +1.800.755.4222

United Kingdom/Europe +44 (0) 20 8867 6153

Alternatively, you can access the Rocket Customer Portal and report a problem, download an update, or read answers to FAQs. You will be prompted to log in with the credentials supplied as part of your product maintenance agreement.

To log in to the Rocket Customer Portal, go to:

<http://www.rocketsoftware.com/support>

Contents

About this manual

Audience	xi
Model 204 documentation set	xi
Documentation conventions	xi

1 Introduction

In this chapter.....	1
Overview	1
Software support	1
Error messages	1
General installation notes.....	2

2 CA-ACF2 MVS Interface

In this chapter.....	3
Overview	3
CA-ACF2 MVS	3
Model 204 CA-ACF2 MVS Interface	4
CA-ACF2 MVS Interface components.....	4
Brief introduction to CA-ACF2 MVS	5
CA-ACF2 processing.....	5
System manager options	6
CA-ACF2 MVS and the CA-ACF2 MVS Interface	6
Logonid record	6
User identification string	7
UID masks.....	7
Data access control.....	7
Generalized Resource control.....	7
Running the CA-ACF2 MVS Interface with Model 204	8
Model 204 user privileges	8
Managing the authorization process	8
Using the CA-ACF2 MVS Interface	9
LOGIN or LOGON command	9
IFSTRT function	10
IFLOG function within IFAM1	11
Dynamic allocation considerations	11
Job submission considerations	11
Sequential and VSAM data set considerations	12
Login processing.....	13
When a user logs in	13
Verifying the login account	14
CCASTAT considerations	14
User 0 login	14
AUTHCTL command.....	14

Using trusted login for CRAM users	15
Maintaining the CA-ACF2 MVS Interface.....	17
Defining user privileges	17
User privilege names.....	17
CCASTAT users.....	18
Defining corresponding Generalized Resource Rules	18
Preparing the ACF2PARM parameter module with ACF2GEN	19
SECPLIST parameter	20
ACF2GEN macro	20
Sample ACF2PARM module.....	23
Installing the Model 204 CA-ACF2 MVS Interface	24
Terminal security considerations.....	24
Decrypting CA-ACF2.....	24
Assembling SBA2OS	25
Assembling ACF2PARM	25
Link-editing Model 204	25
Defining Model 204 as a CA-ACF2 MUSASS.....	26
Adding the @MUSASS macro to ACFFDR.....	27
Defining Logonid fields for Model 204.....	27
Inserting a CA-ACF2 default Logonid.....	29
Defining Model 204 user privileges	29
ACF2 Interface storage requirements	29
Converting users from CCASTAT security to CA-ACF2 security	30
Setting the SECTRLOG parameter	30
Conversion tasks and considerations.....	31

3 CA-ACF2 VM Interface

In this chapter.....	33
Overview	34
CA-ACF2 VM	34
Model 204 CA-ACF2 VM interface	34
CA-ACF2 VM Interface configurations	35
Inline validation	35
Service machine validation.....	35
Brief introduction to CA-ACF2 VM	38
CA-ACF2 processing.....	38
System manager options	38
CA-ACF2 VM and the Model 204 CA-ACF2 VM Interface	38
Logonid record	39
User identification string.....	39
CA-ACF2 Generalized Resource Control.....	40
Running the CA-ACF2 VM Interface with Model 204.....	40
Managing the authorization process	40
ACFDIAG macro	41
ACF2CMS module	41
Using the CA-ACF2 VM Interface	41
LOGIN or LOGON command	42
Account validation	43
Source validation.....	43
IFSTRT function	43

IFLOG function within IFAM1	43
Login processing	43
Online login processing	44
IFAM2 login processing	44
CA-AFC2 Logonid record	44
CCASTAT and CA-ACF2	45
User 0 login	46
User 0 processing	46
Single-user Onlines	47
Maintaining the CA-ACF2 VM Interface	47
Defining user privileges	47
Security service machine	48
Security service machine commands	49
CA-ACF2 statistics	51
Preparing ACF2PARM with ACF2GEN	52
SECPLIST parameter	52
ACF2GEN macro	53
Sample ACFPARM module	56
Installing the CA-ACF2 VM Interface	56
Modifying the VM directory	58
Creating the PROFILE EXECs	59
Decrypting object modules	60
Using M204CRYP	60
Creating Model 204 modules and saved segments	60
Generating the Model 204 modules	60
Generating the security module	61
Applying ACFDIAGC	61
Saving Model 204 saved segments	61
Creating ACF2PARM	61
Inserting the SECUR204 CA-ACF2 Logonid	62
Updating Model 204 CA-ACF2 Logonids	62
Writing execute-only rules	63
Sample rules	63
Updating the ACFFDR and LIDREC DSECT	64
Updating @RESTYPE macro	64
Adding @CFDE macros	64
Updating the LIDREC dsect	65
Giving values to Model 204 Logonid record fields	66
Writing rules	66
Writing SECPLIST rules	67
Defining Model 204 user privileges	67
Validating the login	68
Validating the account	68
Writing source entry rules	68
Updating Model 204 EXECs	69
Updating Model 204 CCAIN files	69

4 Model 204 Security Server (formerly RACF) Interface

In this chapter	71
Overview	71

Security Server.....	71
Model 204 Security Server Interface	72
Model 204 Security Server Interface components	72
Brief introduction to Security Server.....	73
Security Server processing	74
Authorization checking	74
Global access checking	75
Using profiles to protect resources	75
Generic profile	76
Discrete profile	76
Running the Security Server Interface with Model 204	77
Using the Security Server Interface	77
LOGIN or LOGON command	78
User ID information	78
Security Server processing	79
IFSTRT function	79
IFLOG function within IFAM1	79
Dynamic allocation considerations	79
Job submission considerations	80
Sequential and VSAM data set considerations	81
Login processing.....	82
Login processing	82
User 0 login	83
AUTHCTL command.....	83
Using trusted login for CRAM users	84
Maintaining the Security Server Interface	86
Defining user privileges	86
Defining corresponding generic profiles	87
Installing the Security Server Interface	88
Terminal security considerations.....	88
Decrypting Security Server.....	89
SECPLIST parameter	89
Preparing a RACFPARM parameter module with RACFGEN	90
Model 204 link-editing requirements	93
Defining Model 204 to Security Server.....	94
Defining Model 204 users in Security Server	94
Security Server default user ID	94
Setting up the SECTRLOG parameter for trusted login	95
Conversion tasks and considerations.....	95

5 CA-Top Secret Interface

In this chapter.....	97
Overview	97
CA-Top Secret	97
Model 204 CA-Top Secret Interface.....	98
CA-Top Secret components.....	99
Brief introduction to CA-Top Secret	99
CA-Top Secret processing	100
Accessor ID (ACID).....	100
System organizational elements	100

Facilities	102
Resources	102
Access controls	103
Access levels	103
Ownership and authorization	104
Running the CA-Top Secret Interface with Model 204	105
Managing the authorization process	105
Using the CA-Top Secret Interface	106
LOGIN or LOGON command	106
IFSTRT function	107
IFLOG function within IFAM1	107
Dynamic allocation considerations	107
Job submission considerations	108
Sequential and VSAM data set considerations	108
Login processing	109
Login processing	109
Messages	110
User 0 login	111
AUTHCTL command	111
Using trusted login for CRAM users	112
Maintaining the CA-Top Secret Interface	113
Defining Model 204 to CA-Top Secret	114
Defining the Model 204 ACID	115
Defining the Model 204 default user ACID	116
Allowing users to log in to Model 204	116
Defining user privileges	117
Sample definition of Model 204 pseudo data set names	118
CA-Top Secret default ACID	119
Terminal security considerations	119
Installing the CA-Top Secret Interface	120
Decrypting the CA-Top Secret Interface	120
SECPLIST parameter	120
Preparing a TOPSPARM parameter module with TOPSGEN	121
Model 204 link-editing requirements	124
Setting the SECTRLOG parameter for trusted login	124
Conversion tasks and considerations	125

6 SQL Security Exits

In this chapter	127
Overview	127
Requirements	128
Security exit functions	128
SQL security exit restrictions	128
Using the CDTB module	129
DDLPRIV exit	130
Security for DDL statements	131
DMLPRIV exit	132
Parameters passed to SQL	133
Return codes for DDLPRIV and DMLPRIV security exits	134
Installing the SQL security exits	135

Index

About this manual

Audience

This manual is directed to the person or people responsible for installing and maintaining security interfaces at your site.

Model 204 documentation set

The complete commercially released documentation for the latest version of Model 204 is available for download from the Rocket M204 customer portal.

To access the Rocket Model 204 documentation:

1. Navigate to:
<http://www.rocketsoftware.com/m204>
2. From the drop-down menu, select **Products > Model 204 > Documentation**.
3. Click the link to the current release and select the document you want from the list.
4. Click the .zip file containing the document.
5. Choose whether to open or save the document:
 - Select **Open** and double-click the pdf file to open the document.
 - Select **Save as** and select a location to save the zip file to.

Documentation conventions

This manual uses the following standard notation conventions in statement syntax and examples:

Convention	Description
TABLE	Uppercase represents a keyword that you must enter exactly as shown.
TABLE <i>tablename</i>	In text, italics are used for variables and for emphasis. In examples, italics denote a variable value that you must supply. In this example, you must supply a value for <i>tablename</i> .
READ [SCREEN]	Square brackets ([]) enclose an optional argument or portion of an argument. In this case, specify READ or READ SCREEN.
UNIQUE PRIMARY KEY	A vertical bar () separates alternative options. In this example, specify either UNIQUE or PRIMARY KEY.

Convention	Description
TRUST <u>NOTRUST</u>	Underlining indicates the default. In this example, NOTRUST is the default.
IS {NOT LIKE}	Braces ({ }) indicate that one of the enclosed alternatives is required. In this example, you must specify either IS NOT or IS LIKE.
item ...	An ellipsis (. . .) indicates that you can repeat the preceding item.
item ,...	An ellipsis preceded by a comma indicates that a comma is required to separate repeated items.
All other symbols	In syntax, all other symbols (such as parentheses) are literal syntactic elements and must appear as shown.
<i>nested-key</i> ::= <i>column_name</i>	A double colon followed by an equal sign indicates an equivalence. In this case, <i>nested-key</i> is equivalent to <i>column_name</i> .
Enter your account: sales11	In examples that include both system-supplied and user-entered text, or system prompts and user commands, boldface indicates what you enter. In this example, the system prompts for an account and the user enters sales11 .
File > Save As	A right angle bracket (>) identifies the sequence of actions that you perform to select a command from a pull-down menu. In this example, select the Save As command from the File menu.
EDIT	Partial bolding indicates a usable abbreviation, such as E for EDIT in this example.

1

Introduction

In this chapter

- Overview

Overview

In addition to Rocket Model 204 security, three types of security interfaces are available to Model 204 users:

- CA-ACF2
- Security Server (formerly RACF)
- CA-Top Secret

In addition, if your site uses a security interface along with Connect★, you can write your own security exits, which allows you to use your security interface instead of standard Model 204 SQL security. This manual details each of the security interfaces and the user security exits and describes the steps necessary to install them.

Software support

Model 204 supports the current releases of CA-ACF2, Security Server, and CA-Top Secret under z/OS and CA-ACF2 VM.

Error messages

Each of the security interfaces generates Model 204 error and informational messages. If you receive an error message and need further information, refer to the *Rocket Model 204 Messages Manual*.

General installation notes

Model 204 is distributed with all the materials necessary to install and use Model 204 as well as any separately purchased features. The Security Interface object modules described in this manual are distributed in encrypted form. Until they are decrypted, they are useless to any utilities that process object modules, such as the linkage editor and loader.

2

CA-ACF2 MVS Interface

In this chapter

- Overview
- Brief introduction to CA-ACF2 MVS
- Running the CA-ACF2 MVS Interface with Model 204
- Using the CA-ACF2 MVS Interface
- Login processing
- Maintaining the CA-ACF2 MVS Interface
- Preparing the ACF2PARM parameter module with ACF2GEN
- Installing the Model 204 CA-ACF2 MVS Interface

Overview

This chapter presents an overview of how the Access Control Facility/2 (ACF2) Interface from Computer Associates, Inc. works in the z/OS environment and describes the impact of the CA-ACF2 MVS Interface on the Model 204 user, system manager, and CA-ACF2 installation security officer. Installation requirements and a conversion task summary also are presented.

CA-ACF2 MVS

CA-ACF2 MVS provides comprehensive data security functions for the IBM z/OS operating system. CA-ACF2 MVS protects all data and installation-defined resources by default, sharing data only when explicitly requested to by the owner of the data or by a security officer. CA-ACF2 MVS differs from other

security systems that force you to define what is to be protected, rather than what is to be shared.

Model 204 CA-ACF2 MVS Interface

The Model 204 CA-ACF2 MVS Interface provides for an orderly migration from Model 204 security to CA-ACF2 security and allows an installation to use a combination of either or both security methods.

The interface provides the tools and instructions needed to:

- Define Model 204 as a CA-ACF2 Multiuser Single Address Space System (MUSASS)
- Define the meaning of Model 204 user privileges
- Define CA-ACF2 Generalized Resource Rules for CA-ACF2 authorization of Model 204 user login privileges
- Define an installation default user priority class or the offset of a priority class byte within the user's CA-ACF2 Logonid record
- Define the offset and length of a default login account within a CA-ACF2 Logonid record, and provide account authorization services
- Define the offset and bit within the CA-ACF2 Logonid record that determines whether or not a user can log in to a particular Model 204 job
- Define the offset and length within the CA-ACF2 Logonid record that contains data to be used as the user's record security key field
- Activate the external CA-ACF2 Interface security mechanism

In summary, the CA-ACF2 MVS Interface provides all the facilities an installation site needs to implement CA-ACF2 security within the Model 204 environment.

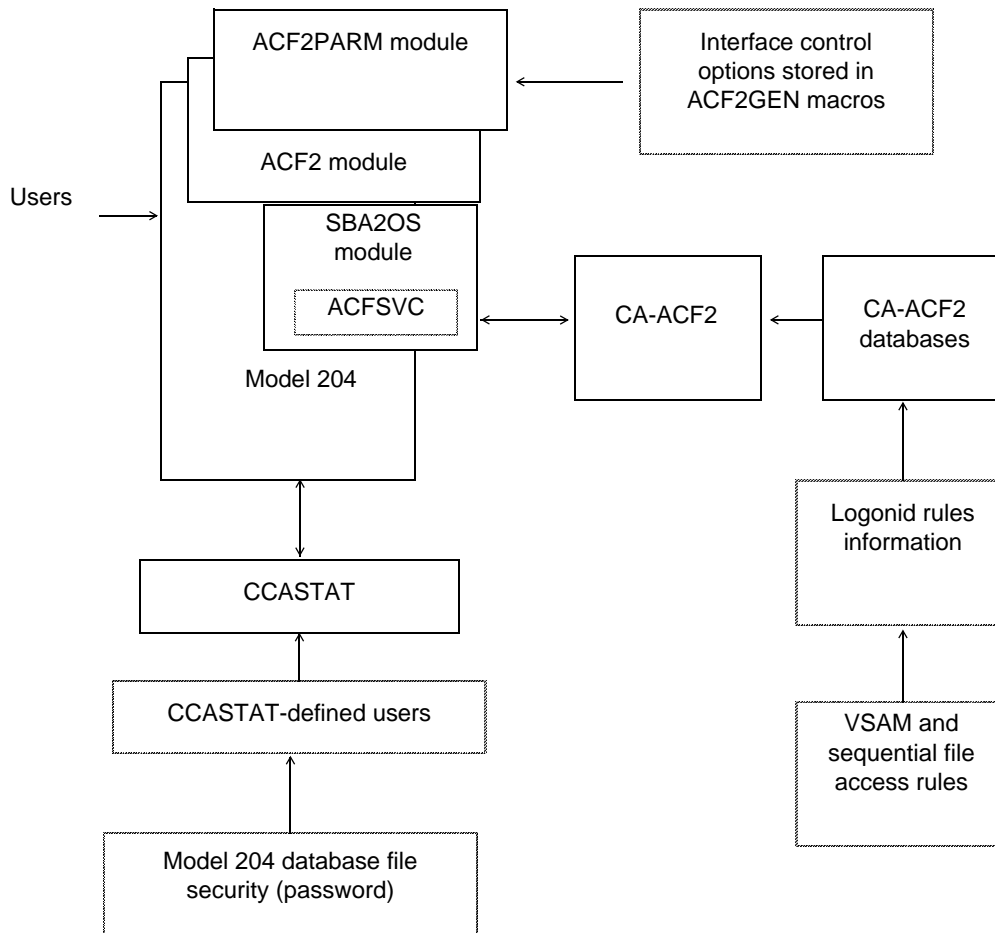
CA-ACF2 MVS Interface components

Figure 2-1 illustrates the components of the CA-ACF2 MVS Interface, which consists of:

- Model 204
- ACF2 Interface module, which is linked into Model 204
- ACF2PARM module, which can be linked with Model 204 or loaded dynamically at initialization
- SBA20S module, which is linked into Model 204 and contains the ACFSVC macro that invokes the CA-ACF2 SVC
- Model 204 system file, CCASTAT, which contains user IDs subject to Model 204 security

- CA-ACF2 MVS, which retrieves data from its VSAM databases and returns it to Model 204 for processing

Figure 2-1. CA-ACF2 MVS Interface



Brief introduction to CA-ACF2 MVS

This section provides a brief summary of key CA-ACF2 MVS features that are used in the discussion of the Model 204 CA-ACF2 MVS Interface.

For more information about CA-ACF2 MVS, see the documentation associated with that product.

CA-ACF2 processing

When a user logs in to the z/OS environment, CA-ACF2 MVS performs the following functions:

- Validates the password found for the Logonid in the CA-ACF2 Logonid database
- Checks for password expiration

- Optionally, ensures that the user is accessing the system from a specific terminal or other source
- Optionally, ensures that the user is only accessing the system during certain dates and times

System manager options

The system manager has the following additional options when a user logs in to Model 204 with a CA-ACF2 Logonid:

- Require each user's login to go through Login or Account validation, defined by the ACF2GEN macro
- Set the priority for each user through the user's Logonid record or set one priority for all CA-ACF2 users
- Change the user's record security key and not default to the Model 204 user ID

CA-ACF2 MVS and the CA-ACF2 MVS Interface

The Model 204 CA-ACF2 MVS Interface uses the following CA-ACF2 SVC macro calls to send validation requests to CA-ACF2:

ACFSVCA	Performs login validation requests
ACFSVCS	Performs DASD validation requests of sequential data, VSAM files, and new DASD space allocation.

Since validation requests are performed in the main Task Control Block (TCB) in the Model 204 job's address space, Model 204 must be defined as a MUSASS (Multiuser Single Address Space System) to allow multiple user validation in the same address space.

Logonid record

CA-ACF2 MVS maintains a Logonid record, or LIDREC, for each user on the system and each Logonid has an associated password. All passwords are stored in an encrypted format that cannot be reversed. CA-ACF2 MVS validates users by acquiring and processing the Logonid and password during TSO LOGON, IMS SIGNON, batch job submission, CICS SIGNON, and other TP product signons (COM-LETE or ROSCOE, for example).

Users are defined only once to CA-ACF2 MVS, regardless of how many subsystems they access. To define differing authorizations for different subsystems, users can be identified via a UID (user Identification) string.

User identification string

A UID string is a character string, up to 24 bytes long, constructed from the user's Logonid and other site-definable fields such as department number or employee ID. CA-ACF2 MVS compares the user's UID with the UID masks specified in a rule set to determine whether a user can have access to a resource.

UID masks

A UID mask can be placed on any rule to allow or prevent access to a resource by a group of users based on their UID string. For example, a rule in the Rules database can have a UID mask that allows everyone with a department code of DEV to have access to a certain resource.

Data access control

CA-ACF2 MVS protects all data by default from all users except the data owner. Each Logonid on the system has an owned data set prefix that controls this function.

When any user tries to access a data set, the CA-ACF2 high-level index is compared to the value of the prefix field in the Logonid record. If the two are equal, access is automatically allowed. If not, CA-ACF2 MVS searches for an access rule set that governs access to that index. If an access rule does not exist for that index, access is denied. If CA-ACF2 MVS obtains an access rule set, the access rules are interpreted to locate one that matches the currently existing environment. If a matching access rule is not found, access is denied.

If a matching access rule is located, the purpose of the access request is compared with the read, write, allocate, and execute-only permission values specified in the access rule, and access is either allowed, allowed and journaled, or prevented and journaled.

CA-ACF2 MVS uses character string patterns to perform the validation matching function. The patterns can define the data set, volume, UID string, program library, and program name fields in the Logonid to be verified.

Generalized Resource control

CA-ACF2's Generalized Resource Rules are similar to data access control rules. However, they control access to resources other than files. These system resources can be defined either by CA-ACF2 MVS or by each site locally.

By allowing each site to specify which local resources require security protection, the rules for a data processing installation can be complex enough to define any arbitrary or real system resources needed. Generalized Resource Rules typically are used for TSO account numbers, IMS transaction names, and CICS transaction names.

Model 204 user login privileges are defined as general system resources, and local installation rules control who can use those resources. These rules are written by the Model 204 system manager and the CA-ACF2 MVS installation security officer. For more information, see page 18.

Running the CA-ACF2 MVS Interface with Model 204

A Model 204 multiuser Online always runs as a CA-ACF2 MUSASS (Multiuser Single Address Space System). A MUSASS is defined by CA-ACF2 MVS as any program that runs as a single job in one address space that supports more than one user.

Other Model 204 configurations such as BATCH204 and IFAM1 do not normally run as a MUSASS, because only a single user's authorization is in question.

IFAM4 might run as a MUSASS, but each IFAM4 user application must also be APF-authorized in addition to the IFAM4 Load Module, otherwise the multiple IFAM4 threads must log in as the same user.

Model 204 user privileges

In either a MUSASS or single-user environment, Model 204 queries CA-ACF2 MVS for the authority to grant access to various system resources on behalf of the user. User privileges are defined as resources to be protected, and rules are written to determine who is authorized to use of those resources.

Managing the authorization process

In a MUSASS, Model 204 manages the authorization process because only Model 204 can identify which user initiates a request for a resource. Model 204 assists CA-ACF2 MVS in validating those requests and responds to users who issue requests for unavailable resources.

The three distinct areas where Model 204 interfaces with CA-ACF2 MVS are:

- User logins
 - LOGIN/LOGON command
 - IFSTRT function
 - IFLOG function
- User resource requests
 - Job submission (USE \$JOB command)
 - Dynamic allocation of new DASD space (ALLOCATE command)
 - All sequential file and VSAM file opens for READ/WRITE access
 - Sequential data set handling (record I/O)
 - DUMP/RESTORE commands

- OPEN command naming a deferred update data set

Note: Because Model 204 databases are not validated on behalf of the user by the security interface, CA-ACF2 MVS must grant the owner of an address space permission to open Model 204 file data sets for update, regardless of whether the owner has write or read-only privileges. This allows Model 204 to write back updates to the FPL (File Parameter List) page, as required by the database management system, regardless of the Model 204 file open privileges.

- User logouts
 - LOGOUT/LOGOFF command
 - IFFNSH/IFDTHRD function

The CA-ACF2 MVS Interface uses the standard SVC calls supplied by Computer Associates International, Inc. (CA) to validate user requests for Model 204 resources. No source code modifications or exits to the CA-ACF2 MVS software are necessary. However, CA-ACF2 MVS system parameters must be modified as described in “Preparing the ACF2PARM parameter module with ACF2GEN” on page 19.

Using the CA-ACF2 MVS Interface

This section describes how to log in to Model 204 and identifies the changes required for:

- LOGIN/LOGON
- IFSTRT function
- IFLOG function within IFAM1
- Dynamic allocation
- Job submission
- Sequential and VSAM data set handling

LOGIN or LOGON command

The LOGIN/LOGON command allows you to gain access to Model 204. To log in, enter:

LOGIN *userid* [*account*]

or

LOGON *userid* [*account*]

where:

userid	Is a character string that identifies you.
account	Is an optional character string that identifies the account under which you log in.

If Model 204 is providing security authorization checks, every Model 204 user has a user ID, password, and user privileges assigned by the system manager consisting of:

- User ID that identifies you to Model 204
- Password that provides access to the system
- User privileges associated with your user ID and password that define the particular type of access you have
- Default priority class assigned to your user ID

When native Model 204 security is in effect, this user ID information is stored in a record in the system file CCASTAT. This record can be deleted from CCASTAT when the system manager puts the user under CA-ACF2 MVS security. The ACF2PARM module supplies a default Logonid for CCASTAT login access validations. In addition, ACF2PARM defines the following:

- Other CA-ACF2 ID validations to be done at login
- Login validation for the particular Model 204 job
- Defaults and/or location within the your CA-ACF2 Logonid record where the data can be found

When CA-ACF2 MVS is performing login validation, the user ID (that is, the Logonid) must be eight characters or less. The user ID is verified by CA-ACF2 MVS before you can log in to Model 204.

If your installation performs CA-ACF2 account validation, any value entered in the account field is verified via CA-ACF2 services. If you do not enter an account, a default account is supplied from the CA-ACF2 user Logonid record as discussed on page 29.

You can change your CA-ACF2 Logonid password when you log in to Model 204. Refer to the LOGIN or LOGON command discussion in the *Rocket Model 204 Parameter and Command Reference Manual* for more information.

IFSTRT function

In the IFAM1 and IFAM4 environments, the IFSTRT function allocates a Host Language Interface thread for a Host Language program. IFSTRT also establishes the calling convention, performs a user login, and determines whether the thread has update privileges.

When processing the login argument of the IFSTRT call, the user login process follows the rules of a User 0 login (unless you have defined IFAM4 as a MUSASS and APF-authorized all user applications). See “User 0 login” on page 14 for more information about User 0 login.

IFLOG function within IFAM1

The IFLOG function is used only in the IFAM1 environment. It is called following IFSTRT to identify the user when a login is required. This function is necessary in any IFAM1 program where CA-ACF2 MVS validates user authorization.

As with the IFSTRT function, the user login process is the same as the process for User 0 login (refer to page 14).

Dynamic allocation considerations

Dynamic allocation services for new data sets in a CA-ACF2 environment are subject to CA-ACF2 system rules for data set validation. For example, you can restrict data set allocation to only allow certain data set name patterns to be created on certain predefined volumes. The rules for data set access are determined by the CA-ACF2 installation security officer.

To dynamically allocate a *new* data set, you must have ACF2 ALLOC authority in CA-ACF2. If you issue an ALLOCATE command for a new data set and you do not have ALLOC authority, you receive a Model 204 error message.

If you log in to Model 204 via CCASTAT, your allocation privileges are determined by the CA-ACF2 default user’s Logonid authorization.

Note: When you open a new or existing sequential or a VSAM data set, it is validated for read/write access. CA-ACF2 does not perform access authorization on Model 204 data sets, because these data sets are currently protected under Model 204 security.

Job submission considerations

When you issue a USE command to submit a batch job, Model 204 must identify who submitted the job so that CA-ACF2 MVS can properly determine the authorization for that execution. When the CA-ACF2 MVS Interface is running, the following JCL statement is added automatically to the submitted job to identify the user and the user’s level of authorization:

```
//*JOBFROM userid | source
```

where:

userid	Is your Model 204 user ID. CA-ACF2 MVS uses your ID to determine your batch job authorization at execution time.
--------	--

source	Identifies the terminal, using TERMID, from which you submit the job. If you are connected via a CRAM interface, such as TSO, CICS, or IFDIAL, the source becomes the appropriate CRAM channel name that was used to gain access to Model 204.
--------	--

CCASTAT IDs use the default Logonid defined in the ACF2PARM module. The source is the job name, and User 0's source is also the job name.

Neither you nor the system manager can modify this information. If you supply your own `/*JOBFROM userid/source` statement, the system-supplied statement overrides it.

To submit a job, the Model 204 Online requires the correct authority, which means:

- MUSASS ID must have JOBFROM privileges.
- All user IDs must have TSO SUBMIT permission to issue USE \$JOB.
- If the TSO option of JOBCK is specified, a user ID and the MUSASS ID must have JOB privileges to submit jobs with the USE \$JOB command.

Model 204-submitted jobs go directly to the internal reader. If the local site has a TSO SUBMIT exit in effect to enable you to modify submitted jobs, the exit does not receive control.

To submit batch jobs, you must be defined to CA-ACF2 MVS as having the authority to do so. Model 204 checks the standard CA-ACF2 TSO JCL/SUBMIT attribute flag to determine whether or not you have the authority to submit JCL. If you are not authorized to submit a job, you receive a Model 204 error message.

Sequential and VSAM data set considerations

To use a sequential or VSAM data set, you must have the appropriate Model 204 privileges. CA-ACF2 MVS does an authorization check to ensure that you have the appropriate privileges:

With these privileges...	You can...
CA-ACF2 write privileges	Open a deferred update data set or write to the output sequential files of the DUMP, DUMPG, or USE data set commands
CA-ACF2 read privileges	Read sequential or VSAM files from User Language or read the sequential input file specified in a RESTORE or RESTOREG command
ALLOC authority	Dynamically allocate a new data set (any new DASD space)

If you issue a sequential or VSAM data set OPEN command that fails for CA-ACF2 reasons, a Model 204 error message is displayed.

If you log in to Model 204 via CCASTAT, the CA-ACF2 default user's LIDREC determines your authorization.

Model 204 sequential file data sets (such as CCASTAT, CCAAUDIT, CCAJRNL, or CCAJLOG) are also checked to determine if the owner of the address space has the authority to write to them.

The CA-ACF2 Interface directly checks the following Model 204 commands:

- DUMP and DUMPG
- RESTORE and RESTOREG
- USE OUTXXX (DATASET)
- ALLOCATE a new data set
- OPEN DATASET XXXX for input
- OPEN *filename* with deferred update

Login processing

The system manager is responsible for defining and controlling security processing in a Model 204 installation in which CA-ACF2 MVS supplies authorization services. This section describes the login processing performed by the interface.

When a user logs in

When a user logs in, Model 204 acquires the user ID and account and searches CCASTAT to determine whether or not that Model 204 user ID exists. If the user ID is found, Model 204 queries the user for a password and proceeds with authorization processing. If the user ID is not found in CCASTAT and CA-ACF2 MVS security is in effect, Model 204 uses CA-ACF2 facilities to authorize the user login.

Model 204 passes the Logonid and password to CA-ACF2 to verify that the user is allowed to log in. After the user's CA-ACF2 user ID and password are verified, CA-ACF2 presents the user's Logonid record to Model 204.

Next, if the authorization check to use Model 204 is active, Model 204 examines the CA-ACF2 Logonid record to ensure that the user is allowed to use Model 204. If this check is successful, account processing occurs.

If a default account location and length were specified at installation time and the user does not enter an account, the default account is retrieved from the user's Logonid record and the user is allowed in to Model 204.

Verifying the login account

If the installation has chosen to verify the account, the value entered is compared to the user's default account. If the values match, the user is allowed in to Model 204. If the values do not match, standard CA-ACF2 account validation is performed to ensure that the user is allowed to log in under the specified account.

If CA-ACF2 denies system access because of an invalid password or account, or if the user is not authorized to use Model 204, no login occurs and Model 204 issues an error message. In addition, any login failure caused by entering an invalid password counts against the user's CA-ACF2 password violation counter. This counter is maintained by CA-ACF2 and, when the count reaches an installation maximum, CA-ACF2 suspends the user.

Once the user has successfully logged in, the user is granted Model 204 privileges of X'00'. Any additional privileges are determined at the time a Model 204 command requires a specific privilege.

CCASTAT considerations

To provide a smooth transition from Model 204 security to CA-ACF2 security, user ID data can continue to be stored in CCASTAT. As users become secured by CA-ACF2, delete the standard Model 204 CCASTAT user ID entries.

All the system manager commands concerning CCASTAT maintenance functions continue to work as usual. ZBLDTAB also continues to function as usual when creating the initial CCASTAT.

File, group, and subsystem security functions are defined and described in the *Model 204 File Manager's Guide* and the *Model 204 System Manager's Guide*.

User 0 login

When the CA-ACF2 MVS Interface is active, User 0 is validated as the owner of the Model 204 run, regardless of whether the run is Online or BATCH204.

Model 204 always attempts to log in User 0 automatically and verify that the user ID supplied matches the user ID of the owner of the address space. Do not supply a user ID or password on the login command for User 0; Model 204 determines the owner user ID and supplies it automatically.

AUTHCTL command

The AUTHCTL command is available to view or delete control information from releases of Model 204 before Version 2.1. If your site started using CA-ACF2 MVS after Version 2.1, there is no information for you to view. The AUTHCTL functions were replaced by the ACF2GEN macro.

The format of the AUTHCTL command is as follows:

AUTHCTL {*function*} ACF2

where:

function	Specifies one of the following functions:
	<ul style="list-style-type: none">• <i>D</i> deletes CA-ACF2 control information from the CCASTAT data set• <i>LIST</i> displays the security interface control options (from CCASTAT)• <i>VIEW</i> displays the interface control options currently in effect for the interface that is active (CCASTAT and ACF2GEN)

For example, if enter the following command:

AUTHCTL VIEW

A list similar to the following is displayed:

```
ACF2 INTERFACE OPTIONS
RESOURCE          JST          ACF2 RESOURCE TYPE
LOGIN             X'0172'    AUTHORITY BYTE OFFSET IN LIDREC
                  X'80'      BIT DEFINING MODEL204 AUTHORITY
ACCOUNT           X'0176'    ACCOUNT OFFSET IN LIDREC
                  X'04'      ACCOUNT LENGTH
RECSCTY           X'0000'    RECORD SECURITY KEY OFFSET IN LIDREC
                  X'00'      RECORD SECURITY KEY LENGTH
PRIORITY          LOW        PRIORITY DEFAULT OR OFFSET IN LIDREC
VALIDATE                                ACCOUNT VALIDATION OPTION IN EFFECT
DLMCHECK                                USE $JOB DLM CHECKING OPTION
                                      M204USER DEFAULT LOGONID IN USE
```

The last line displayed indicates the current default Logonid in use. For more information about the default Logonid, refer to “Inserting a CA-ACF2 default Logonid” on page 29.

The data originally stored by the AUTHCTL command in Release 9.0 or earlier versions of Model 204 is retrieved by Model 204 only during initialization. (This data can be displayed by issuing an AUTHCTL LIST ACF2 command.) For information stored after Release 9.0, see the ACF2GEN macro.

Note: If you do not delete the CA-ACF2 MVS Interface control record from CCASTAT, it is used during initiation, and the values in ACF2PARM override all values except the authorization bits.

Using trusted login for CRAM users

If your site uses CRAM, you can use the trusted login feature, which allows users to issue login commands or calls that do not include a user ID and password. For a user to log in as trusted, the user ID must be defined to CA-ACF2 MVS. User IDs defined to CCASTAT are not allowed to log in as trusted users. CICS users must be using the CA-ACF2 MVS interface for CICS. Only CICS user IDs that log in through CA-ACF2 can log in as trusted users.

Trusted login can be used with the following CRAM thread IODEV types:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

To set up trusted login for a user, set the SECTRLOG user parameter, as discussed on “Setting the SECTRLOG parameter” on page 30.

Login processing for trusted login

For users connecting with a CRAM thread that allows trusted login, the user login processing routines are changed to handle a LOGIN command or IFSTRT call without a user ID and password.

- The CRFS and CRIO channel threads handle User Language statements. These users ordinarily issue a LOGIN or LOGON request with the following format:

```
LOGIN userid [account];password [:new password];apsynname
```

For trusted logins, the format is:

```
LOGIN;;apsynname
```

- The IFAM channel threads handle IFAM2 statements. These users ordinarily issue an IFSTRT or IFSTRTN call with the following format:

```
IFSTRT (RETCODE,LANG_IND,LOGIN,THRD_TYP,THRD_ID)
```

```
IFSTRTN (RETCODE,LANG_IND,LOGIN,THRD_TYP,THRD_ID,CHAN)
```

The LOGIN parameter is required and supplies the user ID and password as a character string using the following format:

```
'userid [account];password [:new password];'
```

For trusted logins, the statement format is the same but the login character string is a semicolon surrounded by single quotes (';').

Trusted login errors

When using trusted login, you might receive one of the following errors:

- The following message is generated when either:
 - a. Model 204 job requesting the trusted login feature is running without a Model 204 Security Interface.
 - b. Model 204 address space does not have enough storage to allocate an internal work area for the trusted login feature. In this situation, the Model204 job does not initialize, because it still has to allocate storage for the Model 204 file buffers.

M204.2378: SECURITY TRUSTED LOGIN FEATURE DISABLED

- The following message is issued when the trusted user ID passed by CRAM is not between 1-8 bytes long:

M204.2379: INVALID TRUSTED USERID LENGTH = *length*.

Maintaining the CA-ACF2 MVS Interface

At each CA-ACF2 MVS site, the installation security officer (ISO) performs systemwide maintenance functions within CA-ACF2 MVS. This section describes the role of the security officer in maintaining the Model 204 CA-ACF2 MVS Interface.

Defining user privileges

The security officer must identify Model 204 privileges to CA-ACF2 MVS.

The Model 204 login privileges for CA-ACF2 MVS users are defined as Generalized Resource Rules. The RESOURC argument of ACF2GEN is used along with fixed CA-ACF2 key names. If an argument is not specified, RESOURC defaults to 204.

User privilege names

The following set of user privilege names define the possible privilege rules for a user:

User privilege name	Defines users...
'PRIV.SUPER.USER'	With Superuser privileges. A superuser can create a file with the CREATE command. This privilege name corresponds to the LOGCTL setting X'80'.
'PRIV.SYSTEM.ADMIN'	With System Administrator privileges. A system administrator can issue commands such as LOGWHO, MONITOR, PRIORITY, and WARN. This privilege name corresponds to the LOGCTL setting X'40'.
'PRIV.CHANGE.FILE.PASSWORD'	Who can change CCASTAT file passwords when opening a file. This privilege name corresponds to the LOGCTL setting X'20'.

User privilege name	Defines users...
'PRIV.SYSTEM.MANAGER'	Who can issue system manager commands. A system manager can issue all system administrator commands plus certain other privileged commands. This privilege name corresponds to the LOGCTL setting X'08'.
'PRIV.OVERRIDE.RECORD.SECURITY'	Who can override record security. This privilege name corresponds to the LOGCTL setting X'04'.

CCASTAT users

For CCASTAT users, privileges are defined to the CCASTAT user ID by the Model 204 LOGCTL command.

Note: The LOGCTL command contains an additional privilege bit (x'10') that indicates whether or not CCASTAT users can change their own login passwords at login time. This privilege bit is not required for the Interface, because you set this privilege for all CA-ACF2 MVS users within CA-ACF2.

Defining corresponding Generalized Resource Rules

Generalized Resource Rules that correspond to the defined privileges must be defined to CA-ACF2 MVS. The rules (and ACF command) to define the standard Model 204 privileges to CA-ACF2 are as follows:

```

ACF                                (Begins CA-ACF2 rule maintenance)
SET RESOURCE                       (Sets CA-ACF2 to resource mode)
*COMPILE

$KEY (PRIV.SUPER.USER)             TYPE (acf2type)
  UID (authorized user)            ALLOW
  UID (unauthorized user)          PREVENT
  .
  .
  .
$KEY (PRIV.SYSTEM.ADMIN)           TYPE (acf2type)
  UID (authorized user)            ALLOW
  .
  .
  .
$KEY (PRIV.CHANGE.FILE.PASSWORD)   TYPE (acf2type)
  UID (authorized user)            ALLOW
  .
  .

```

```

      .
$KEY (PRIV.SYSTEM.MANAGER) TYPE (acf2type)
      UID (authorized user) ALLOW
      .
      .
      .
$KEY (PRIV.OVERRIDE.RECORD.SECURITY) TYPE (acf2type)
      UID (authorized user) ALLOW
      .
      .
      .
STORE
END

```

where:

PRIV.privilege.type	Specifies one of the fixed privilege names (discussed on page 17) that Model 204 uses to build CA-ACF2 retrieval search keys for the rules
acf2type	Is the Generalized Resource Type field as defined to Model 204 by the ACF2GEN RESOURC argument

The CA-ACF2 UID entry referred to here either allows or prevents a specified user from accessing that named resource.

A privilege rule is tested whenever a user issues a command that requires a specific privilege. If the user is authorized to have that privilege, the command succeeds. If not, the user typically receives a Model 204 error message and the attempt is logged as a CA-ACF2 violation.

Setting up the rules and masking patterns is a detailed process and is not described fully here. Information regarding setting up CA-ACF2 UID strings and writing rules using masks referring to these strings is documented thoroughly in the site's CA-ACF2 MVS documentation.

Preparing the ACF2PARM parameter module with ACF2GEN

The ACF2PARM module allows you to define CA-ACF2 MVS arguments for your site. ACF2PARM consists of one or more ACF2GEN macros that are coded, assembled, and generated by your site.

The ACF2GEN macro generates a named set of arguments that govern login and other security processes. At run time one of these named sets is used by the CA-ACF2 MVS Interface. The ACF2PARM parameter module can be linked with Model 204 or dynamically loaded when the CA-ACF2 Interface is initialized.

SECPLIST parameter

The SECPLIST User 0 parameter in CCAIN allows you to specify the name of the ACF2GEN argument set to initialize the interface. The name is defined by the assembler label name of the ACF2GEN macro.

If the SECPLIST parameter is not in CCAIN, ACF2PARM is used as the default name of the argument set. If no match is found for the SECPLIST name or the ACF2PARM default name in the ACF2PARM module, the interface is initialized using the precoded default parameters of the CA-ACF2 Interface. In this case, Login, Account, and Record Security Key validation are not performed.

ACF2GEN macro

The JCL needed to assemble ACF2PARM is in the installation JCL library (as member ACF2GEN).

The format of the ACF2GEN macro is as follows:

```
TITLE 'GENERATE AN ACF2 PARAMETER MODULE'

NAME  ACF2GEN RESOURC=204,    ACF2 resource type           X
      LOGIN=(0,0),           Login ensure position and bit   X
      ACCOUNT=(0,0),         Account position and length      X
      RECSCTY=(0,0),         Record secty key position & length X
      VALIDAT=,              Account validation              X
      PRTY=,                 Priority position                 X
      LOGONID=M204USER,      Default Logonid                  X
      NODLMCHECK/DLMCHECK    Perform DLM check
ACF2GEN TYPE=END             End of macro definitions
END
```

where:

<i>Name</i>	Defines the name of this set of CA-ACF2 arguments. Because any number of argument sets can be in the ACF2PARM module, each set must be given a unique name. Use the default name of ACF2PARM for one of the argument sets in case a SECPLIST is not specified.
<i>RESOURC</i> argument	Specifies the 3-character CA-ACF2 Generalized Resource Type code for this argument set. The Generalized Resource Rules for Model 204 privileges are defined and grouped by this code and stored in CA-ACF2. The code must match the Generalized Resource Type code described to CA-ACF2 in the ACFFDR statements for your site. The default is 204.

<i>LOGIN</i> argument	Specifies the position and bit mask in the Logonid record that determines if a user is allowed to log in to Model 204. If you specify position, the CA-ACF2 MVS Interface determines if the appropriate bit is set before allowing access. If you do not specify position, all users have access to Model 204, provided that they pass the user ID/password and optional account validation.
-----------------------	---

The rules for specifying position and bit mask are:

- *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. To verify the position, look in the LIDREC DSECT of the ACFFDR listing.
- Specify position as a hexadecimal value (X'nnnn').
- *Bitmask* indicates which bit within the byte at the specified location is being tested. Specify bit mask as a hexadecimal byte (X'mm').
- If you specify LOGIN, the operands must be separated by a comma and enclosed within parentheses.
- *ACCOUNT* argument specifies the position and length within the Logonid record where the user's default account can be found. If there is a position defined, and the user does not enter an account while logging in, the account in the user's Logonid is used.

The rules for specifying position and length are as follows:

- *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. To verify the position, look in the LIDREC DSECT of the ACFFDR listing.

Specify position as a hexadecimal value (X'nnnn').

- *Length* is a hexadecimal value from 1 to 10.

If you specify ACCOUNT, the operands must be separated by a comma and enclosed within parentheses.

- The *RECSCTY* argument specifies the position and length within the Logonid record where the user's default record security key can be found. This value overrides the standard Model 204 record security key, which defaults to the user ID.

The rules for specifying position and length are as follows:

- *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. To verify the position, look in the LIDREC DSECT of the ACFFDR listing.

Specify position as a hexadecimal value (X'nnnn').

- *Length* is a hexadecimal value from 1 to 10.

If you specify RECSCTY, the operands must be separated by a comma and enclosed within parentheses.

- The *VALIDAT* argument has only one valid parameter, *ACCOUNT*, which specifies that any account entered by the user during the login process that does not match the LIDREC default account value is validated by CA-ACF2.

VALIDAT=ACCOUNT compares the account value entered by the user to the value in the user's Logonid. If the values match, the user is allowed in to Model 204. If the values do not match, the account entered by the user is validated against the standard CA-ACF2 Resource type TAC (refer to the *CA-ACF2 System Programmer's Guide* for more information). If this validation is successful, the user is allowed into Model 204. Otherwise, the login fails.

- The *PRTY* argument specifies the default user priority or a position within the CA-ACF2 Logonid where a priority character can be found. *PRTY* options are:
 - *Priority* specified by a one-character value:
 - H (high)
 - S (standard)
 - L (low)
 - N (none)

PRTY=priority indicates the default user priority for all users logged in under CA-ACF2.
 - *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. To verify the position, look in the LIDREC DSECT of the ACFFDR listing.

Specify position as a hexadecimal value (X'nnnn').

The default priority is Standard if the *PRTY* keyword is omitted or if an invalid character is found at the specified offset in the Logonid record. For more information about priorities, refer to the *Model 204 System Manager's Guide*.

- The *LOGONID* argument specifies the 1- to 8-character default Logonid described earlier in this section.

If this field is left blank, CCASTAT-defined users are *not* allowed to log in to Model 204. Only valid CA-ACF2 users can utilize the running system.

If this value is specified as 'JOBLID', the Logonid of the executing job is used as the default Logonid.
- The *DLMCHECK/NODLMCHECK* argument specifies DLM processing options for jobs submitted through the internal reader using the *USE \$JOB* command. The DLM parameter on a *DD ** or *DD DATA* statement allows users to submit jobs that can, in turn, submit other jobs. This can potentially compromise security. The *DLMCHECK* argument allows you to prevent additional parameters from being processed when coding the *DLM=* parameter.

DLM processing options are as follows:

- *DLMCHECK* requires that if a DLM= parameter is used in a JCL stream, it must be the only parameter supplied. In this case, only the following forms of these statements are correct:

```
//DDNAME DD *,DLM=' ; ; '
```

or

```
//DDNAME DD DATA,DLM='&&&&'
```

In the statements above, DLM follows the rules described in IBM JCL documentation: any other parameters supplied result in an error. If there is a job statement following the offending statement, Model 204 inserts *//*JOBFROM* after the JOB statement set and treats it like an independent job.

- *NODLMCHECK* checks only the validity of the DLM= parameter and does not look for the other parameters that can be specified on the JCL statement. All JCL statements after the DLM= parameter are sent to the internal reader without being checked.

NODLMCHECK does not guarantee that an error on the statement with the DLM= parameter is caught before submission. In case of an error, the JCL following the offending statement is submitted with the CA-ACF2 authority from the MUSASS.

The default is *DLMCHECK*.

Sample ACF2PARM module

The following ACF2PARM module contains two sets of ACF2 arguments. In the first set, the name is LOG1 and account security is in effect. If the user is logged on through CCASTAT, the default Logonid is M204USER. In the second set, the name is LOG2 and both account and login security are in effect. In addition, if the user is logged on through CCASTAT, the Logonid of the executing job is considered the default Logonid.

```
TITLE 'ACF2PARM MODULE'

LOG1  ACF2GEN RESOURC=204,          X
      LOGIN=(0,0),                  X
      ACCOUNT=(X'0141',X'0A'),      X
      RECSCTY=(0,0)                 X
      VALIDAT=ACCOUNT,              X
      PRTY=S,                        X
      LOGONID=M204USER,              X
      DLMCHECK

LOG2  ACF2GEN RESOURC=204,          X
      LOGIN=(X'012F',X'04'),        X
      ACCOUNT=(X'0132',X'0A'),      X
      RECSCTY=(0,0)                 X
```

VALIDAT=ACCOUNT,	X
PRTY=S,	X
LOGONID=JOBLID,	X
DLMCHECK	
ACF2GEN TYPE=END	
END	

Installing the Model 204 CA-ACF2 MVS Interface

Follow these steps to install the CA-ACF2 MVS Interface.

1. Decrypt CA-ACF2 MVS.
2. Assemble SBA2OS.
3. Assemble ACF2PARM and optionally link it as a separate load module.
4. Link-edit Model 204.
5. Define Model 204 as a CA-ACF2 MUSASS.
6. Add the @MUSASS macro to ACFFDR.
7. Define Logonid fields for Model 204.
8. Insert a CA-ACF2 default Logonid.
9. Define Model 204 user privileges.
10. Update CCAIN with the SECPLIST parameter.

Terminal security considerations

ACF2 can enforce terminal security when a user logs in. At that time, the source of the login is passed to CA-ACF2. For systems running Model 204 with the SNA Communications Server (formerly VTAM) interface, Model 204 recognizes the terminal name and stores it for use.

Model 204 configurations that use CRAM (TSFS, CICS, IFDIAL) must use the CRAM channel name instead of the terminal ID as the user source, because there is no secured mechanism in CRAM for identifying the source of the user. Because CRAM channel names can be identified as valid sources, the Logonids allowed to use those sources can be validated. Using the CRAM channel name temporarily ensures that entry to Model 204 is from an approved source. CRAM sites can also use the trusted login feature to allow users to log in to Model 204 without supplying a user ID. See “Using trusted login for CRAM users” on page 15 for more information.

Decrypting CA-ACF2

As part of INS204, the M204DECR job is generated with all the steps needed to decrypt CA-ACF2. For more information, refer to the *Model 204 Installation Guide for IBM z/OS*.

Assembling SBA2OS

All CA-ACF2 MVS Interface users must assemble SBA2OS.

Assemble SBA2OS with the CA-ACF2 macro library. To assemble SBA2OS, customize and run the supplied SBA2ASM JCL member from your Model 204 JCLLIB (PDS data set) and place the object code in the Model 204 object library (OBJLIB). Follow the instructions in the JCL comments to customize the job; it expects the source to come from the Model 204 macro library (MACLIB).

Assembling ACF2PARM

The JCL needed to assemble ACF2PARM is in the installation JCL library (as member ACF2GEN). Refer to “Preparing the ACF2PARM parameter module with ACF2GEN” on page 19 for more information on ACF2GEN.

The result of the assembly of the ACF2GEN macro can be link-edited to an authorized library for retrieval during Model 204 initialization, or provided at the time of the Model 204 link-edit.

Optionally linking ACF2PARM as a separate load module

If you link ACF2PARM as a separate load module, use the SECRLINK job in the JCL library. Modify the job according to the comments.

If you link ACF2PARM with the Model 204 configuration, add the following line in SYSLIN for the link-edit steps for Online, BATCH204, IFAM1, and IFAM4:

```
INCLUDE OBJLIB (ACF2PARM)
```

Link-editing Model 204

To link Model 204, make the following changes to Model 204 link-edit JCL and control statements (for the link-edit of the Online, BATCH204, IFAM1, and IFAM4 load modules) in order to include the interface modules supplied by Computer Associates and used by the CA-ACF2 Interface. The actual data set names used for the CA-ACF2 modules depend on your site. An example of the JCL statement is:

```
//ACFLIB DD DSN=SYS1.ACFAMOD,DISP=SHR
```

The corresponding link-edit statements to add to the current Model 204 link-edit input are:

```
INCLUDE ACFLIB ($ACFGCVT)
INCLUDE ACFLIB (ACF$FGCB)
INCLUDE OBJLIB (SBA2OS)
INCLUDE OBJLIB (ACF2)
```

Customize the JCL according to the comments in the job before running it. When the job successfully completes, the new load modules must be copied back into the Model 204 load library, which replaces the old load modules.

In order for Model 204 to execute as a MUSASS, Model 204 must be linked into an APF-authorized library. This means that any Online configuration, or batch jobs that perform multiple logins for different user IDs, must be linked to an authorized library.

Note: All concatenated members of a STEPLIB must be APF-authorized for the Model 204 LOADLIB to stay APF-authorized.

BATCH204 or IFAM can be linked to a nonauthorized library, because the user who logs in is the same user who starts the job, or the user logs in with a user ID that is on CCASTAT. Any other CA-ACF2 Logonid fails.

Defining Model 204 as a CA-ACF2 MUSASS

The Model 204 Online is a MUSASS (Multiuser Single Address Space System) to CA-ACF2. A MUSASS provides services to many users through its own processing; however, CA-ACF2 is aware only of the main task in the address space (in this instance, Model 204).

To define an Online MUSASS to CA-ACF2, CA-ACF2 must be modified through the @MUSASS macro supplied by Computer Associates in order to identify one or more Model 204 Online environments as a MUSASS. This process defines the named Online environment as having the appropriate CA-ACF2 privileges to perform services on behalf of Model 204 users.

This definition process is necessary only for the job that starts a Model 204 Online execution with many users. It need not be performed for IFAM1/IFAM4 or for any copies of Model 204 that run as BATCH204, because these do not issue protected MUSASS service calls when running only one user Logonid. Defining a formal MUSASS to CA-ACF2 provides better control of the address space.

When defining a MUSASS, you can ensure that an Online MUSASS is not canceled in case of a security violation by specifying the NON-CNCL attribute.

Use the NON-CNCL attribute rather than an z/OS non cancellable PPT entry to achieve this, because a security violation can cause non cancelable loops in Model 204. It is valid to have Model 204 marked as non cancellable to z/OS, as long as the CA-ACF2 Logonid describing the MUSASS also has the NON-CNCL attribute.

The steps in the definition process are:

1. Select a name and insert a Logonid for the MUSASS.
2. Define an @MUSASS macro in the ACFFDR.

The following example illustrates how to create a MUSASS Logonid through the ACF command in either TSO or with a batch job:

```
INSERT      MODEL204 NAME (MODEL 204 MUSASS PROGRAM) -  
            MUSASS -  
            NO-SMC -  
            SECURITY ACCOUNT NON-CNCL JOBFROM
```

where:

MODEL204	Is the name of the job or started task running as a MUSASS.
----------	---

Note: The name following the INSERT (here, MODEL204), must be the same as the name in the @MUSASS entry.

Adding the @MUSASS macro to ACFFDR

The following modification must be made to the standard CA-supplied macro definitions. The ACFFDR must be reassembled as described in the CA-ACF2 documentation.

The @MUSASS entry that must be made in the ACF2 Field Definition Record (ACFFDR) follows:

```
@MUSASS MODEL204,MLID=ACF2,FASTPTH=YES,CACHE=NO
```

where:

MODEL204	Is the logonid of the submitted Online system or started task that executes Model 204.
<i>Other options</i>	Are installation-definable options to control the storage and processing of rules. No specific options are required for the operation of the CA-ACF2 Interface.

Defining Logonid fields for Model 204

In CA-ACF2, the Logonid record contains all information for a defined user of a computer system. Depending on the parameters specified in one or more ACF2GEN macros, fields might need to be added to the Logonid records. This requires the following steps:

1. Add the assembler field definitions to the USERLID extension of the CA-ACF2 LIDREC
2. Add @CFDE entries to the ACFFDR

The optional CA-ACF2 MVS Interface features that might require fields in the Logonid record are:

- Login validation (see the ACF2GEN LOGIN parameter)
- Account validation (see the ACF2GEN ACCOUNT parameter)
- Individual user priorities (see the ACF2GEN PRTY=position parameter)

The following sample illustrates how to define a field in CA-ACF2 that is tested for user authority to log in to various Online copies of Model 204.

Assume the following field definition exists in the USERLID extension of CA-ACF2 LIDREC:

M204AUTH	DS	X	<i>M204 authority to use privilege</i>
M204TEST	EQU	X'01'	<i>may use M204TEST online</i>
M204PROD	EQU	X'02'	<i>may use M204PROD online</i>

The following @CFDE entries would be made in the ACFFDR to define the CA-ACF2 maintenance requirements. These attributes could then be assigned specific CA-ACF2 Logonids.

@CFDE M204TEST,M204AUTH,BIT,	<i>may use TEST M204</i>
BITMAP=M204TEST,	<i>use this bit</i>
GROUP=2,ALTER=SECURITY,	<i>secty group and admin auth</i>
LIST=ALL,	<i>who can list data</i>
FLAGS=NULL+RESTRICT,PRTN=3,RRTN=3	<i>field default/ list routines</i>
@CFDE M204PROD,M204AUTH,BIT,	<i>may use production M204</i>
BITMAP=M204PROD,	<i>use this bit</i>
GROUP=2,ALTER=SECURITY,	<i>secty group and admin auth</i>
LIST=ALL,	<i>who can list data</i>
FLAGS=NULL+RESTRICT,PRTN=3,RRTN=3	<i>field default/ list routines</i>

These examples illustrate the relationship between the LIDREC field and usage of that field on the @CFDE definition. The GROUP, ALTER, LIST, and FLAGS keywords are variable depending on local requirements.

When these modifications are incorporated into an ACFFDR and reassembled and relinked, each user Logonid can be given the authority to use the attribute of M204PROD and/or M204TEST.

If the M204AUTH field is at Logonid location x'22F', the LOGIN parameter in ACF2GEN for the test MODEL204 address space indicates:

```
ACF2GEN ....other entries...,LOGIN=(X'22F',X'01')
```

This also illustrates how other fields are defined for use by Model 204, with the exception of different field lengths and the CA-ACF2 specification necessary to work with these fields.

Inserting a CA-ACF2 default Logonid

When Model 204 is executing as a MUSASS, the CA-ACF2 MVS Interface might require a default Logonid. This default Logonid limits the authorization of users who are still defined in CCASTAT. Each user not logged directly into CA-ACF2 is assigned the authorization of the default Logonid. This Logonid defaults to M204USER, unless it has been changed by the installation via the ACF2PARM module described page 19.

The default user's Logonid must be inserted in CA-ACF2 for CCASTAT-defined users to log in. Because the login occurs without a password, the default Logonid definition must be assigned the RESTRICT attribute (refer to the CA-ACF2 documentation for a complete description of this attribute).

If RESTRICT is not specified, the login for the default user fails. If the login for the default user fails, users cannot log in to Model 204 unless they are CA-ACF2 users. Additionally, the default Logonid must have job submit authority for Model 204 CCASTAT users to submit jobs.

If the LOGIN argument of the ACF2GEN macro is specified, the default user's authority to log in to Model 204 is checked for authorization. If the default Logonid does not have authority to log in to this Model 204 job, then no CCASTAT-defined user has authority to log in to Model 204.

For security purposes a site can set up rules restricting who can modify and relink a new ACF2PARM module. Therefore, a system manager cannot alter the default Logonid. If a different default Logonid is required, the security officer or systems programmer must modify and link a new ACF2PARM parameter module, or allow access to a different parameter set in the ACF2PARM module.

If the LOGON ID parameter is blank, all user logins from CCASTAT are disallowed. This mode of operation forces all users to log in to CA-ACF2 to gain access to Model 204.

Note: Default Logonid privileges are in effect for the duration of a run. Changes to the privileges become effective after the Online is recycled.

Defining Model 204 user privileges

In either a MUSASS or single-user environment, Model 204 queries CA-ACF2 for the authority to grant access to various system resources on behalf of the user. User privileges are defined as resources to be protected, and rules are written to determine who is authorized for use of that resource. See "Defining corresponding Generalized Resource Rules" on page 18 for more information about Generalized Resource Rules.

ACF2 Interface storage requirements

When operating as a CA-ACF2 MUSASS, CA-ACF2 is responsible for building several control blocks for each user. Because the storage allocation comes

from the address space in which the MUSASS is running, it is necessary to increase the value of the Model 204 SPCORE parameter for Online executions and all MUSASS controlled Model 204 jobs.

Converting users from CCASTAT security to CA-ACF2 security

When the CA-ACF2 MVS Interface is running, user IDs can be moved from CCASTAT to CA-ACF2 security. Once the user IDs have been transferred to CA-ACF2, they can be deleted from CCASTAT.

Setting the SECTRLOG parameter

The SECTRLOG user parameter defines which CRAM thread applications are allowed to log in to Model 204 with a trusted user ID. SECTRLOG must be set only if your site is using the trusted login feature described on page 15. The CRAM threads for which trusted login applications are allowed are:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

SECTRLOG must be set on the first IODEV line for each trusted CRAM thread, because it is picked up during the initialization of each CRAM channel. The following settings are valid:

Setting	Meaning
X'00'	Trusted Login <i>not</i> allowed (default)
X'01'	CICS applications (CRFS, CRIO, IFAM)
X'02'	TSO applications (CRIO, CRFS)
X'04'	Batch applications (CRIO, IFAM)

Conversion tasks and considerations

The following checklist identifies the general tasks that must be completed to install and activate the CA-ACF2 Interface:

Table 2-1. Conversion checklist for CA-ACF2 MVS

Step	Task
1.	Define the Generalized Resource Type (204 or the value that will be specified in the ACF2GEN RESOURC argument). Change the ACFFDR system definition by adding: <ul style="list-style-type: none">• Definition of the Model 204 authority bit, if the LOGIN argument is specified in the ACF2GEN macro• Definition of the default account field, if the ACCOUNT argument is specified in the ACF2GEN macro• Definition of the PRIORITY byte, if individual priorities are assigned to each other• @MUSASS macro to define Model 204 as a MUSASS
2.	If an ACF2PARM module is to be linked with Model 204, execute member ACF2GEN in the installation JCL library. If ACF2PARM will be loaded dynamically, also execute SECLINK.
3.	Unload the installation software (if necessary) and assemble SBA2OS. Link Model 204 with the modules ACF2, \$ACFGCVT, ACF\$FGCB, SBA2OS, and ACF2PARM.
4.	Insert the MUSASS Logonid that defines Model 204 to CA-ACF2. Add the default Logonid with the RESTRICT attribute.
5.	Prepare batch JCL or a started task cataloged procedure to execute the defined Model 204 MUSASS name.
6.	Prepare user privilege rule sets to correspond with fixed privilege names in Model 204. Specify which users are allowed or prevented from using those privileges.
7.	Provide valid CA-ACF2 users the privilege to use Model 204.
8.	Change User 0 input in CCAIN to include the SECPLIST= <i>parametername</i> parameter.
9.	Delete Model 204 users from CCASTAT, forcing logins through CA-ACF2.

3

CA-ACF2 VM Interface

In this chapter

- Overview
- CA-ACF2 VM Interface configurations
- Brief introduction to CA-ACF2 VM
- Running the CA-ACF2 VM Interface with Model 204
- Using the CA-ACF2 VM Interface
- Login processing
- Maintaining the CA-ACF2 VM Interface
- Preparing ACF2PARM with ACF2GEN
- Installing the CA-ACF2 VM Interface
- Modifying the VM directory
- Creating the PROFILE EXECs
- Decrypting object modules
- Creating Model 204 modules and saved segments
- Creating ACF2PARM
- Inserting the SECUR204 CA-ACF2 Logonid
- Writing rules
- Updating Model 204 EXECs

- Updating Model 204 CCAIN files

Overview

This chapter presents an overview of the Model 204 CA-ACF2 VM Interface and how it affects the Model 204 user, system manager, and CA-ACF2 installation security officer. Installation requirements and a conversion task summary also are presented.

CA-ACF2 VM

CA-ACF2 VM is an extension of the IBM VM/CMS environment that provides comprehensive data security functions. CA-ACF2 VM protects all data (or installation-defined resources) by default; that is, it protects all resources from all users unless specifically allowed by the resource owner or a security officer. This differs from other security systems that force you to define what is to be protected, rather than what is to be shared.

Model 204 CA-ACF2 VM interface

The Model 204 CA-ACF2 VM Interface provides for an orderly migration from Model 204 security to CA-ACF2 security and allows you to use either or both security methods. The Interface provides the tools and instructions necessary to:

- Define a security service machine to issue all CA-ACF2 validation requests or to require the MODEL204 service machine to issue CA-ACF2 requests online
- Make the changes required to the ACFFDR and reload the ACFFDR
- Define the CA-ACF2 Logonid privileges required by the security service machine or the Model 204 service machine or both
- Specify CA-ACF2 VM Interface parameters in the ACF2GEN macro and create ACF2PARM
- Define CA-ACF2 Generalized Resource Rules to control access to ACF2PARM parameter sets defined by one or more ACF2GEN macros
- Define CA-ACF2 Generalized Resource Rules so CA-ACF2 can authorize Model 204 user privileges
- Define a CA-ACF2 Generalized Resource Rule to control system entry to Model 204
- Define an installation default user priority class, or a priority class field within a CA-ACF2 Logonid record
- Define a default user account within a CA-ACF2 Logonid record, and provide account authorization services

- Define a record security key field within the CA-ACF2 Logonid record; define source entry rules to enforce terminal security
- Migrate CCASTAT users to CA-ACF2 or force all users to log in through CA-ACF2
- Automatically log in CCASTAT and CA-ACF2 users without requiring a password or require all users to enter passwords at all times (optional)

In summary, the CA-ACF2 VM Interface enables you to implement CA-ACF2 LOGIN and LOGOUT security, to validate account, Model 204 user, and Model 204 execution privileges, and to set user priorities and record security keys.

CA-ACF2 VM Interface configurations

Two configurations are allowed for the CA-ACF2 VM Interface:

Inline validation	Validates all CA-ACF2 information via a security service machine
Service machine validation	Performs validation inline within the MODEL204 machine

These configurations are interchangeable; either or both can be used at the same site simultaneously in different virtual machines.

Inline validation

CA-ACF2 requests are synchronous. If CA-ACF2 requests are made inline from the Model 204 machine, whenever any user requires CA-ACF2 validation, everyone on that machine waits for the validation to finish.

In a batch or single-user environment, inline validation does not affect performance. But in a multiuser Online, a CA-ACF2 request by one user forces all other users to wait until the request has been processed.

Service machine validation

If CA-ACF2 requests are made through a security service machine, the security service machine issues the CA-ACF2 requests and sends the results back to Model 204. Using a security service machine is usually faster; only the user who requires CA-ACF2 validation is kept waiting.

Another advantage to using a security service machine, besides speed in a multiuser environment, is that only the security service machine needs special CA-ACF2 privileges. That is, if multiple Onlines execute in more than one machine rather than each machine having its own set of CA-ACF2 VM SRF and ACCOUNT privileges, all CA-ACF2 special privileges can reside on one virtual machine.

One possible configuration is to have multiuser Onlines off-load CA-ACF2 requests to the security service machine while batch, single-user, and IFAM1 users issue CA-ACF2 requests directly from the Online. This prevents multiuser users from having to wait while giving single users the fastest possible validation.

Figure 3-1 illustrates the CA-ACF2 VM Interface using the security service machine.

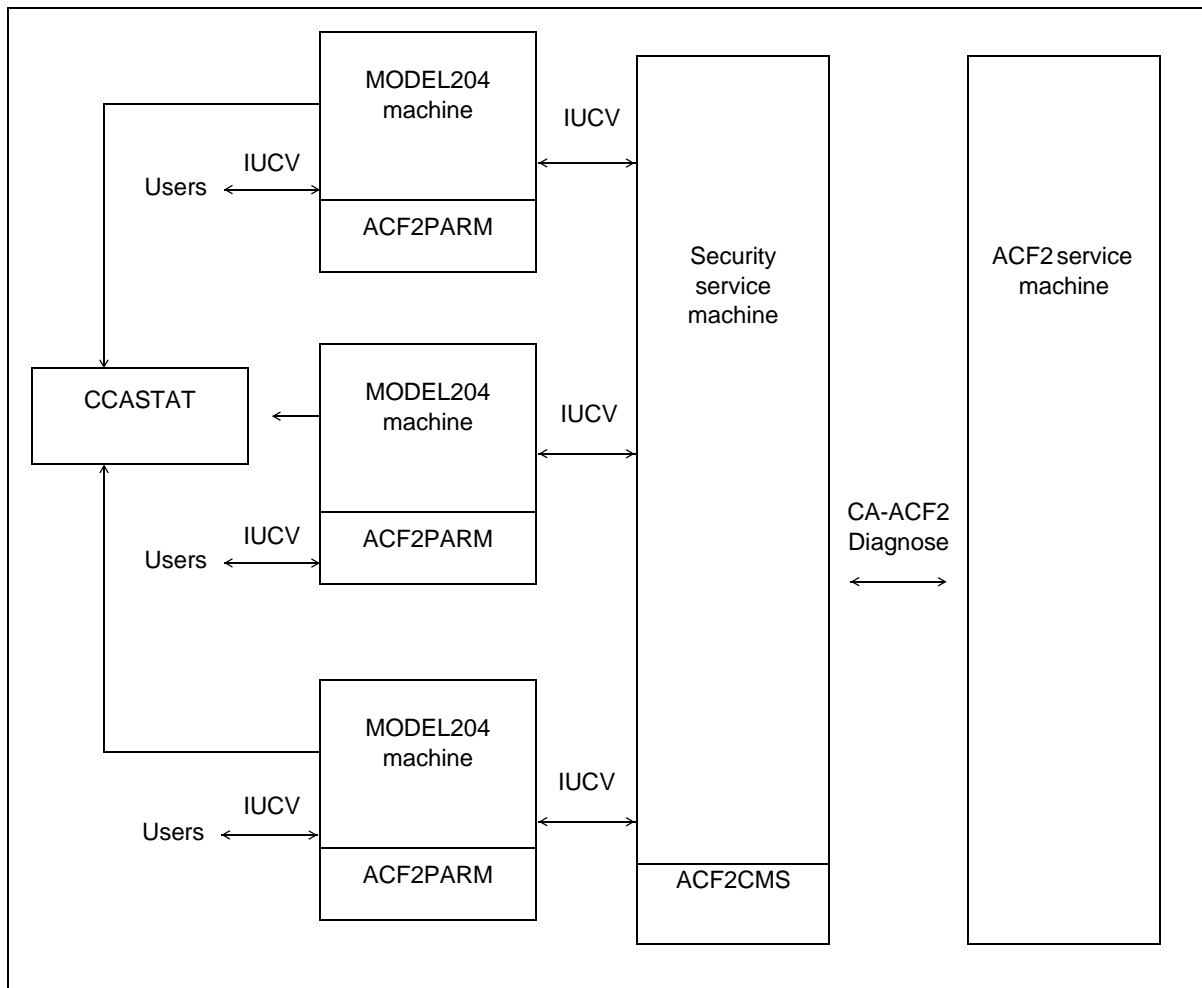


Figure 3-1. CA-ACF2 AM Interface using a security service machine

The CA-ACF2 VM Interface using the security service machine consists of the following components:

Model 204	Communicates with the security service machine via IUCV
ACF2PARM	Is loaded dynamically at initialization by Model 204
ACF2CMS text deck	Contains the CA-ACF2 VM Interface object code and invokes the System Request Facility (SRF) described on page 38.

Figure 3-2 illustrates the CA-ACF2 VM Interface without a security service machine.

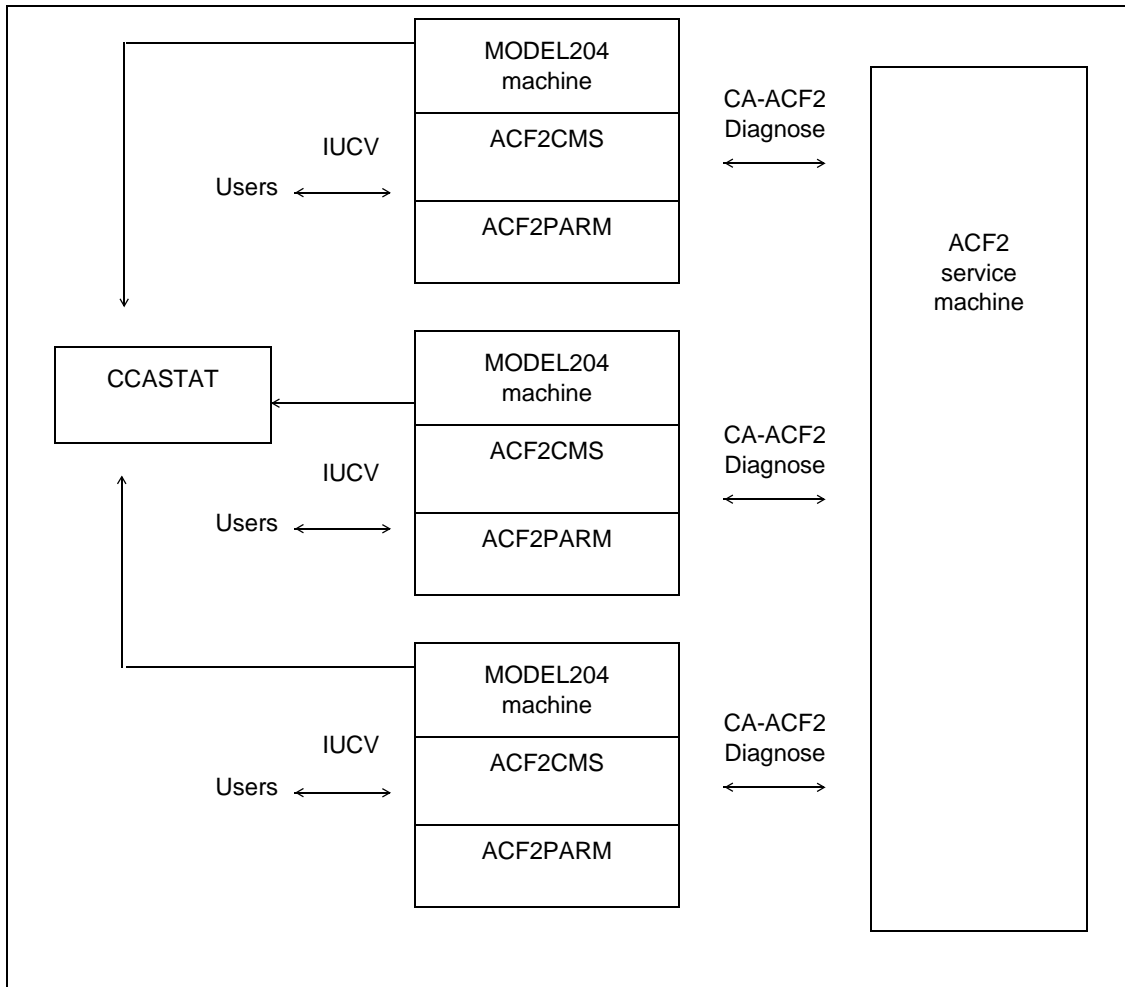


Figure 3-2. The CA-ACF2 VM Interface without a security service machine

Figure 3-2 illustrates the CA-ACF2 VM Interface without a security service machine, consisting of:

Model 204	
ACF2PARM	Is loaded dynamically at initialization
ACF2CMS	Contains the CA-ACF2 VM Interface object code and invokes the SRF facility
CCASTAT	Is a file defined to Model 204 machines via a FILEDEF

Brief introduction to CA-ACF2 VM

This section provides a brief summary of key CA-ACF2 VM features that are used in the discussion of the Model 204 CA-ACF2 VM Interface.

For more information about CA-ACF2 VM, see the documentation associated with that product.

CA-ACF2 processing

When a user logs in to VM, CA-ACF2 performs the following functions:

- Validates the password found for the Logonid in the CA-ACF2 Logonid database
- Checks for password expiration
- Optionally, ensures that the user is accessing the system from a specific terminal or other source
- Optionally, ensures that the user is accessing the system only during certain dates and times

System manager options

The system manager has the following additional options when a user logs in to Model 204 as a CA-ACF2 user:

- Require each user's login to go through Login or Account validation, defined by the ACF2GEN macro
- Either allow automatic login if the user ID is the same as the user's CMS Logonid or always require passwords
- Either allow user IDs from CCASTAT or require that all user IDs are CA-ACF2 Logonids
- Set the priority globally for all users or individually through a user's Logonid record
- If the priority is not set for each Logonid, Model 204 assumes global priority
- Change the user's record security key and not default to the Model 204 user ID

CA-ACF2 VM and the Model 204 CA-ACF2 VM Interface

The Model 204 CA-ACF2 VM Interface uses the System Request Facility (SRF) to send validation requests to the CA-ACF2 service machine. The two forms of SRF are single-user and multiuser.

Single-user SRF executes all CA-ACF2 validation requests using the Logonid of the CMS virtual machine running Model 204. The Model 204 configurations of single-user SRF are:

- Single-user Online or BATCH
- IFAM1

Multiuser SRF allows one CMS virtual machine to issue CA-ACF2 validation requests on behalf of other CMS virtual machines or Logonids. The terms multiuser SRF and MUSASS (Multiuser Single Address Space System) are interchangeable. Sample Model 204 configurations are:

- Multiuser Online
- Single-user Online

Logonid record

CA-ACF2 maintains a Logonid record (or LIDREC) for each user. The Logonid record includes the Logonid, which is the same as the CMS user ID. Each user on a CA-ACF2-protected system has a password associated with the Logonid. The passwords are stored in an encrypted format that cannot be reversed.

With CA-ACF2 VM, each user is defined once, regardless of the number of subsystems they can access. To help an installation define differing authorities for different subsystems, users are identified via a user identification (UID) field.

User identification string

The user identification string (UID) is a field on the Logonid record. It is a 24-byte character string constructed from the user's Logonid and other site-definable fields such as department number or employee ID. The default UID is the Logonid. CA-ACF2 compares the user's UID with the UID masks specified in a rule set to determine whether a user can have access to a resource.

A UID mask can be placed on any rule to allow or prevent access by a group of users based on their UID string. For example, a rule in the Rules database can have a UID mask that allows everyone with a department code of DEV to have access to a certain resource.

The ACMCB control block, which CA-ACF2 uses to perform validations, is created from information in the Logonid record. When using single-user SRF, this control block is in CP storage and cannot be inspected or changed by the users. When using a MUSASS (Multiuser Single Address Space Systems), a copy of the ACMCB for each user is made available to allow the Model 204 Online or the security service machine to make security validation requests on behalf of other users.

CA-ACF2 Generalized Resource Control

CA-ACF2's Generalized Resource Rules are similar to access control rules. However, they control access to resources other than files. These system resources can be defined either by CA-ACF2 or locally by each site.

By allowing each site to specify which local resources require security protection, the rules for a data processing installation can be complex enough to define any arbitrary or real system resources needed. One example of user-defined local resources is Model 204 user privileges.

Generalized Resource Rules are most often written by the CA-ACF2 security officer. For more information, see "Maintaining the CA-ACF2 VM Interface" on page 47.

Running the CA-ACF2 VM Interface with Model 204

A Model 204 multiuser Online always runs as a CA-ACF2 MUSASS (Multiuser Single Address Space System). A MUSASS is defined by CA-ACF2 as any program that runs as a single job in one virtual machine that supports more than one user. Other Model 204 configurations such as single-user and IFAM1 do not normally run as a MUSASS, because only a single user's authorization is in question.

In either a MUSASS or single-user environment, Model 204 queries CA-ACF2 for the authorization to grant access to various system resources on behalf of the user. For example, user privileges are defined as resources to be protected, and rules are written to determine who is authorized to use those resources.

Managing the authorization process

In a MUSASS, Model 204 manages the authorization process, because only Model 204 can identify which user initiates a request for a resource. Model 204 assists CA-ACF2 in validating those requests and responds to users who issue requests for unavailable resources.

Model 204 interfaces with CA-ACF2 in the following areas:

- User logins
 - LOGIN/LOGON command
 - IFSTRT function
 - IFLOG function
- User logouts
 - LOGOUT/LOGOFF command
 - IFFNSH/IFDTHRD function
- Model 204 user resources
 - Job submission (USE \$JOB command)

- Dynamic allocation of new DASD space (ALLOCATE command)
- All sequential file and VSAM file opens for READ/WRITE access
- Sequential data set handling (Record I/O)
- DUMP/RESTORE commands
- OPEN command naming a deferred update data set

Note: Because Model 204 databases are not validated on behalf of the user by the security interface, CA-ACF2 VM must grant the owner of an address space permission to open Model 204 file data sets for update, regardless of whether the owner has write or read-only privileges. This allows Model 204 to write back updates to the FPL (File Parameter List) page, as required by the database management system, regardless of the Model 204 file open privileges.

The CA-ACF2 VM Interface uses the ACFDIAG macro, discussed below, to validate user requests for Model 204 resources. No source code modifications or exits to the CA-ACF2 software are necessary. CA-ACF2 system parameters do, however, need modifying as described later in this chapter.

ACFDIAG macro

Whenever the CA-ACF2 VM Interface encounters a request for validation, the Interface issues a call to ACFDIAG, which in turn issues a diagnosis. The virtual machine then waits while the ACF2VM virtual machine processes the request and returns the results to the MODEL204 machine. The ACFDIAG macro is provided by SRF of CA-ACF2 VM.

ACF2CMS module

ACF2CMS, a Model 204 object module, builds all SRF argument blocks, issues the ACFDIAG macro, and retrieves data returned from ACF2VM (CA-ACF2's service machine). The ACF2CMS module is linked into the Model 204 Online, IFAM1, and the CA-ACF2 security service machine.

Where ACF2CMS issues its CA-ACF2 calls from is determined by the SECURID argument in the ACF2GEN macro as discussed in "ACF2GEN macro" on page 53.

- If the SECURID argument is defined as the CMS Logonid of the security service machine, ACF2CMS requests are made from the security service machine.
- If the CA-ACF2 SECURID argument is left blank, CA-ACF2 requests are made from the Online (or IFAM1).

Using the CA-ACF2 VM Interface

This section discusses the changes to Model 204 when running with the CA-ACF2 VM Interface. The areas affected include:

- LOGON or LOGIN
- IFSTRT function
- IFLOG function within IFAM1

LOGIN or LOGON command

To log in, enter:

LOGIN *userid* [*account*]

or

LOGON *userid* [*account*]

where

userid	Is a character string that identifies you
account	Is an optional character string that identifies the account under which you log in

Every Model 204 user has a user ID, password, and user privileges assigned by the system manager consisting of:

- User ID that identifies you to Model 204
- Password that provides access to the system
- User privileges associated with your user ID and password to define the particular type of access that you have
- Default priority class assigned to your user ID

When native Model 204 security is in effect, this user ID information is stored in a record in the CCASTAT system file. The record is deleted from CCASTAT when the system manager puts the user under CA-ACF2 security.

After the record is deleted from CCASTAT, your user ID and password are verified by CA-ACF2. Your user ID must be 8 characters or less or CA-ACF2 cannot validate it. Model 204 prompts for a password unless the NOPASSWORD argument of ACF2GEN is specified and the user ID is the same as your CMS Logonid. You can change the CA-ACF2 Logonid password when you log in to Model 204 (refer to the *Rocket Model 204 Parameter and Command Reference Manual*).

If the LOGIN option is specified on the M204 EXEC, a LOGIN command is automatically generated using your CMS Logonid as your Model 204 user ID. If the NOPASSWORD option of ACF2GEN is also specified, you are logged in and not prompted for a password when you issue M204 EXEC.

Account validation

Account validation can occur as part of account processing at your site. The two parts to account processing are:

- Provide a default account value when you do not specify an account on the LOGON/LOGIN command. In this case, the account is supplied from the CA-ACF2 user Logonid record.
- Provide account validation if you do specify an account value on the LOGON/LOGIN command.

Account validation ensures that your Logonid belongs to a certain group, or account, such as DEV or MKTG. If your site performs CA-ACF2 account validation, any value you enter in the account field is verified via CA-ACF2 services.

Source validation

Source entry validation is when CA-ACF2 ensures that you log in only from certain specified sources, such as a specific terminal. If your site performs source entry validation, the physical terminal source is your CMS Logonid. CA-ACF2 validates the source of the login along with the password.

IFSTRT function

In IFAM2 environments, the IFSTRT function allocates a Host Language Interface thread in a Host Language program. IFSTRT also establishes the calling convention.

When processing the login argument of the IFSTRT call, the user login process is the same process as for User 0 login (see page 46).

IFLOG function within IFAM1

The IFLOG function is used only in the IFAM1 environment. It is called following IFSTRT to identify the user when a login is required. This function is necessary in any IFAM1 program where user authorization is validated by CA-ACF2.

As with the IFSTRT function, the user login process is the same as the process for User 0 login (see page 46).

Login processing

The system manager is responsible for defining and controlling security processing in a Model 204 installation in which CA-ACF2 supplies authorization services. This section describes the login processing performed by the CA-ACF2 VM Interface.

Online login processing

When a user login occurs, Model 204 acquires the user ID and account and searches CCASTAT to determine whether or not the Model 204 user ID exists. If the user ID is found, Model 204 proceeds with CCASTAT security processing.

If the user ID is not in CCASTAT and CA-ACF2 security is in effect, Model 204 uses CA-ACF2 facilities to authorize the user login. Model 204 prompts for a password for CCASTAT and CA-ACF2 users unless the NOPASSWORD argument is specified in ACF2GEN and the user ID in the LOGIN command is the same as the CMS Logonid.

Note: The PASSWORD/NOPASSWORD argument of ACF2GEN overrides the LOGCTL NP CMS and P CMS options for CCASTAT and CA-ACF2 users.

IFAM2 login processing

If the login is processed via IFAM2, a password is always required, as the NOPASSWORD argument of ACF2GEN and the LOGCTL NP CMS option do not work in conjunction with IFAM2.

Source entry validation is performed if there is a SOURCE field on the Logonid record. If your site performs source entry validation, the physical terminal source is the user's CMS Logonid.

CA-AFC2 Logonid record

After the user's CA-ACF2 user ID and password are verified, the CA-ACF2 VM Interface has access to the CA-ACF2 Logonid record. Model 204 continues processing the login via the Logonid record and the following parameters are specified in ACF2GEN:

Parameter	Function
VALIDAT=LOGIN	Performs login validation to determine if a user is authorized to log in to Model 204. Model 204 does a Generalized Resource Call using the resource type defined by the RESOURC field of ACF2GEN and the PRIV.LOGIN.M204 key name.

Parameter	Function
ACCOUNT=	Performs account processing. If default account location and length are specified in ACF2GEN and the user does not enter an account, the default account is retrieved from the Logonid record and the user is allowed in to Model 204. Account processing is taken further if the VALIDAT=ACCOUNT and ACCTRES arguments are specified in ACF2GEN. If the user provides an account and there is a value in ACCOUNT= in ACF2GEN, the value entered is compared to the value in the Logonid record. If the values match, the user is allowed in to Model 204. If the values do not match, CA-ACF2 validates the account using the resource code defined by the ACCTRES argument of the ACF2GEN macro and the account value as the key name.
PRTY=	Specifies the user priority. The priority field is taken from the Logonid record if PRTY is specified as an offset into the Logonid record in ACF2GEN. Otherwise, if PRTY is set to H, S, or L, all CA-ACF2 users receive the same priority.
RECSCTY=	Specifies record security. The record security key is also taken from the Logonid record if RECSCTY is specified as an offset into the Logonid record in ACF2GEN. If it is not specified, the Model 204 user ID is used.

If CA-ACF2 denies system access because of an invalid password, account, or source, or if the user is not authorized to use Model 204, the user is not logged in and Model 204 issues an error message.

In addition, if the user enters an invalid password, CA-ACF2 increments a password violation counter, which is maintained by CA-ACF2. When the counter reaches a maximum number that you set, CA-ACF2 prevents entry to the system.

After successfully logging in, the user is granted Model 204 privileges of X'008'. Any additional privileges are determined whenever a user issues a command that requires a specific privilege.

CCASTAT and CA-ACF2

To provide a smooth transition from Model 204 security to CA-ACF2 security, user ID data can continue to be stored in CCASTAT. As users are secured by CA-ACF2, you can delete the standard Model 204 CCASTAT user ID entries.

If the LOGONID argument of ACF2GEN macro is left blank, CCASTAT users are not allowed to log in to Model 204. Only CA-ACF2 users are allowed access unless the LOGONID argument specifies VMID or JOBLID.

All the system manager commands concerning CCASTAT maintenance functions continue to work as usual. ZBLDTAB also continues to function as usual when creating the initial CCASTAT.

File, group, and subsystem security functions are defined and described in the *Model 204 File Manager's Guide* and the *Model 204 System Manager's Guide*.

User 0 login

User 0 is treated differently than ordinary users logging in to an Online through an IODEV.

There are several ways to log in User 0. The first is the method that Technical Support recommends:

- Code a LOGIN command with no user ID or password (recommended approach).

For example:

```
PGSIZE=6184
. . .
IODEV=41, POLLNO=10
LOGIN
BROADCAST . . .
. . .
```

Model 204 logs in User 0 as a CA-ACF2 user by appending the Logonid of the MODEL204 machine to the LOGIN command as the user ID. No password is required.

- Use a CCASTAT user ID and password:

```
LOGIN userid [account]
password
```

- Enter the Logonid of the MODEL204 machine on the LOGIN command without a password

The user ID must equal the Logonid or VMID of the virtual machine or the login fails.

User 0 processing

Model 204 first checks CCASTAT for the user ID in the LOGIN command:

- If the user ID is found, Model 204 checks the LOGONID argument of ACF2GEN to determine if a CCASTAT User 0 is allowed access.
- If the LOGONID argument is left blank, the CCASTAT user ID is not allowed and the User 0 login fails. Only a User 0 logged in through the CA-ACF2 VM Interface has access to Model 204.
- If the LOGONID argument is set to the VMID or JOBLID, a password is required and the CCASTAT User 0 receives its Model 204 privileges from the user ID entry in CCASTAT.

- If the user ID is the same as the Logonid, do not add a password. If there is a password, it is registered as an invalid Model 204 command.
- If CCASTAT does not contain the user ID, Model 204 assumes it is a CA-ACF2 login. Model 204 does a Generalized Resource check to see if User 0 is allowed to log in to Model 204. If the user ID is from CCASTAT, no check is done.

The rules for coding LOGIN for the IFAM1 IFLOG commands are the same as for User 0. For security reasons, Technical Support recommends that, if a CA-ACF2 Logonid is provided, code the LOGIN command without a user ID and password.

Single-user Onlines

Logins from single-user Onlines can come from a command in a CCAIN file or interactively from a terminal and have the attribute of being User 0. A login from CCAIN is similar to User 0 login from a multiuser Online or IFAM1 thread. However, a login from a terminal has characteristics similar to a user logging in to a MUSASS through an IODEV thread. Flexibility is provided by treating all LOGIN commands from CCAIN as User 0 logins. However, once the end of CCAIN is reached and commands can be entered interactively through the terminal, all LOGIN commands go through password, account, login privilege, and terminal security processing as if through an IODEV thread.

After the commands are issued interactively from the terminal, the single-user Online can usually issue a login for any CCASTAT or CA-ACF2 user ID. One exception is if the Online is issuing its own CA-ACF2 diagnose commands within the MODEL204 machine and the machine is not SRF-authorized. In this case, the user ID must either be from CCASTAT or be the Model 204 virtual machine Logonid. If the user ID is the virtual machine Logonid, Model 204 does not prompt for a password and the user is logged in automatically.

Maintaining the CA-ACF2 VM Interface

At each CA-ACF2 VM site, the installation security officer (ISO) performs system-wide maintenance functions within CA-ACF2. The security officer is responsible for defining and maintaining the CA-ACF2 Generalized Resource Rules for privilege security that will eventually be used by Model 204. In addition, the security officer also creates the ACF2PARM module, which defines the CA-ACF2 environment to Model 204. This section describes the Model 204 data that the security officer maintains.

Defining user privileges

You must identify Model 204 privileges to CA-ACF2.

The Model 204 privileges for CA-ACF2 users are defined as CA-ACF2 Generalized Resource Rules. The RESOURC argument of ACF2GEN is used

along with fixed CA-ACF2 key names. If an argument is not specified, RESOURC defaults to 204.

The following user privilege names define the possible privilege rules for a user:

This privilege...	Defines users who can...	Corresponding LOGCTL setting
'PRIV.SUPER.USER'	Exercise superuser privilege and can create a file with the CREATE command.	X'80'
'PRIV.SYSTEM.ADMIN'	Exercise system administrator privilege and issue commands such as LOGWHO, MONITOR, PRIORITY, and WARN.	X'40'
'PRIV.CHANGE.FILE.PASSWORD'	Change CCASTAT file passwords when opening a file.	X'20'
'PRIV.SYSTEM.MANAGER'	Issue system manager commands, that is issue all system administrator commands plus certain other privileged commands.	X'08'
'PRIV.OVERRIDE.RECORD.SECURITY'	Override record security.	X'04'

For CCASTAT users, privileges are defined to the CCASTAT user ID by the Model 204 LOGCTL command. The LOGCTL command contains an additional privilege bit (X'10') that indicates whether or not CCASTAT users can change their own login passwords at login time. This privilege bit is not required for the CA-ACF2 VM Interface, because you set this privilege for all CA-ACF2 users within CA-ACF2.

Security service machine

Once the security service machine is active, any MODEL204 machine can connect to it. You can set up multiple security service machines for test and production purposes. After a security service machine is initialized, it:

- Allows MODEL204 machines to connect to the security service machine via IUCV
- Performs all CA-ACF2 validation requests sent to it
- Relays return codes, reason codes, and data back to Model 204 for each validation request
- Allows MODEL204 machines to terminate and sever the IUCV connection

The IUCV connection between each MODEL204 machine and the security service machine is severed when the MODEL204 machine terminates.

The SECURID argument of the ACF2GEN macro specifies the Logonid of the security service machine to which the MODEL204 machine sends CA-ACF2 validation requests. If SECURID is left blank, CA-ACF2 diagnose commands are issued from within the MODEL204 virtual machine and a security service machine is not used.

Security service machine commands

The following commands control execution of the security service machine.

SECURITY

Use the SECURITY command to start execution of the security service machine.

The command format is:

Syntax EXEC SECURITY [*datasetname fm* | *fn ft fm*]

where:

<i>datasetname fm</i>	Specify the name and file mode of a file on a variable-formatted disk
<i>fn ft fm</i>	Specify the file name, file type, and file mode of a CMS data set

The security service machine writes information to the file including statistics for each MODEL204 machine connected with the security service machine, error messages, and terminal request commands.

STATUS

Use the STATUS command to request information while the security service machine is running. STATUS provides information on all MODEL204 machines connected and how many CA-ACF2 users are logged in through each Model 204 machine. For example:

STATUS

```
*****
***** STATUS:  NUMBER OF CONNECTED MODEL204 MACHINES = 4
***** STATUS:  MACHINES CONNECTED ARE: MODEL204 M204A
***** STATUS:  M204B      M204C
***** STATUS:  MODEL204  NUMBER OF LOGGED IN USERS = 5
***** STATUS:  M204A      NUMBER OF LOGGED IN USERS = 50
***** STATUS:  M204B      NUMBER OF LOGGED IN USERS = 11
***** STATUS:  M204C      NUMBER OF LOGGED IN USERS = 8
```

STOP

To terminate security service machine processing in an orderly fashion, use the STOP command. STOP prevents MODEL204 machines from initializing and establishing new IUCV connections, but allows machines with existing connections to continue until those MODEL204 machines terminate.

The command format is:

Syntax STOP

START

If you terminated processing with the STOP command, the START command reverses the process and allows new MODEL204 connections onto the security service machine.

The command format is:

Syntax START

FORCE

To end the connection of a particular MODEL204 machine with the security service machine, use the FORCE command. FORCE severs the IUCV connection between the two machines and forces the MODEL204 machine to terminate.

The command format is:

Syntax FORCE *virtual-machine-name*

PURGE

CA-ACF2 VM keeps a copy of the CA-ACF2 Generalized Resource Rules used by Model 204 in the security service machine's virtual storage. When the Installation Security Officer (ISO) changes a rule for any Model 204 key name and reloads the resource code with the ACFSERV command, the old rule remains in virtual storage until a PURGE is issued. PURGE flushes the old rule set from virtual storage. After a PURGE command is issued, CA-ACF2 acquires the new rule the next time the CA-ACF2 VM Interface asks CA-ACF2 to validate a Generalized Resource Request.

Note: To purge an old rule set from MODEL204 virtual storage when the CA-ACF2 VM Interface is running within the MODEL204 machine, you must terminate Model 204 and re-IPL the CMS virtual machine.

The command format is:

Syntax PURGE

EOJ

To terminate processing in the security service machine, use the EOJ command. Issue a STOP command first to prevent any new MODEL204 machines from establishing connections and give machines with existing connections time to terminate. If any machines have connections when you enter EOJ, the connections are severed and the MODEL204 machines terminate.

The command format is:

Syntax EOJ

CA-ACF2 statistics

For each MODEL204 machine, statistics are provided when Model 204 terminates. Information includes the number of:

- Successful logins
- Successful logouts
- Failed logins
- Successful and failed privilege requests

When the security service machine terminates, additional statistics include the maximum:

- IUCV connections allowed as specified in the VM directory of the security service machine (MAXCONN)
- IUCV connections established at any one time during execution
- Number of outstanding IUCV messages that were queued and waiting to be processed for that MODEL204 machine in the security service machine at any one time
- Number of messages allowed on a path as specified in the VM directory of the security service machine (MSGLIMIT)
- Number of outstanding messages in the queue that existed at any one time for all paths

Check these statistics occasionally to ensure that you are not reaching the limits set by MAXCONN and MSGLIMIT. Reaching these limits may affect VM performance.

If the number of IUCV connections established during execution is often close to or equal to MAXCONN, increase MAXCONN. The default for MAXCONN is four.

If the number of IUCV messages queued and waiting for processing is often close to or equal to MSGLIMIT, increase MSGLIMIT. The maximum for MSGLIMIT is 255; the default is 10.

For more information about MAXCONN and MSGLIMIT, see “Modifying the VM directory” on page 58.

Preparing ACF2PARM with ACF2GEN

The ACF2PARM parameter module allows you to define CA-ACF2 arguments for your site. ACF2PARM consists of one or more ACF2GEN macros that are coded, assembled, and generated by your site.

ACF2PARM is both a text deck and a module. To activate the CA-ACF2 VM Interface, an ACF2PARM text deck must be available to be dynamically loaded by an ONLINE module. If Model 204 is IFAM1 or an Online saved segment, the ACF2PARM module must also be available. If ACF2PARM is loaded, the CA-ACF2 VM Interface must initialize successfully before Model 204 can be initialized. If ACF2PARM is not available to be loaded, Model 204 initializes without the CA-ACF2 VM Interface active.

SECPLIST parameter

The SECPLIST User 0 parameter in CCAIN allows you to specify the name of the ACF2GEN argument set with which to initialize the interface. The name is defined by the assembler label name of the ACF2GEN macro.

CA-ACF2 does a Generalized Resource check to determine if a virtual machine running Model 204 is authorized to use a particular set of parameters specified by an ACF2GEN macro. CA-ACF2 looks at the resource code specified by the RESOURC argument of ACF2GEN and the key name PRIV.SECPLIST.name (where *name* is the name of an ACF2GEN macro). If the check fails, Model 204 cannot be initialized.

If the named set is not found in CCAIN, the default arguments of the ACF2GEN macro are automatically used, which allows Model 204 to try to initialize the Interface. However, Model 204 does not initialize under the following circumstances:

- If the CA-ACF2 system components do not contain the Model 204 rules assumed by the ACF2GEN default arguments
- If the MODEL204 machine is not authorized to use the default set of CA-ACF2 parameters

In the first example on page 52, LOG1, the resource code is 204, and the key name PRIV.SECPLIST.LOG1. In the second example, LOG2, the resource code is TST, and the key name is PRIV.SECPLIST.LOG2.

	TITLE 'ACF2PARM MODULE'	X
LOG1	ACF2GEN SECURID=SECUR204,	X
	RESOURC= 204 ,	X


```

ACCOUNT=(X'0141',X'0A'),          X
VALIDAT=LOGON,                     X
PASSWORD,                          X
PRTY=X'0140',                      X
LOGONID=

LOG2    ACF2GEN SECURID=,           X
        RESOURC= TST ,              X
        ACCOUNT=(X'0204',X'08'),    X
        VALIDAT=ACCOUNT,            X
        ACCTRES=ACT,                X
        NOPASSWORD,                 X
        RECSCTY=(X'020C',X'0A'),    X
        PRTY=S,                     X
        LOGONID=VMID
        ACF2GEN TYPE=END
END

```

If ACF2PARM is available on an accessed disk but no SECPLIST parameter was specified in CCAIN, the default values of ACF2GEN are used with a resource code of 204 and a key name of PRIV.SECPLIST.DEFAULT.

ACF2GEN macro

The ACF2GEN macro defines the way an installation uses the CA-ACF2 VM Interface. The arguments of the ACF2GEN macro are described in detail in this section.

Sample ACF2GEN macros are available with the distributed software in the ACF2PARM ASSEMBLE file. In the following example, all defaults are shown.

The format of the ACF2GEN macro is:

NAME	ACF2GEN	SECURID=,	<i>Security service machine</i>
		RESOURC=204,	<i>CA-ACF2 resource type for M204</i>
		ACCOUNT=(0,0),	<i>Acct field position & length</i>
		VALIDAT=,	<i>Validate login and/or acct</i>
		ACCTRES=,	<i>CA-ACF2 resource type for acct</i>
		NOPASSWORD/PASSWORD	<i>Password not req'd/req'd</i>
		RECSCTY=(0,0),	<i>Record security pos. & length</i>
		PRTY=S,	<i>Priority</i>
		LOGONID=,	<i>CCASTAT UID allowed/not allowed</i>
		TYPE=	<i>End of macro definitions</i>

where:

- *NAME* defines the name of this set of CA-ACF2 arguments. Each ACF2GEN macro must have a unique name. During Model 204

initialization, the name specified in the CCAIN parameter SECPLIST must identify a particular ACF2GEN macro with which to initialize the CA-ACF2 VM Interface. If SECPLIST is not defined in CCAIN, the default values of the ACF2GEN macro are used if ACF2PARM is loaded.

- *SECURID* specifies the virtual machine name of the security service machine. If SECURID is not coded or left blank, the CA-ACF2 diagnoses are issued from within the MODEL204 machine and not sent to a security service machine.
- *RESOURC* specifies the 3-character CA-ACF2 resource code for this installation. Model 204 user, login, and SECPLIST privileges are defined.

The code must match the resource code described to CA-ACF2 in the ACFFDR @RESTYPE macro.

The default code is 204.

- *ACCOUNT* specifies the position and length of the field within the Logonid record where the user's default account can be found. If a position is defined and the user does not enter an account on the LOGIN command, CA-ACF2 uses the default account in the user's Logonid.

If you do not specify the ACCOUNT argument, no account processing is performed.

ACCOUNT must be 1–10 characters in length.

The ACCOUNT argument must be specified if ACCTRES and VALIDAT=ACCOUNT are set.

The rules for specifying position and length are as follows:

- *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. The field must be defined in the LIDREC dsect and by a @CFDE entry in the ACFFDR. Specify position as a 4-digit hexadecimal value (for example X'0132').
 - *Length* must be specified as a 2-digit hexadecimal value from 1–10 (X'01' to X'0A').
- *VALIDAT* has two options: LOGIN and ACCOUNT.

You can specify parameters in any order, but they must be enclosed in parentheses and separated by a comma. For example:

```
VALIDAT= (LOGIN, ACCOUNT)
VALIDAT= (ACCOUNT, LOGIN)
VALIDAT=ACCOUNT
VALIDAT=LOGIN
```

- *LOGIN* specifies that Model 204 determines if a user is authorized to log in to Model 204. If you do not specify VALIDAT=LOGIN, all users are allowed access, provided that they pass the Logonid/password and optional source and account validation. If LOGIN is specified, Model 204 performs a Generalized Resource check using the type specified in RESOURC and the 'PRIV.LOGIN.M204' key name.

- *ACCOUNT* parameter (not to be confused with the *ACCOUNT* argument described previously) specifies that any account entered by the user during the login process is validated by CA-ACF2.

VALIDAT=ACCOUNT causes the account value entered by the user to be compared against the account value found in the user's Logonid record. If the values match, the user is allowed in to Model 204. If the values do not match, the account is validated using the *ACCTRES* resource type and the account value as the key name. If this validation is successful, the user is allowed in to Model 204. Otherwise, the login fails.

The *VALIDAT=ACCOUNT* parameter can be specified only if both *ACCTRES* and the *ACCOUNT* arguments are specified.

- *ACCTRES* specifies the 3-character CA-ACF2 resource code selected to validate the account value entered by the user during the login process. The code must match a resource code defined in the *@RESTYPE* macro of the *ACFFDR* for this installation. *ACCOUNT* and *VALIDAT=ACCOUNT* must be specified if *ACCTRES* is present.

There is no default and it can be the same code defined by the *RESOURCE* argument.

- *NOPASSWORD* directs Model 204 to bypass the password prompt if the *LOGIN* command contains a user ID equal to the CMS user ID of the virtual machine on which the login originated.

If they are not equal, Model 204 prompts for a password.

- *PASSWORD* specifies that a password is always required.

When the CA-ACF2 VM Interface is active, the NP CMS and P CMS options of the *LOGCTL* command are ignored for CA-ACF2 and *CCASTAT* user IDs.

PASSWORD is the default.

- *RECSCTY* specifies the position and length of the field in the Logonid record where the record security key can be found. This value overrides the default, which is the Model 204 user ID.

The rules for specifying position and length are as follows:

- *Position* indicates the offset within the CA-ACF2 Logonid record where the data is stored. The field must be defined in the *LIDREC* dsect and by a *@CFDE* entry in the *ACFFDR*. Specify position as a 4-digit hexadecimal value (for example *X'0132'*).
- *Length* must be specified as a 2-digit hexadecimal value from 1–10 (*X'01'* to *X'0A'*).

- *PRTY* specifies the default user priority. The options are as follows:

- A 1-character value of H, S, or L for High, Standard, and Low, respectively, that defines the default user priority for all users logged in under CA-ACF2.

- The position of a field in the Logonid record where the priority for each individual user is specified. Position must be specified as a 4-digit hexadecimal value (for example X'0142').

The field must be defined in a @CFDE entry in the ACFFDR and the LIDREC dsect, and contain a 1-character value of either H, S, or L.

The default priority is STANDARD if the PRTY keyword is omitted or if there is an invalid character at the specified position in the Logonid record.

- *LOGONID* can be specified as VMID, JOBLID, or left blank. If left blank, only CA-ACF2 users can access the system; CCASTAT-defined users are not allowed to log in to Model 204.

If LOGONID is specified as JOBLID or VMID, CCASTAT users can access Model 204.

- *TYPE* indicates the end of the ACF2PARM module. TYPE must be coded only once, following all other ACF2GEN macros, and as the only argument in an ACF2GEN macro.

Sample ACFPARM module

The following sample ACF2PARM module contains two sets of CA-ACF2 parameters called LOG1 and LOG2.

```

      TITLE 'ACF2PARM MODULE'
LOG1  ACF2GEN  SECURID=SECUR204,           X
      RESOURC=204,                         X
      ACCOUNT=(X'0141',X'0A'),             X
      VALIDAT=LOGON,                       X
      PASSWORD,                           X
      PRTY=X'0140',                        X
      LOGONID=
LOG2  ACF2GEN  SECURID=,                   X
      RESOURC=TST,                         X
      ACCOUNT=(X'0204',X'08'),             X
      VALIDAT=ACCOUNT,                     X
      ACCTRES=ACT,                         X
      NOPASSWORD,                         X
      RECSCTY=(X'020C',X'0A'),             X
      PRTY=S,                             X
      LOGONID=VMID
      ACF2GEN TYPE=END
      END

```

Installing the CA-ACF2 VM Interface

Follow these steps to install the Model 204 CA-ACF2 VM Interface. Each item in the list is discussed in more detail in later sections.

1. Update the VM directory to include a new entry for SECUR204, the security service machine. Update the directory entries for MAINT204 and for all Model 204 service machines issuing CA-ACF2 diagnoses Online.
2. Create a new PROFILE EXEC for SECUR204. Update the PROFILE EXEC for MAINT204 and for all the Model 204 service machines issuing CA-ACF2 diagnoses Online.
3. Decrypt the object modules that comprise the CA-ACF2 VM Interface.
4. Generate the ONLINE, IFAM1, and SECURITY modules.
5. Apply a zap to the ACFDIAGC CSECT of the Model 204 SECURITY, ONLINE, and IFAM1 modules.
6. Save the Model 204 saved segments again.
7. Determine security requirements for each MODEL204 machine. Code the ACF2GEN macros. Assemble ACF2PARM and generate the ACF2PARM module.
8. Insert a CA-ACF2 Logonid record for SECUR204. Update the Logonid records for each virtual machine running Model 204.
9. Update the ACFFDR to include @SRF macros for the security service machine and any multiuser or single-user MODEL204 machines that authorize requests on behalf of other CA-ACF2 Logonids.
10. Update the ACFFDR to include in the @RESTYPE macro the new resource codes required to write CA-ACF2 Generalized Resource Rules to validate user privileges, CCAIN SECPLIST values, and optional login and account validations.
11. Update the ACFFDR to include @CFDE macro entries for three new fields on the Logonid record required for setting individual user default account values, priority, and record security keys. All three fields are optional depending on installation requirements.
12. Update the LIDREC dsect to include any of the three new fields defined by @CFDE entries in the ACFFDR.
13. Write CA-ACF2 Generalized Resource Rules for:
 - SECPLIST validations
 - Model 204 user privileges
 - Model 204 login privileges — optional
 - Account validations — optional
 - Terminal security — optional
14. Update the EXEC1 and EXEC2 execs specified in the ONLINE command when invoking Model 204 and the EXECname on the IFAM1 command to include a FILEDEF for the CA-ACF2 VM loadlib named ACFSRF.

15. Update the CCAIN files to increment NSUBTKS and include the SECPLIST parameter.

Modifying the VM directory

Before installing the interface, add a VM directory for the security service machine and modify the VM directories for the Model 204 maintenance machine and MODEL204 service machine. This section describes how to modify the directory entries needed for the CA-ACF2 VM Interface.

SECUR204 - Security Service Machine

Add a VM directory entry for the security service machine if CA-ACF2 diagnoses are to be made from a security service machine. A sample directory entry is:

```
USER SECUR204 password 2M 2M G
OPTION BMX MIH MAXCONN nnnnn
IUCV ALLOW MSGLIMIT limit
ACCOUNT account distcode
IPL CMS PARM AUTOCR
CONSOLE 009 3210 T userid
SPOOL 00C 2540 READER A
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A
LINK CMSowner 190 190 RR
LINK CMSowner 19E 19E RR
LINK MAINT204 193 193 RR
LINK CMSowner aaa bbb RR
MDISK 191 type start size volser MR
```

where:

MAXCONN	Is the maximum number of IUCV connections that can be open at one time. It represents the maximum number of MODEL204 machines that can communicating with the security service machine at the same time. The default is 4.
MSGLIMIT	Is the maximum number of outstanding messages allowed on any path authorized by this IUCV directory control statement. MSGLIMIT depends upon the number of outstanding IUCV security requests for any one Online. VM performance might be affected if this limit is reached. The default is 10 and the maximum is 255.
ACFSRF LOADLIB (CMSowner <i>aaa bbb</i>) supplied by CA- ACF2 VM	Must be available to the security service machine. Link to the appropriate minidisk by either providing a LINK entry in the VM directory entry or a LINK command in the PROFILE EXEC.

MAINT204 - Model 204 Maintenance Machine

The Model 204 maintenance machine requires the ACF2VM TXTLIB to be available on an accessed minidisk when generating the ONLINE, IFAM1, and SECURITY modules. Either the VM directory entry for MAINT204 must have a LINK statement or the PROFILE EXEC must include a LINK command to the minidisk containing ACF2VM TXTLIB.

MODEL204 - Model 204 Service Machines

Virtual machines running any configuration of Model 204 might or might not require an update to their VM directory entries. If a Model 204 virtual machine sends its security requests to the security service machine, no change is necessary. If, however, Model 204 issues the CA-ACF2 diagnoses itself Online, the ACFSRF LOADLIB must be available on an accessed disk. Either the VM directory entry for Model 204 must have a LINK statement or the PROFILE EXEC must include a LINK command to the minidisk containing the ACFSRF LOADLIB.

Creating the PROFILE EXECs

The next step is to create appropriate PROFILE EXECs for the security service machine, and modify the PROFILE EXECs for the MAINT204 and MODEL204 service machines. The commands needed in the PROFILE EXECs follow.

SECUR204 — Security service machine

The security service machine needs only to access the necessary disks and to start up the SECURITY EXEC. Therefore, a sample PROFILE EXEC for the security service machine is as follows:

```
/* SAMPLE PROFILE EXEC FOR SECURITY SERVICE MACHINE */  
  
'ACC 193 C'      /* access MAINT204's 193 mdisk  
*/  
'ACC bbb fm'     /* access the mdisk containing ACFSRF  
LOADLIB */  
'EXEC SECURITY SECURITY CCAUDIT A'  
'LOGOUT'  
exit rc
```

MAINT204 — Model 204 maintenance machine

If the VM directory entry does not have a LINK statement linking to the minidisk that contains the ACF2VM TXTLIB, there must be a LINK command in the MAINT204 PROFILE EXEC. An ACCESS command to the minidisk must also be included in the PROFILE EXEC regardless of where the LINK is located.

MODEL204 — Model 204 service machines

If the VM directory entries do not have a LINK statement linking to the minidisk that contains the ACFSRF LOADLIB, the PROFILE EXEC for the virtual machines running Model 204 and issuing CA-ACF2 diagnoses Online requires a LINK command. An ACCESS command to the minidisk must be included in the PROFILE EXEC.

Decrypting object modules

The object modules that comprise the CA-ACF2 VM Interface are distributed with the Model 204 software in an encrypted format. To include the interface, you must decrypt the object modules using M204CRYP.

Using M204CRYP

You must have the decryption key for the CA-ACF2 VM Interface available. The key is located on the Authorization Form packaged with your distributed software. The syntax for M204CRYP is:

Syntax M204CRYP DECODE ACF2 *key*

where:

key	Is the decryption key on the Authorization Form included with the distributed software.
-----	---

For more information about M204CRYP refer to the *Model 204 Installation Guide for IBM z/VM*.

Creating Model 204 modules and saved segments

To create the appropriate Model 204 modules for the CA-ACF2 VM Interface you must:

- Generate the security, ONLINE, and IFAM1 modules
- Apply appropriate Early Warnings and zaps
- Save Model 204 saved segments

This section discusses these steps in detail.

Generating the Model 204 modules

Regenerate the modules for the following Model 204 configurations:

- CMS
- ONLINE
- IFAM1

More more information about how to regenerate modules, refer to the *Model 204 Installation Guide for IBM z/VM*.

Note: The ACF2VM TXTLIB must be on an accessed minidisk when generating the ONLINE and IFAM1 modules or the ACFDIAGC object module is not linked in.

Generating the security module

If you did not generate an M204SCTY module for a previous release, you need to create it. M204SCTY runs the security service machine. M204GEN EXEC generates this module with the command:

```
M204GEN SECURITY
```

For more information about the M204GEN command, refer to the *Model 204 Installation Guide for IBM z/VM*.

Note: The ACF2VM TXTLIB must be available on an accessed minidisk when generating the SECURITY module or the ACFDIAGC text deck is not linked into M204SCTY.

Applying ACFDIAGC

Apply the following zap to the SECURITY, ONLINE, and IFAM1 modules. This is a special zap that is applied against the CA-ACF2 VM Release 3.1 ACFDIAGC CSECT. This zap allows the Model 204 MVS Interface to run with CA-ACF2 VM:

NAME	xxxx	ACFDIAGC
VER	0054	0ACA
REP	0054	4700,0000,0700
VER	0066	0ACA
REP	0066	4700,0000,0700

Note: The name of the new security module is M204SCTY.

Saving Model 204 saved segments

The CMS, ONLINE, and IFAM1 saved segments must be saved if the Saved Segment facility is used. For more information about saving a Model 204 saved segment again, refer to the *Model 204 Installation Guide for IBM z/VM*.

Creating ACF2PARM

To activate the CA-ACF2 VM Interface during Model 204 initialization, ACF2PARM must be available on an accessed minidisk. If Model 204 is running as an ONLINE module, ACF2PARM must be a text deck. If the Model 204 configuration is IFAM1 or Online running as a saved segment,

ACF2PARM must be a module. See “ACF2GEN macro” on page 53 for information on coding the ACF2GEN macros that comprise ACF2PARM.

A sample ACF2PARM ASSEMBLE is provided on the maintenance machine’s 194 disk.

Code the ACF2GEN macros for your security environment. When you finish, issue the following commands to assemble ACF2PARM and generate a module:

```
VMFASM ACF2PARM NCMS 204
COPY ACF2PARM TXT220 A = TEXT C
LOAD ACF2PARM (ORIGIN TRANS
GENMOD ACF2PARM MODULE C
```

Inserting the SECUR204 CA-ACF2 Logonid

The following CA-ACF2 Logonid record for the security service machine must be added to the Logonid database using the ACF2 subcommand INSERT:

```
INSERT    SECUR204 NAME (SECURITY 204 SERVER)  -
          PASSWORD (xxxxxxxx)                  -
          SRF ACCOUNT
```

The security service machine requires the SRF privilege. The SRF privilege allows a virtual machine to issue CA-ACF2 requests on behalf of other users.

SECURITY, ACCOUNT, or both privileges must be given to the security service machine. ACCOUNT is usually sufficient. SECURITY privilege is necessary only if a security officer logs in to Model 204 using a Logonid having the SECURITY privilege. The ACCOUNT privilege is required by the ACALT parameter block from the SRF, which the CA-ACF2 VM Interface issues.

Updating Model 204 CA-ACF2 Logonids

For each MODEL204 machine, decide if you want the CA-ACF2 diagnoses to be sent to the security service machine or if it can issue its own CA-ACF2 diagnoses. You do not need to change the Logonid records for MODEL 204 machines sending CA-ACF2 diagnoses to the security service machine.

MODEL 204 machines issuing their own diagnoses might need the SRF and ACCOUNT (and possibly SECURITY, see above) privileges depending upon the Model 204 configuration they are executing. A multiuser Online requires the SRF and ACCOUNT privileges:

- Batch or IFAM1 job does not need any of these privileges.
- Single-user Online might or might not need these special privileges depending upon whether or not the user ID in a single-user Online logs in using a CA-ACF2 Logonid different from its z/OS virtual machine user ID.

See “Single-user Onlines” on page 47 for more information on executing single-user Onlines.

Writing execute-only rules

Writing CA-ACF2 access rules to protect ACF2PARM TEXT and MODULE, M204SCTY MODULE, SECURITY EXEC, M204XFER, and the M204ONLN MODULE prevents users from bypassing CA-ACF2 security in Model 204. These rules ensure that, when required, Model 204 initializes with the CA-ACF2 VM Interface active. Execute-only rules ensure that ordinary users cannot thwart security.

Sample rules

The following sample rules provide execute privileges to all users and read, write, and execute privileges only to MAINT204.

```
$KEY(MAINT204)
$MODE(ABORT)
.....
* Control link access to MAINT204's 193 MDISK
V193.VOLUME UID(MAINT204) READ(A) WRITE(L) EXEC(A)
V193.VOLUME UID(MODEL204) READ(A) EXEC(A)
V193.VOLUME UID(*) READ(A) EXEC(A)

* Control access to ACF2PARM TEXT and MODULE
V193.ACF2PARM.* UID(MAINT204) READ(A) WRITE(L) EXEC(A)
V193.ACF2PARM.* UID(MODEL204) EXEC(A)
V193.ACF2PARM.* UID(*) EXEC(A)

* Control access to M204SCTY MODULE
V193.M204SCTY.MODULE UID(MAINT204) READ(A) WRITE(L)
EXEC(A)
V193.M204SCTY.MODULE UID(MODEL204) EXEC(A)
V193.M204SCTY.MODULE UID(*) EXEC(A)

* Control access to SECURITY EXEC
V193.SECURITY.EXEC UID(MAINT204) READ(A) WRITE(L) EXEC(A)
V193.SECURITY.EXEC UID(MODEL204) EXEC(A)
V193.SECURITY.EXEC UID(*) EXEC(A)

* Control access to the M204ONLN MODULE
V193.M204ONLN.MODULE UID(MAINT204) READ(A) WRITE(L)
EXEC(A)
V193.M204ONLN.MODULE UID(MODEL204) EXEC(A)
V193.M204ONLN.MODULE UID(*) EXEC(A)

* Control access to the M204XFER MODULE
V193.M204XFER.MODULE UID(MAINT204) READ(A) WRITE(L)
EXEC(A)
```

```
V193.M204XFER.MODULE UID(MODEL204) EXEC(A)  
V193.M204XFER.MODULE UID(*) EXEC(A)
```

M204IFM1 is not protected, because it can execute only as a saved segment. X204ONLN is most likely executed as a saved segment only. But if a user's virtual machine is large enough, it can run X204ONLN as a module and, therefore, needs to be protected.

Updating the ACFFDR and LIDREC DSECT

The CA-ACF2 VM Interface requires updates to the ACFFDR (ACF Field Definition Record). Change the @SRF and @RESTYPE macros as discussed here. Adding @CFDE macros and updating the LIDREC dsect is optional depending upon parameters coded in the ACF2GEN macro of ACF2PARM.

Adding @SRF macros

An @SRF macro must be added to the ACFFDR for the security service machine and for each MODEL204 machine that issues in-line CA-ACF2 requests on behalf of other users. The requirement for the @SRF macro is the same as that for the SRF Logonid privilege. The following sample @SRF macro is for the security service machine:

```
@SRF SECUR204,MLID=VM,MODE=ABORT
```

For more information about @SRF, see the CA-ACF2 VM documentation.

Updating @RESTYPE macro

Add the resource codes specified in the RESOURC and ACCTRES parameters of the ACF2GEN macro to the @RESTYPE macro in the ACFFDR. The following example defines 204 and ACT as the new resource codes along with the standard CA-ACF2 codes for VM:

```
@RESTYPE  
SIZE=64K,KEYTYPE=(204,ACT,ALG,GRP,DIA,IUC,VMC)
```

For more information about the @RESTYPE macro, refer to the CA-ACF2 VM documentation.

Adding @CFDE macros

An @CFDE macro must be coded in the ACFFDR for each new Logonid record field specified in the ACCOUNT, RECSCTY, and PRTY parameters of ACF2GEN. The @CFDE macro defines both an external Logonid record field name used, for example, in the ACF command and the symbolic label given to the Logonid field in the LIDREC dsect. A sample follows illustrating how to code the @CFDE macros for the three fields.

The ACFFDR contains the following @CFDE macro entries:

```
@CFDE  MACCOUNT,M204ACCT,CHAR,
        ALTER=SECURITY+ACCOUNT,
        LIST=ALL,
        FLAGS=RESTRICT,PRTN=1,RRTN=1
```

```
@CFDE  MRECSCTY,M204SCTY,CHAR,
        ALTER=SECURITY,
        LIST=ALL,
        FLAGS=RESTRICT,PRTN=1,RRTN=1
```

```
@CFDE  MPRTY,M204PRTY,CHAR,
        ALTER=SECURITY,
        LIST=ALL,
        FLAGS=RESTRICT,PRTN=1,RRTN=1
```

Note: M204ACCT, M204SCTY, and M204PRTY are symbolic names defined in the LIDREC dsect representing fields on the Logonid record. MACCOUNT, MRECSCTY, and MPRTY are the field names recognized by the ACF command.

Information on updating the LIDREC dsect is in the section “Updating the LIDREC dsect”. For more information about the @CFDE macro, refer to the CA-ACF2 VM documentation.

Updating the LIDREC dsect

The symbolic names described in the @CFDE macros must be added to the LIDREC dsect. These new fields are added to the user definition section by updating the USERLID or USERXLID copy members (or both) and saving the copy members in the ACF2VM MACLIB.

Continuing with the sample given above defining @CFDE entries, the following example defines the three fields in the LIDREC. Update the LIDREC dsect to include the following:

```
M204ACCT DS  CL10
M204SCTY DS  CL8
M204PRTY DS  C
```

For more information about updating the user definition refer to the CA-ACF2 VM documentation.

Note: The ACF2GEN macro requires a position on the LIDREC where the ACCOUNT, RECSCTY, and PRTY fields are defined. After the ACFFDR has been assembled (the next step), the hexadecimal positions of the fields can be determined by reviewing the ACFFDR assembled listing.

Assembling and reloading the ACFFDR

After the ACFFDR has been updated and the LIDREC dsect changed, the ACFFDR is reassembled and reloaded in order for the new ACFFDR options to take effect. Reload the ACFFDR with the command:

```
ACFSERV RELOAD FDR
```

For more information about how to assemble and reload the ACFFDR, refer to CA-ACF2 VM documentation.

Giving values to Model 204 Logonid record fields

After the ACFFDR has been reassembled and reloaded, the ACCOUNT, RECSCITY, and PRTY Logonid record fields need values that correspond with security requirements. Use the CA-ACF2 subcommand CHANGE to give values to these new fields. For example:

```
CHANGE M204USR1      MACCOUNT (ACCOUNTING) -  
                     MRECSCITY (RECSECKEY) -  
                     PRTY (S)
```

Writing rules

The CA-ACF2 VM Interface provides up to five types of rules for validating Model 204 user requests:

- CCAIN SECPLIST parameter name (required)
- Model 204 user privileges (required)
- Login privilege (optional)
- Account validation (optional)
- Source entry rules for terminal security (optional)

CA-ACF2 Generalized Resource Rules are stored on the Information Storage database and consist of a 3-character type code, a 1-40 character name that identifies a particular resource, and a set of individual rule entries defining who can access the resource.

For SECPLIST, login, and user privileges, the 3-character type code defaults to 204 unless specified differently in the ACF2GEN parameter RESOURC. Account validation has no default resource code and is specified by ACCTRES. The 1- to 4-character name is the \$KEY parameter specifying one of the fixed key names that describe Model 204 privileges or an account value. The UID string defines the users who can access the resource identified by the \$KEY.

The resource code and key names for each of the four types of CA-ACF2 Generalized Resource Rules are detailed below. The source entry rules for terminal security are also explained.

For more information about ACF2GEN, refer to “ACF2GEN macro” on page 53. For more information about writing CA-ACF2 Generalized Resource and Source Entry Rules, refer to the CA-ACF2 VM documentation.

Writing SECPLIST rules

SECPLIST is a CCAIN parameter that identifies what ACF2GEN macro in ACF2PARM to use for a particular Model 204 job. All Model 204 jobs that activate the CA-ACF2 VM Interface go through SECPLIST validation. A virtual machine must be authorized to run Model 204 with a particular set of security parameters delineated by an ACF2GEN macro.

The resource code to validate SECPLIST is defined by the RESOURC parameter of ACF2GEN. RESOURC defaults to 204 if not specified. The key name is PRIV.SECPLIST.DEFAULT or PRIV.SECPLIST.name. If CCAIN does not contain SECPLIST or the SECPLIST name is not found in ACF2PARM, the PRIV.SECPLIST.DEFAULT key name is used. If the name is found in ACF2PARM, it becomes part of the key name.

The following sample ACF session uses 204 as the resource code and the SECPLIST names of PROD, TEST and the default:

```
ACF
SET RESOURCE(204)
COMPILE
$KEY(PRIV.SECPLIST.PROD) TYPE(204)
  UID(authorized user) ALLOW
....
$KEY(PRIV.SECPLIST.TEST) TYPE(204)
  UID(authorized user) ALLOW
....
$KEY(PRIV.SECPLIST.DEFAULT) TYPE(204)
  UID(authorized user) ALLOW
....
STORE
```

Defining Model 204 user privileges

Rules that define Model 204 user privileges use the type code defined by the RESOURC parameter of ACF2GEN or default to 204. The five key names are for the five types of Model 204 privileges. Five sets of rules must be written. The fixed key names are:

- PRIV.SUPER.USER
- PRIV.SYSTEM.ADMIN
- PRIV.CHANGE.FILE.PASSWORD
- PRIV.SYSTEM.MANAGER

- PRIV.OVERRIDE.RECORD.SECURITY

Commands for entering user privileges are similar to CA-ACF2 rules shown in the sample ACF session.

Validating the login

Validating a CA-ACF2 user's privilege to log in to Model 204 is optional. This feature is activated if VALIDAT=LOGIN is coded in the ACF2GEN macro and results in a Generalized Resource Call. The resource code depends upon the RESOURC parameter of ACF2GEN. The fixed key name is: PRIV.LOGIN.M204.

Validating the account

The optional account validation feature is activated if ACCTRES, ACCOUNT, and VALIDAT=ACCOUNT are specified in the ACF2GEN macro. ACCTRES defines the resource code for the Generalized Resource Call. The key names are the account values themselves. For example, if a CA-ACF2 user entered FINANCE for an account, the key name is FINANCE.

```
ACF
SET RESOURCE (ACT)
COMPILE
$KEY (FINANCE) TYPE (ACT)
  UID (authorized user) ALLOW
....
$KEY (ACCOUNTING) TYPE (204)
  UID (authorized user) ALLOW
....

STORE
```

Writing source entry rules

The CA-ACF2 VM Interface uses the z/OS virtual machine name as the physical terminal source of the user at Model 204 login time. If the CA-ACF2 Logonid specified on the Model 204 LOGIN command does not have a SOURCE field on its Logonid record, then no source entry rules are necessary. If the SOURCE field is present, source entry rules are required or else CA-ACF2 VM prevents the login.

A source entry rule correlates a physical source to a logical name. At login CA-ACF2 VM translates the physical source name to a logical name and then compares the logical name with the SOURCE field on the CA-ACF2 Logonid record. If the logical name and the SOURCE field match, login processing continues. If they do not match or no rule exists defining the physical source, the login is prevented.

For example, say z/OS user ID PSELLERS tries to log in to Model 204 using two different CA-ACF2 Logonids. The first time PSELLERS uses JANEDOE as the Model 204 user ID. If the JANEDOE Logonid record does not have a SOURCE field, CA-ACF2 VM does not perform terminal security validation and PSELLERS can log in to Model 204 if he knows the password and passes all other tests.

The second time PSELLERS enters the Logonid SMANAGER in the LOGIN command. The SMANAGER Logonid record has a SOURCE field with the value RM200. PSELLERS is denied entry to Model 204 unless an input source record or source group record is created that translates the physical source name of PSELLERS to the logical name of RM200. The following sample demonstrates the rule required:

```
ACF
SET ENTRY(SRC)
INSERT PSELLERS NEWDATA(RM200)
```

For more information about adding source entry rules, refer to the CA-ACF2 VM documentation.

Updating Model 204 EXECs

If Model 204 issues its own CA-ACF2 diagnoses Online and does not send its CA-ACF2 requests to the security service machine, a FILEDEF for the ACFSRF LOADLIB is required. Add the FILEDEF for the ACFSRF LOADLIB to the EXEC1 and EXEC2 EXECs that are part of the EXEC ONLINE command, and the EXECname EXECs that are part of the EXEC IFAM1 command:

```
FILEDEF ACFSRF DISK ACFSRF LOADLIB *
```

Without the FILEDEF, Model 204 receives the following messages:

```
DMSNXL589E MISSING FILEDEF FOR DDNAME ACFSRF
ACFV8028 NUCXLOAD FAILED - R0 HAS RC FROM CMS
```

For more information about the ONLINE and IFAM1 EXEC, refer to the *Model 204 Installation Guide for IBM z/VM*.

Updating Model 204 CCAIN files

The CCAIN files for ONLINE and IFAM1 configurations might require the following changes:

- Increase NSUBTKS by 1 if Model 204 is using the security service machine to issue CA-ACF2 diagnoses.
- Include the new parameter SECPLIST to specify which set of Interface parameters to use to run Model 204.

4

Model 204 Security Server (formerly RACF) Interface

In this chapter

- Overview
- Brief introduction to Security Server
- Running the Security Server Interface with Model 204
- Using the Security Server Interface
- Login processing
- Maintaining the Security Server Interface
- Installing the Security Server Interface

Overview

This chapter presents an overview of how Security Server works and describes the impact of the Security Server Interface on the Model 204 user, system manager, and Security Server security administrator. Installation requirements and a conversion task summary also are presented.

Security Server

Security Server is an extension of the IBM z/OS operating system that provides comprehensive data security. The Security Server Interface allows a Model 204 installation to use Security Server services within the Security Server environment.

Model 204 Security Server Interface

The Model 204 Security Server Interface allows an orderly migration from Model 204 security to Security Server security and allows an installation to use a combination of either or both security methods. The interface provides the tools and instructions necessary to:

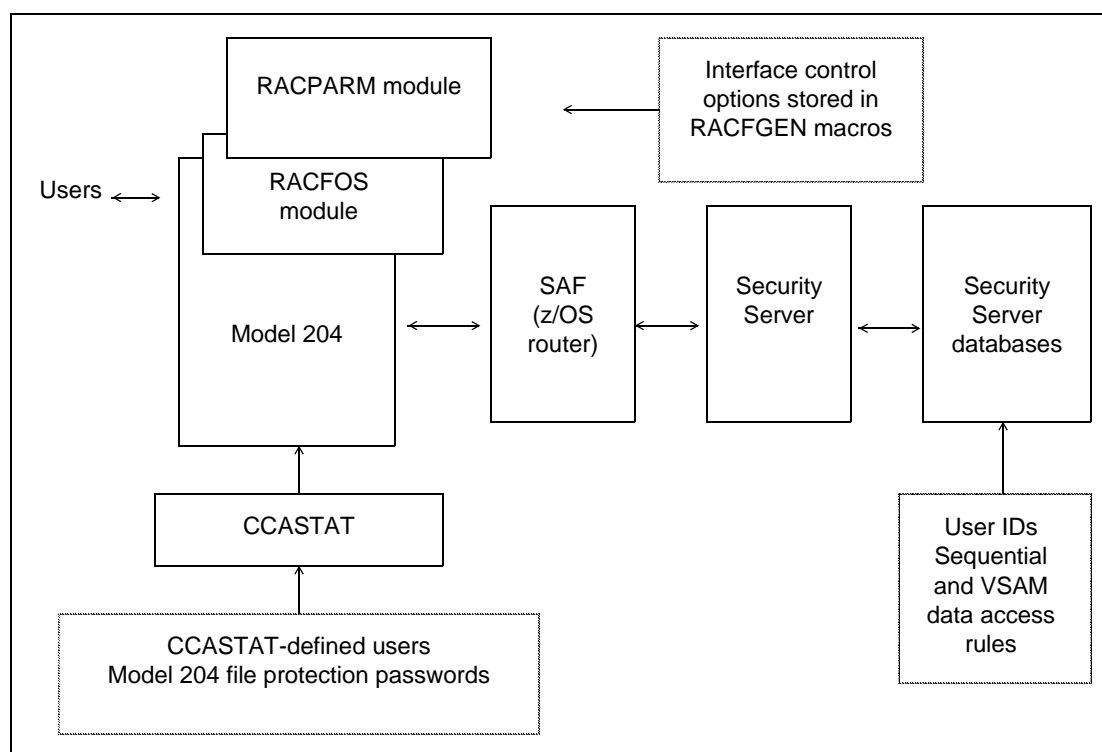
- Define a Security Server control group so that Model 204 can distinguish between separately running copies (that is, test and production)
- Define Security Server generic profiles for Security Server authorization of Model 204 user privileges within a control group
- Define a generic profile within Security Server that is associated with a control group, and determines whether a user can log in to Model 204
- Define a Security Server common control group for Model 204 resources that can be shared between running copies
- Define a default user priority class, or define the Security Server generic profiles that permit certain users to have specified Model 204 priorities
- Provide a mechanism to validate accounting information
- Determine a generic profile within Security Server that can define whether a user is allowed to submit JCL through the USE command
- Allow a system manager to add, change, or delete Security Server Interface options that control interface operations
- Provide a mechanism that establishes a shadow login capability to restrict user authorization via a default user ID for users who do not log directly in to Security Server
- Provide a method for a Security Server security administrator to define a default group, user ID, and password for the default user ID
- Provide a method for a Security Server security administrator to force users to log in through Security Server and not through CCASTAT

Model 204 Security Server Interface components

In summary, the Security Server Interface provides all the facilities an installation needs to implement Security Server security within the Model 204 environment.

Figure 4-1 illustrates the components of the Security Server Interface.

Figure 4-1. Security Server Interface



The Security Server Interface consists of the following components, which are described later in this chapter:

Model 204	Links with the Security Server Interface module
Security Server Interface module, RACFOS	Communicates with Security Server by using the System Authorization Facility (SAF) z/OS router
RACFPARM module	Can be linked with Model 204 or loaded dynamically at initialization
Model 204 system file, CCASTAT	Contains file protection passwords, user IDs, and user ID passwords

Brief introduction to Security Server

This section provides a brief summary of key Security Server features that are used in the discussion of the Model 204 Security Server Interface.

For more information about the Security Server, see the documentation associated with that product.

Security Server processing

Security Server identifies and verifies users accessing a system in which Security Server is installed and determines whether or not the user:

- Is defined to Security Server
- Has supplied a valid password (and/or operator identification card) and group name
- Has the REVOKE attribute, which prevents a Security Server defined user from entering the system at all or from entering the system with certain groups
- Can use the system only at certain times or on certain days as specified by the system manager or Security Server security officer
- Is authorized to access the terminal (which also can include day and time restrictions for accessing that terminal)
- Is authorized to access the application

The following types of checking are performed by Security Server:

- Authorization
- Global access

Authorization checking

When Security Server verifies a user's identity, Security Server specifies the scope of that user's authorization for the current terminal session or batch job in a control block called the accessor environment element (ACEE). Security Server controls the interaction between the user and protected resources and authorizes not only which resources the user can access, but the way in which the user can access them (for example, as read-only or update).

At the start of processing, Security Server performs the following checks that are related to the security classification of users and data:

- Compares the security level in the user and resource profiles. If the resource has a higher security level than the user, Security Server denies the request.
- Checks a category list (that is, a list of site-defined names corresponding to departments or areas within an organization) in the user's profile against the category list in the resource profile. If the resource profile contains a category that is not in the user's profile, Security Server denies the request.

If Security Server has not denied the request as a result of the security classification checks, processing continues.

Security Server permits access to a resource if the user satisfies any of a number of conditions, such as:

- Resource is a data set and the high-level qualifier is the user's user ID
- User's user ID is in the access list with sufficient authority
- User's current connect group is in the access list with sufficient authority
- When list-of-group checking is active, one of the other groups with which the user is connected is in the access list with sufficient authority
- Universal access authority (UACC) is sufficiently high
- User has the OPERATIONS attribute (or the resource is within the scope of a group in which the user has the group-OPERATIONS attribute) and the user's user ID or any group name the user is connected to is in the access list with an authority not less than the user's intended access

Security Server provides two installation exits that a site can use during RACHECK processing, as well as a quicker form of authorization checking called FRACHECK, or fast path RACHECK processing.

Global access checking

In addition to authorization checking, Security Server also provides global access checking. Global access checking allows a site to store a system-wide table of default authorization levels for selected resources. Security Server checks this table to determine if access to a resource is permitted. If access is permitted, Security Server bypasses further RACHECK processing. If the global access check fails, RACHECK processing continues. Frequently accessed resources with generalized access rules are good candidates for global access checking.

Global access checking handles resources in both the data set and the general resource classes, with the exception of group resource classes. Global access checking characteristics are that it:

- Grants access, but does not, by itself, deny any request for access
- Does not allow logging of permitted access or gather statistics
- Does not offer a postprocessing installation exit
- Performs no I/O on the Security Server data set, thus offering a high-performance checking option

Using profiles to protect resources

Security Server protects data sets (both DASD and tape) and general resources such as tape volumes and terminals. These resources are protected through profiles defined by the site. Authorized users can create, modify, list, and delete profiles using Security Server commands.

When the ADDSD statement is used to define and protect a data set, Security Server builds a data set profile and stores it in the Security Server data set.

Profiles can be either generic or discrete depending on the nature of the resource.

Generic profile

A generic profile protects a single resource such as one data set, or a number of related resources; for example, those having similar naming structures, or the same access, authorization, and auditing requirements.

A generic profile contains a list of authorized users and the access authority of each user. A single generic profile can protect many data sets that have similar naming structures. Data sets protected by generic profiles need not be defined individually to Security Server, which saves the system manager from having to enter and maintain individual profiles.

Many users can protect all their data sets with a single generic profile consisting of their user ID and an asterisk:

```
user_id.*
```

This profile protects all the user's data sets that begin with the user's user ID, regardless of the number of qualifiers.

Discrete profile

A discrete profile protects resources that have specific or unique access authorization or logging requirements such as a single sensitive data set on a specified volume.

A discrete profile contains the same kind of information as a generic profile, but it protects only the one identified data set on the specified volume or volumes.

If the access authorization requirements are general, define a generic profile. If the data set has unique access authorization requirements, define a discrete profile.

Either type of profile can protect tape data sets and:

- Cataloged and uncataloged non-VSAM data sets
- VSAM data sets
- Data sets with the same name residing on different volumes
- Generation data group (GDG) data sets
- Data sets and catalogs with single level names via a site-supplied prefix

In Model 204, the Security Server Interface user privileges and other authorization items are defined as data set names and treated as system resources. Generic profiles control who can use those resources by permitting individual users to access them. These rules are written by the Model 204 system manager and the Security Server security administrator.

Running the Security Server Interface with Model 204

When running either as an Online or in batch mode, Model 204 checks Security Server for the system resources a user has permission to access. Model 204 manages the authorization process, because only Model 204 can identify the user requesting a resource. Model 204 asks Security Server to validate those requests and responds to users who have been denied requests for resources.

Model 204 interfaces with Security Server in the following distinct areas:

- User logins
 - LOGIN/LOGON command
 - IFSTRT function
 - IFLOG function
- User resource requests
 - Dynamic allocation of new DASD space (ALLOCATE command)
 - Job submission (USE \$JOB command)
 - Sequential data set handling (Record I/O)
 - VSAM data set handling (External I/O)

Note: Because Model 204 databases are not validated on behalf of the user by the security interface, Security Server must grant the owner of an address space permission to open Model 204 file data sets for update, regardless of whether the owner has write or read-only privileges. This allows Model 204 to write back updates to the FPL (File Parameter List) page, as required by the database management system, regardless of the Model 204 file open privileges.

- User logouts
 - LOGOUT/LOGOFF command
 - IFFNSH/IFDTHRD function

The Security Server Interface uses the standard IBM-supplied calls to validate user requests for Model 204 resources. No source code modifications or exits to the Security Server software are needed. However, you can define an additional Security Server control group for nonshared resources as well as for resources shared between multiple copies of Model 204.

Using the Security Server Interface

This section describes the changes that affect a Model 204 user. They include:

- LOGIN or LOGON
- IFSTRT function
- IFLOG function within IFAM1

- Dynamic allocation
- Job submission
- Sequential and VSAM data set handling

LOGIN or LOGON command

The LOGIN/LOGON command allows you to gain access to Model 204. Once you are connected to Model 204, and if the system manager has set Model 204 to require logins, any commands you enter (other than LOGIN or LOGON) produce the following message:

```
*** PLEASE LOGIN
```

To log in, enter:

```
LOGIN userid [account]
```

or

```
LOGON userid [account]
```

where:

<i>userid</i>	Is a character string that identifies you
<i>account</i>	Is an optional character string that identifies the account under which you log in

User ID information

If Model 204 provides security authorization checks, the system manager assigns a user ID, password, and user privileges to every Model 204 user consisting of:

- User ID that identifies you to Model 204
- Password that provides access to the system
- User privileges associated with your user ID and password to define the particular type of access that you have
- Default priority class assigned to your user ID

When native Model 204 security is in effect, this ID information is stored in a record in the system file CCASTAT. This record is deleted from CCASTAT when the system manager moves the user into Security Server security mode. The RACFPARM module (see “Preparing a RACFPARM parameter module with RACFGEN” on page 90) supplies a default Logonid for CCASTAT Logonids to do validation requests on Security Server.

Security Server processing

When Security Server performs login validation, the user ID must be eight characters or less (seven characters if jobs are submitted through the internal reader). Your user ID is verified by Security Server before you can log in to Model 204.

If your site chooses to perform Security Server account validation, any value entered in the account field is verified via Security Server services. When Security Server is performing account validation, the account is limited to a maximum length of eight characters. If you do not enter an account, the default account becomes your user ID. Refer to “Login processing” on page 82 for more information on login processing.

If you have the appropriate authority, you can change your Security Server user ID password when you log in to Model 204. For more information about the LOGIN and LOGON command, refer to the *Rocket Model 204 Parameter and Command Reference Manual*.

IFSTRT function

In the IFAM1 and IFAM4 environments, the IFSTRT function is used in a Host Language Interface program to allocate a Host Language Interface thread. IFSTRT also establishes the calling convention, performs a user login, and determines whether or not the thread has updating privileges.

When processing the login argument of the IFSTRT call, the user login process follows the rules for User 0 login (refer to “User 0 login” on page 83).

IFLOG function within IFAM1

The IFLOG function is used only in the IFAM1 environment. It is called following IFSTRT to identify you when a login is required. This function is necessary in any IFAM1 program where user authorization is validated by Security Server.

As with the IFSTRT function, the user login process is the same as the process for User 0 login (refer to “User 0 login” on page 83).

Dynamic allocation considerations

All dynamic allocation services in a Security Server environment are subject to Security Server system rules for data set validation. For example, data set allocation can be restricted to allow the creation of data sets with certain name patterns. The rules for data set access are determined by the Security Server security administrator.

To dynamically allocate a *new* data set, you must have Security Server ALTER authority. If you issue an ALLOCATE command and you do not have ALTER authority, you receive a Model 204 error message.

If you log in to Model 204 via CCASTAT, your allocation privileges are determined by the Security Server default user's authorization.

Note: When you open a new or existing sequential or a VSAM data set, it is validated for read/write access. Security Server does not perform open access authorization on Model 204 data sets, because these data sets are currently protected under Model 204 security.

Job submission considerations

When an Online user issues the USE \$JOB command to submit a batch job, Model 204 identifies the user so that Security Server can determine if the user has proper authorization. If the submitter is IFAM1, IFAM4, or User 0, no additional processing occurs; the submitted job inherits the privileges from the submitting job, because it has already been authorized to run by Security Server.

In a Model 204 system in which the Security Server Interface is running, the following JCL statement is appended automatically to the submitted job statement(s) to identify the Model 204 user:

```
// USER=userid,PASSWORD=password,GROUP=groupname
```

where:

<i>userid</i>	Is the submitting user's Security Server user ID, or the default user ID if the user is logged in through CCASTAT.
<i>password</i>	Is the user password associated with the particular user ID, or the default password for the default user ID if the user is logged in through CCASTAT.
<i>groupname</i>	Is the Security Server group to which the submitting user is assigned. This is the group identified in the user's ACEE that is provided by Security Server during user login, or the default group if the user is logged in through CCASTAT.

The default user ID, password, and group information is defined in the Security Server Interface parameter module (RACFPARM), and cannot be modified by the user or controlled by the system manager. RACFPARM is described later in this section.

If Model 204 finds any of the Security Server control parameters listed previously while processing the submitted JOB, they are deleted from the JCL to prevent users from violating security.

An installation can request that Model 204 check for user authority to submit a job. Because Security Server does not have a facility to restrict job submission, Model 204 checks the generic profile for the comgroup.SUBMIT option. Comgroup.SUBMIT determines whether or not a user is permitted to submit JCL. For more information about comgroup.SUBMIT, refer to the RACFGEN

macro discussion in “Preparing a RACFPARM parameter module with RACFGEN” on page 90.

If an unauthorized user attempts to submit a job, Model 204 issues an error message.

For users logged in via CCASTAT, the authority to submit jobs is determined by the default user’s authorization.

Sequential and VSAM data set considerations

Security Server always performs an authorization check if you try to use a sequential or VSAM data set. To use a sequential or VSAM data set you must be a member of a control group (or optionally a common control group) that has:

- Security Server alter or update privileges to open a deferred update data set or to issue the DUMP or USE commands
- Security Server alter authority to dynamically allocate a new sequential data set
- Security Server read privileges to read sequential files from User Language or to read the file specified in a RESTORE command
- Security Server read privileges to read external VSAM files from User Language.

If you issue a sequential or VSAM data set OPEN command that fails for Security Server reasons, Model 204 issues an error message.

If you log in to Model 204 via CCASTAT, authorization is determined by the default user’s authorization limits.

Model 204 sequential file data sets (such as CCASTAT, CCAAUDIT, CCAJRNL, or CCAJLOG) are also checked to determine if the owner of the address space has the authority to write to them.

Security Server directly checks the following Model 204 commands:

- DUMP and DUMPG
- RESTORE and RESTOREG
- USE OUTXXX
- ALLOCATE a new data set
- OPEN DATASET
- OPEN *filename* with deferred update

Login processing

The system manager is responsible for defining security processing in a Model 204 installation in which Security Server supplies authorization services. This section describes the login processing performed by the Security Server Interface.

Login processing

When a user logs in to Model 204, Model 204 acquires the user ID and account and searches CCASTAT to determine whether or not the user ID exists. If the user ID is found, Model 204 queries the user for a password and proceeds with authorization processing. If the RACFPARM module indicates that CCASTAT defined users cannot log in, the login fails. If the user ID is not found in CCASTAT and Security Server security is in effect, Model 204 uses Security Server facilities to authorize the user login:

- Model 204 passes the login user ID and password to Security Server to verify that the user is a valid Security Server user and that the password is correct.
- If the site has chosen to verify that the user is authorized to use Model 204, Security Server is asked if the user has PERMIT authority to access the generic profile for resource group.LOGIN. If this check is successful, account processing occurs.
- If the installation has chosen to verify the account, the value entered is checked against the Security Server profile for comgroup.ACCOUNT.account. If the value entered has PERMIT authority, the user is allowed into Model 204.

If Security Server denies system access because of an invalid password or account, or if the user is not authorized to use Model 204, no login occurs and Model 204 issues an error message.

Once users are successfully logged in, they are granted Model 204 privileges of X'00'. Any additional privileges are determined whenever they enter a Model 204 command that requires a specific privilege.

Users who do not log in directly to Security Server are automatically restricted in what they can do by the privileges associated with the default user ID. Refer to "Security Server default user ID" on page 94 for more information.

All the system manager commands concerning CCASTAT maintenance functions work as usual. ZBLDTAB also functions as usual when creating the initial CCASTAT.

File, group, and subsystem security functions are defined as described in the Model 204 documentation.

User 0 login

When the Security Server Interface is active, User 0 is validated as the owner of the Model 204 run, regardless of whether the run is Online, BATCH204, IFAM1, or IFAM4. Model 204 always attempts to log in User 0 automatically, and verifies that any user ID supplied matches the user ID of the owner of the address space. It is not necessary to supply a user ID on the login command for User 0; Model 204 determines the owner user ID and supplies it automatically.

If a user ID is supplied on the login command and the user ID does not exist on CCASTAT, it must match the user ID of the owner of the address space or the login fails. If the user ID is found on CCASTAT and CCASTAT users are allowed to log in, all further authorization checking is based on a default user ID specified by the installation (if running as an APF-authorized program).

BATCH204 and IFAM4 can run without APF authorization if the user ID is equal to the ADDR SPACE ID and there is no default login ID (you cannot login from CCASTAT). If you try to log in without meeting these conditions, your login fails or a B78 System ABEND occurs.

Technical Support recommends that you do not specify a user ID in the login for User 0, so jobs can be easily migrated from test to production without having to change the login command.

Whether or not you supply a user ID, no password is needed and Model 204 does not prompt for one. If a password statement is coded in the input stream, it is treated as an invalid Model 204 command. If the user ID is on CCASTAT, Model 204 prompts for a password. If the password is correct, the login succeeds but all future data set authorization checking is based on the default user ID.

AUTHCTL command

The AUTHCTL command is available to view or delete control information from releases of Model 204 before Version 2.1. If your site started using Security Server after Version 2.1, there is no information for you to view. The AUTHCTL functions were replaced by the RACFGEN macro, discussed in “Preparing a RACFPARM parameter module with RACFGEN” on page 90.

The format of the AUTHCTL command is as follows:

```
AUTHCTL {function} RACF
```

where:

<i>function</i> option	Specifies the function to be performed. Options are as follows: <ul style="list-style-type: none">• <i>D</i> deletes the Security Server control information from the CCASTAT data set• <i>LIST</i> displays the interface control options• <i>VIEW</i> displays the interface control options currently in effect for the interface that is active
------------------------	---

The data originally stored by the AUTHCTL command in Release 9.0 or earlier versions of Model 204 is retrieved by Model 204 only during initialization. (This data can be displayed by issuing an AUTHCTL LIST Security Server command.). For information stored after Release 9.0, see the RACFGEN macro, in “Preparing a RACFPARM parameter module with RACFGEN” on page 90.

Note: If you do not delete the Security Server Interface control record from CCASTAT, it is used during initiation, and the values in RACFPARM override all values except the authorization bits.

The VIEW option of the AUTHCTL command displays the interface options in effect for the interface currently in operation. For example, you enter:

```
AUTHCTL VIEW
```

The following list is displayed (assuming this information was used during initialization):

```
Security Server INTERFACE OPTIONS
GROUP          M204TEST      Security Server control group name
COMGROUP       M204COM       Security Server common control group
name
VALIDATE       LOGIN        Validation option in effect
VALIDATE       ACCOUNT      Validation option in effect
PRIORITY       LOW          Priority default
DLMCHECK       Use $JOB DLM checking option
                M204GRP      Default user group
                M204USR      Default user ID
```

The additional lines displayed indicate the current default user ID and group in use. For more information about defaults, refer to “Security Server default user ID” on page 94.

Using trusted login for CRAM users

If your site uses CRAM, you can use the trusted login feature, which allows users to issue login commands or calls that do not include a user ID and password. For a user to log in as trusted, the user ID must be defined to Security Server. User IDs defined to CCASTAT are not allowed to log in as trusted users. CICS users must be using the Security Server interface for CICS.

Only CICS user IDs that log in through Security Server can log in as trusted users.

Trusted login can be used with the following CRAM thread IODEV types:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

To set up trusted login for a user, set the SECRTLOG user parameter, as discussed in “Setting up the SECTRLOG parameter for trusted login” on page 95.

Login processing for trusted login

For users connecting with a CRAM thread that allows trusted login, the user login processing routines are changed to handle a LOGIN command or IFSTRT call without a user ID and password:

- CRFS and CRIO channel threads handle User Language statements. These users ordinarily issue a LOGIN or LOGON request with the following format:

```
LOGIN userid [account] ; password [:new password] ; apsyn-  
ame
```

For trusted logins, the format is:

```
LOGIN ; ; apsynname
```

- The IFAM channel threads handle IFAM2 statements. These users ordinarily issue an IFSTRT or IFSTRTN call with the following format:

```
IFSTRT (RETCODE, LANG_IND, LOGIN, THRD_TYP, THRD_ID)
```

```
IFSTRTN (RETCODE, LANG_IND, LOGIN, THRD_TYP, THRD_ID, CHAN)
```

The LOGIN parameter is required and supplies the user ID and password as a character string using the following format:

```
`userid [account] ; password [:new password] ;`
```

For trusted logins, the statement format is the same but the login character string is a semicolon surrounded by single quotes (';').

Trusted login errors

When using trusted login, you might receive one of the following errors:

- Message below is generated when either:

- a. Model 204 job requesting the trusted login feature is running without a Model 204 Security Interface (CA-ACF2, Security Server, or CA-Top Secret).
- b. Model 204 address space does not have enough storage to allocate an internal work area for the trusted login feature. In this situation, the Model204 job does not initialize, because it still has to allocate storage for the Model 204 file buffers.

M204.2378: SECURITY TRUSTED LOGIN FEATURE DISABLED

- Following message is issued when the trusted user ID passed by CRAM is not between 1-8 bytes long:

M204.2379: INVALID TRUSTED USERID LENGTH = *length*.

Maintaining the Security Server Interface

At each Security Server site, the security administrator performs system-wide maintenance functions within Security Server. The security administrator is responsible for defining and maintaining the Security Server generic profiles and permits for authorization that are eventually used by Model 204. The following discussions describe the Model 204 data maintained by the security administrator.

The privilege names that Model 204 uses are based on generic data set profiles. Although the Security Server ADDSD command describes a resource profile for Model 204, use it as if you are defining a data set.

This method provides the simplest way to describe resources. You do not need to reassemble any Security Server modules because all definition is external. In addition, it is portable across Security Server releases, because no internal modifications are made to Security Server tables and no Security Server exits are used.

In the following examples, the simplest set of conditions is described. You can fully utilize all Security Server options (for example, auditing when a resource is used, changing the universal access authority, or naming a specific owner of the group). Use security access levels for both the profiles and the permits for authorized users of those profiles.

The interface requires that users of these resources are permitted READ authority.

Defining user privileges

You must identify user privilege names as Security Server generic data set profiles. The standard Model 204 user privileges described earlier have been assigned fixed names. The following set of user privilege names defines the possible privilege types for a user. These names are further qualified by the control group name entered in the RACFPARM module.

These user privilege profile names are described in Table 4-1.

Table 4-1. User privilege names

User privilege name...	Defines a user who can...	Corresponding LOGCTL setting
'group.PRIV.SUPER.USER'	Exercise superuser privileges: a user with READ access to this profile can execute a CREATE command.	X'80'
'group.PRIV.SYSTEM.ADMIN'	Exercise system administrator privileges: a user with read access to this profile can issue system administrator commands such as LOGWHO, MONITOR, PRIORITY, and WARN.	X'40'
'group.PRIV.CHANGE.FILE .PASSWORD'	Change the file password that is used to open a file: valid only if the file entry is stored in CCASTAT.	X'20'
'group.PRIV.SYSTEM .MANAGER'	With system manager privileges: a user with read access to this profile can issue all system administrator commands along with certain other privileged commands such as LOGCTL, AUTHCTL, and DUMPG.	X'08'
'group.PRIV.OVERRIDE .RECORD.SECURITY'	Override record security.	X'04'

Defining corresponding generic profiles

Security Server generic data set profiles must be developed and defined to Security Server to correspond to the defined privileges. The Security Server generic data set profiles used to define the standard Model 204 privileges and other resources to Security Server in a batch TSO execution are as follows:

```
//JOBNAME JOB (other information).....
//DOIT EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=A
//SYSTSIN DD *
ADDGROUP group DATA('MODEL204 Security Server CONTROL GROUP NAME')
ADDSD 'group.PRIV.SUPER.USER*' UACC(NONE) GENERIC
ADDSD 'group.PRIV.SYSTEM.ADMIN*' UACC(NONE) GENERIC
ADDSD 'group.PRIV.CHANGE.FILE.PASSWORD.*' UACC(NONE) GENERIC
ADDSD 'group.PRIV.SYSTEM.MANAGER*' UACC(NONE) GENERIC
ADDSD 'group.PRIV.OVERRIDE.RECORD.SECURITY.*' UACC(NONE) GENERIC
ADDSD 'group.LOGIN*' UACC(NONE) GENERIC

ADDGROUP comgroup DATA('MODEL204 Security Server COMMON GROUP NAME')
ADDSD 'comgroup.SUBMIT*' UACC(NONE) GENERIC
ADDSD 'comgroup.PRIORITY.HIGH*' UACC(NONE) GENERIC
ADDSD 'comgroup.PRIORITY.STANDARD.*' UACC(NONE) GENERIC
ADDSD 'comgroup.PRIORITY.LOW*' UACC(NONE) GENERIC
```

ADDSD

'comgroup.ACCOUNT.nnnnnnnn.*'

UACC(NONE) GENERIC

Note: *nnnnnnnn* represents an account number; there are as many as required. Also, high-level qualifiers cannot exceed eight (8) characters.

In the following two sample PERMIT statements, the first is for a system manager and the second is for login authorization to a copy of Model 204:

```
PERMIT      'group.PRIV.SYSTEM.MANAGER*'      ACCESS (READ)
            ID (user ID1,user ID2.....)
PERMIT      'group.LOGIN*'      ACCESS (READ)
            ID (user ID1,user ID2.....)
```

where:

<i>group.PRIV.privilege</i>	Specifies one of the fixed types that Model 204 uses to build Security Server retrieval search keys for the rules
<i>group</i>	Is the Security Server control group defined to Model 204

The previous Security Server PERMIT statement sample permits the specified users to access the named resource. Rules for determining the authorized user are described in the Security Server documentation. The privilege rules are tested whenever a user issues a command that requires a specific privilege. If the user is authorized to have that privilege, the command succeeds. If not, the user typically receives a Model 204 error message and the attempt might be logged as an Security Server violation.

Installing the Security Server Interface

The Model 204 interface to Security Server consists of the following components:

- Code in Model 204 that utilizes Security Server facilities
- Security Server group and generic data set profile definitions defined at your site
- SECPLIST User 0 parameter in CCAIN

The user site requirements that must be met in order to activate the Security Server Interface are identified in this section.

Terminal security considerations

Security Server can enforce terminal security when a user logs in. At that time, the source of the login is passed to Security Server. For systems running Model 204 with the SNA Communications Server (formerly VTAM) interface, this poses no problem because Model 204 recognizes the terminal name and stores it for use.

Configurations using CRAM (TSFS, CICS, IFDIAL) must use the CRAM channel name instead of the terminal ID as the user source, because there is no secured mechanism in CRAM for identifying the source of the user. Because CRAM channel names can be identified as valid sources, the user IDs allowed to use those sources can be validated. Using the CRAM channel name temporarily ensures that entry to Model 204 is from an approved source.

Decrypting Security Server

As part of INS204, the M204DECR job is generated with all the steps needed to decrypt Security Server. Refer to the *Model 204 Installation Guide for IBM z/OS* for more information.

SECPLIST parameter

The SECPLIST User 0 parameter in CCAIN lets you specify the name of the RACFGEN argument set with which to initialize the interface. The name is defined by the assembler label name of the RACFGEN macro.

- If the SECPLIST parameter is not in CCAIN, RACFPARM is used as the default name of the argument set.
- If no match is found for the SECPLIST name or the RACFPARM default name in the RACFPARM module, the interface is initialized using the precoded default parameters defined in the Security Server Interface. In this case, Login, Account, and Priority validation are not performed.

The following RACFPARM module contains two sets of Security Server arguments. In the first set, the name is LOG1 and account security is in effect. If the user is logged on through CCASTAT, the default Logonid is M204USER. In the second set, the name is LOG2 and both account and login security are in effect. In addition, if the user is logged on through CCASTAT, the Logonid of the executing job is considered the default Logonid.

	TITLE 'RACFPARM MODULE'	X
LOG1	RACFGEN GROUP=M204PROD,	X
	COMGRUP=M204RACF,	X
	PRTY=H,	X
	VALIDAT=ACCOUNT,	X
	DEFUSER=M204USER,	X
	DEFGRP=,	X
	DEFPASS=,	X
	DMLCHECK	
LOG2	RACFGEN GROUP=M204TEST,	X
	COMGRUP=M204RACF,	X
	PRTY=S,	X
	VALIDAT=(ACCOUNT, LOGIN),	X
	DEFUSER=JOBNAME,	X
	DEFGRP=M204GRP,	X
	DEFPASS=DEFPASS,	X

```

DMLCHECK
RACFGEN TYPE=END
END

```

Preparing a RACFPARM parameter module with RACFGEN

The RACFGEN macro generates a set of arguments that govern login and other security processes. There can be multiple argument sets in a RACFPARM module. The RACFPARM parameter module can be linked with Model 204 or dynamically loaded when the Security Server Interface is initialized. This module replaces all data previously entered with the AUTHCTL command except the VALIDAT= data, which is merged, making it difficult to tamper with security.

Note: AUTHCTL might be deleted from the Security Server Interface in a future release and all control information will be generated by RACFGEN.

The arguments for the RACFGEN macro are described in detail in this section.

The following is a sample RACFGEN macro with one set of arguments. The name of the argument set is RACFPARM, which is the required name if you are linking statically into your Model 204 modules.

```
TITLE 'GENERATE A Security Server PARAMETER MODULE'
```

RACFPARM RACFGEN GROUP=M204RACF,	<i>Control group</i>	X
COMGRP=M204RACF,	<i>Common control group</i>	X
PRTY=S,	<i>Priority</i>	X
VALIDAT=,	<i>Validation items</i>	X
DEFUSER=M204USR,	<i>Default user ID</i>	X
DEFUGRP=M204GRP,	<i>Default user group</i>	X
DEFPASS=M204PASS,	<i>Default user password</i>	X
DLMCHECK/NODLMCHECK	<i>Check DLM=</i>	
RACFGEN TYPE=END		
END		

where:

- *RACFPARM* defines the name of this set of Security Server arguments. Because there can be any number of argument sets in the RACFPARM module, each set must be given a unique name. There is no default name.

Note: If you want to statically link the RACFPARM module, you must include a Security Server argument set named RACFPARM. Otherwise, the following error message is displayed from M204LINK:

```
IEW2454W 9203 SYMBOL RACFPARM UNRESOLVED. NO AUTOCALL
(NCAL) SPECIFIED
```

- *GROUP* specifies the 1- to 8-character Security Server control group name selected by your site. The rules for Model 204 privileges are defined and grouped by this code and stored in Security Server profiles. The group name must match the generic profile high-level qualifier defined to Security Server in the ADDSD statements for the profiles, all of which take the form of *group.keyword.variable.data*.

The default is M204RACF.

You can use the GROUP name to create a separate set of privilege definitions for an individual copy of Model 204. This allows your site to have different privileges for different versions of Model 204, for instance for test and production.

COMGRUP specifies the 1- to 8-character Security Server common group name selected by your site. The rules for the VALIDAT ACCOUNT, PRIORITY, and SUBMIT options are defined and grouped by this code and stored in Security Server profiles. The common group name must match the generic profile high-level qualifier defined to Security Server in the ADDSD statements for the profiles, all of which take the form of *comgroup.keyword.variable.data*.

The default is M204RACF, if a group name is not specified.

Note: COMGRUP is used to create a common set of privilege definitions shared between copies of Model 204.

- *PRTY* specifies the default priority for any user who successfully logs in through Security Server. Options are H (high), S (standard), L (low), or N (none). The default priority is STANDARD if the PRTY keyword is omitted. For more information about priorities, refer to the *Model 204 System Manager's Guide*.
- *VALIDAT* specifies the type of validation to be performed. Multiple validation types can be specified for the interface.

Options are:

- *ACCOUNT* specifies that Security Server validates any account entered by the user during the login process. VALIDAT ACCOUNT verifies that Security Server permits the account value entered by the user. If so, the user is allowed in to Model 204. If not, the login fails.

This validation uses the *comgroup.ACCOUNT.account#* profile.

- *LOGIN* specifies that Security Server rules are tested to determine if a user is allowed to log in to Model 204. If VALIDAT LOGIN is specified, the Security Server Interface determines if the user is permitted to use Model 204 before allowing access. If LOGIN is not specified, all users who pass the user ID/password and optional account validation are allowed access to Model 204.

This validation is performed using the profile group.LOGIN, so it can be used to limit the ability to log in when several copies of Model 204 are running.

- *PRIORITY* verifies which priority the user is permitted to have. This option can be specified in addition to a standard priority. If priority validation fails, the standard priority is the default.

This validation uses the profiles:

comgroup.PRIORITY.HIGH

comgroup.PRIORITY.STANDARD

comgroup.PRIORITY.LOW

comgroup.PRIORITY.NONE

Validation is checked from highest to lowest priority. If a user has PERMIT access to one of these priorities, the value is assigned. If no PERMIT is found and a standard priority is not specified, the priority for the user is set to STANDARD.

- *SUBMIT* uses Security Server rules to determine whether or not a user is allowed to issue the USE command to submit a job through the internal reader. If this option is specified, Security Server checks the user's authorization privileges before allowing the user to submit the job. If this option is not specified, all users who pass the user ID/password and optional account validation are allowed to submit jobs.

This validation uses the *comgroup.SUBMIT* profile.

- *DEFUSER* defines the default submitting user's Security Server user ID if the user is not directly logged in through Security Server. Refer to "Job submission considerations" on page 80 for a description of how the Security Server user ID is used when submitting jobs.
 - If *DEFUSER=*, that is, this field is left blank, then CCASTAT-defined users are not allowed to log into Model 204. Only valid Security Server users can utilize the running system.
 - If *DEFUSER=JOBNAME*, then the following occurs. The ASID ACEE is moved to DEFUSER, the GROUP ACEE is moved to DEFGROUP, and the default user's ACEE pointer is set to AUCKDPTR.
- *DEFUGRP* defines the default Security Server group to which a submitting user is assigned if the user is not directly logged in through Security Server. See "Job submission considerations" on page 80 for a description of how the Security Server group is used when submitting jobs.
- *DEFPASS* defines the default PASSWORD for the default user ID if the user is not directly logged in through Security Server. Refer to "Job submission considerations" on page 80 for a description of how the Security Server password is used when submitting jobs.
- The *DLMCHECK/NODLMCHECK* argument specifies DLM processing options for jobs submitted through the internal reader using the USE command. The DLM parameter on a DD * or DD DATA statement allows a job to be submitted that can itself submit other jobs. This is a potential security compromise. The *DLMCHECK/NODLMCHECK* argument

provides the ability to enforce certain rules when coding the DLM= parameter.

DLM processing options are as follows:

- *DLMCHECK* enforces the rule that if a DLM= *parameter* is used in a JCL stream, it must be the only parameter supplied. In this case, only the following forms of these statements are correct:

```
//DDNAME DD *,DLM=' ; ; '
```

or

```
//DDNAME DD DATA,DLM=' &&&& '
```

In these statements, the DLM value follows the rules described in IBM JCL documentation; any other parameters supplied result in an error. If there is a job statement following the offending statement, Model 204 appends the following to the job statement set as if it is an independent job:

```
USER= . . . , PASSWORD= . . . GROUP= . . .
```

- *NODLMCHECK* checks only the validity of the DLM= *parameter* and not the other optional parameters that can be specified on the JCL statement. All JCL statements after the DLM= *parameter* are sent to the internal reader without being checked. This argument does not guarantee that an error on the statement with the DLM= *parameter* is caught before it is submitted.

DLMCHECK is the default.

The result from assembling the RACFGEN macro is link-edited to a library that is included in a Model 204 link-edited, or can be optionally linked into a separate library and loaded at execution time.

If you link RACFPARM as a separate load module, use the SECRLINK job in the JCL library. Modify the job according to the comments.

If you link RACFPARM with the Model 204 configuration, add the following line in SYSLIN for the link-edit steps of ONLINE, BATCH204, IFAM1, and IFAM4.

```
INCLUDE OBJLIB(RACFPARM)
```

Model 204 link-editing requirements

To support multiple Model 204 logins for different user IDs, Model 204 must be linked to an authorized library using the appropriate Security Server services. This means that any Online configuration must be linked to an authorized library.

BATCH204 or IFAM can be linked to a nonauthorized library, provided that the user logging in is the same user who starts the job and there is not a default login ID that prevents CCASTAT logins. Otherwise, the login fails or a System B7A ABEND occurs.

Defining Model 204 to Security Server

Model 204 allows multiple users per address space and provides services to many users through its own processing, but Security Server is aware only of the main task in the address space (in this instance, Model 204). Model 204 must be defined as having the appropriate Security Server privileges to perform services on behalf of a Model 204 control group.

You can define an additional Security Server control group for nonshared resources as well as for resources shared between multiple copies of Model 204. The first step in the definition process is to choose a name and define a profile for the Model 204 control group and optionally the common group. For example:

```
ADDGROUP M204RACF OWNER(SEcurity) DATA('MODEL 204 CONTROL GROUP')
ADDGROUP M204COM OWNER(SEcurity) DATA('MODEL 204 COMMON CONTROL GROUP')
```

Defining Model 204 users in Security Server

In Security Server, the user profile record contains all information for a defined user of a computer system. There are no special entries or any differences in the way a Security Server user is defined to the system. All authorization services are performed using the generic profiles for privileges (see “Defining corresponding generic profiles” on page 87).

Security Server default user ID

The default user ID limits the authorization of users that have not logged in directly to Security Server (that is, users still defined in CCASTAT).

Each user not logged directly into Security Server is assigned the authorization provided by Security Server for the default user ID. The authorization includes all user resource requests, as described in “Running the Security Server Interface with Model 204” on page 77.

The default user ID automatically is assigned the user ID M204USR with the password M204PASS and belongs to group M204GRP. A user ID of this name must be defined to Security Server for CCASTAT-defined users to be able to log in.

For security reasons, the system manager cannot modify this group, user ID, and password. If different data is required or if the installation would like to change defaults, the parameter module RACFPARM must be modified by the Security Server security administrator or systems programmer.

RACFPARM provides the mechanism to supply overrides to the default group, user ID, and password. This module can be made available during the Model 204 link-edit. Otherwise, Model 204 attempts to load it at the time the interface is initialized. If the module is loaded dynamically, it must be made available in the STEPLIB containing Model 204 or in a system load library. The RACFPARM module is generated with the RACFGEN macro.

Note: All LOADLIBs concatenated in one STEPLIB must be APF-authorized or you lose your APF authorization for that job step.

Setting up the SECTRLOG parameter for trusted login

The SECTRLOG user parameter defines which CRAM thread applications are allowed to log in to Model 204 with a trusted user ID. The CRAM threads for which trusted login applications are allowed are:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

SECTRLOG must be set on the first IODEV line for each trusted CRAM thread, because it is picked up during the initialization of each CRAM channel. The following settings are valid:

Setting	Meaning
X'00'	Trusted Login <i>not</i> allowed (default)
X'01'	CICS applications (CRFS, CRIO, IFAM)
X'02'	TSO applications (CRIO, CRFS)
X'04'	Batch applications (CRIO, IFAM)

Conversion tasks and considerations

Table 4-2 identifies the general tasks that must be completed to install and activate the Security Server Interface. Before conversion, you also might want to review the RACFPARM module definitions and link a separate copy of Model 204 for testing and production.

Table 4-2. Conversion checklist for Security Server

Step	Task
1.	Define a Security Server GROUP control profile for Model 204.
2.	Define a Security Server GROUP common profile for Model 204.
3.	Define Security Server generic profiles for user privileges, and specify the users who have PERMIT access to have those privileges.

Table 4-2. Conversion checklist for Security Server (Continued)

Step	Task
4.	Define Security Server generic profiles for: <ul style="list-style-type: none">• Model 204 login authority related to the control group if the VALIDAT LOGIN option is specified in the RACFPARM• Authorized account code profiles related to the common group if the VALIDAT ACCOUNT option is specified in the RACFPARM• Authorized PRIORITY profiles if the VALIDAT PRIORITY option is specified to assign individual priorities to users and specify which users have PERMIT access for those privileges
5.	Run RACFGEN to assemble the RACFPARM module.
6.	Link RACFOS into Model 204.
7.	Link RACFPARM\$ into Model 204 directly or use SECRLINK to create a separate LOAD module to be used for dynamic linking at initialization.
8.	Add a new parameter in the user zero input stream called SECPLIST= <i>parametername</i> .
9.	Delete Model 204 users from CCASTAT to turn on Security Server authorization checking.

5

CA-Top Secret Interface

In this chapter

- Overview
- Brief introduction to CA-Top Secret
- Running the CA-Top Secret Interface with Model 204
- Using the CA-Top Secret Interface
- Login processing
- Maintaining the CA-Top Secret Interface
- Installing the CA-Top Secret Interface

Overview

This chapter presents an overview of how CA-Top Secret works and describes the impact of the Model 204 CA-Top Secret Interface on the Model 204 user, system manager, and the CA-Top Secret security administrator. Installation requirements and a conversion task summary also are presented.

Note: Throughout this chapter the security administrator or security officer is referred to as the TSS administrator. There are many levels of control for TSS administrators, and the TSS administrator for Model 204 requires having enough scope of control to administer security functions needed to operate the CA-Top Secret Interface.

CA-Top Secret

CA-Top Secret is a security product owned by Computer Associates, Inc. and driven by z/OS. Using the Standard Security Interface (SU32) or the System

Authorization Facility (SAF) found in all versions of z/OS, CA-Top Secret accepts and validates security checking requests issued by the security drivers in major z/OS components. (IMS and CICS use the Standard Security Interface as do other IBM and vendor products.) Because CA-Top Secret can be installed without modifying the operating system, it is compatible with all software that uses the Standard Security Interface.

Model 204 CA-Top Secret Interface

The Model 204 CA-Top Secret Interface provides for an orderly migration from Model 204 security to CA-Top Secret security, and allows an installation to use a combination of either or both security methods. The interface provides the tools and instructions necessary to:

- Define a CA-Top Secret control accessor ID (ACID) for Model 204 to distinguish between separately running copies (that is, test and production)
- Define CA-Top Secret pseudo data sets for CA-Top Secret authorization of Model 204 user privileges within the control ACID
- Define a CA-Top Secret common control ACID for Model 204 resources that can be shared between running copies
- Define an installation default user priority class, or the CA-Top Secret pseudo data set profiles that permit Model 204 priority usage by permitted users
- Provide a mechanism to validate accounting information
- Allow a system manager or installation security officer to add, change, or delete interface options that can control the operation of the interface
- Allow a system manager to view interface control options while the CA-Top Secret Interface is operational
- Provide a mechanism that establishes a shadow login capability to restrict user authorization via a default ACID when users log in through CCASTAT
- Provide a method for a CA-Top Secret security administrator to define a default ACID and password
- Provide a method for a CA-Top Secret security administrator to force users to log in through CA-Top Secret and not through CCASTAT

In summary, the CA-Top Secret Interface provides all the facilities an installation needs to implement the CA-Top Secret security mechanism within the Model 204 environment.

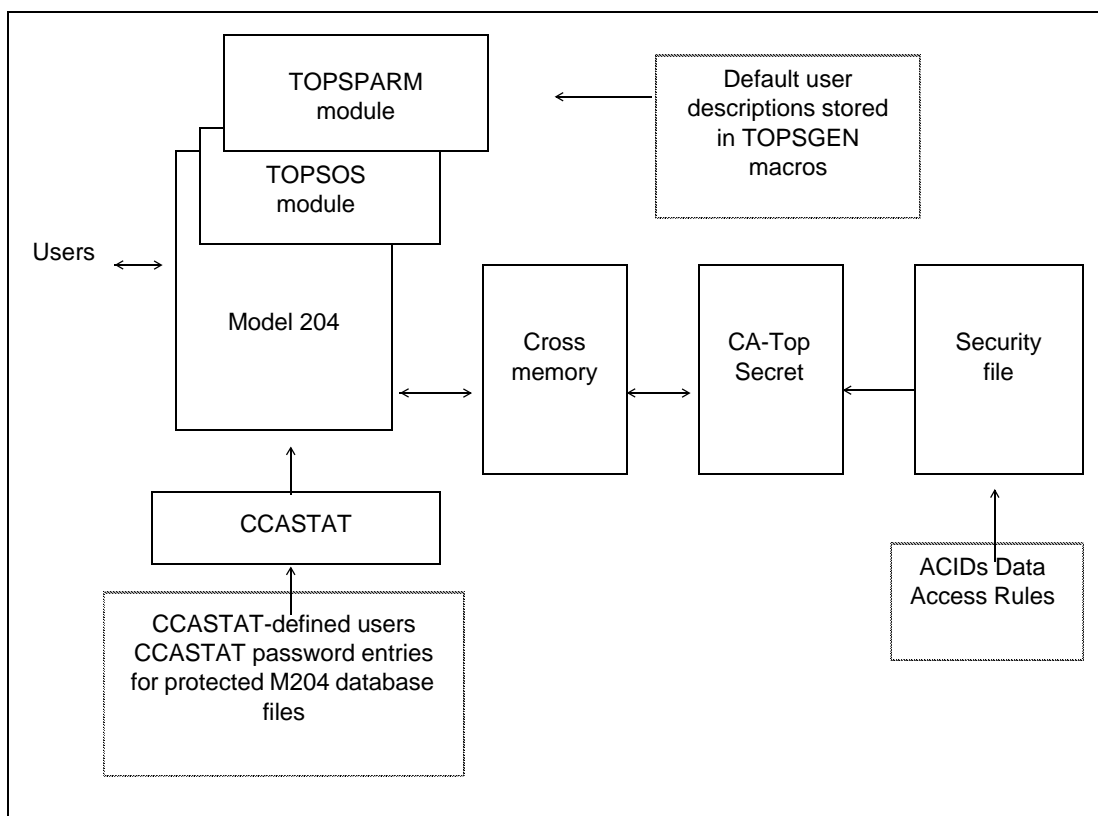
CA-Top Secret components

Figure 5-1 on page 99 illustrates the components of the CA-Top Secret Interface, which consists of:

Model 204	Linked with the CA-Top Secret Interface module
CA-Top Secret Interface module, TOPSOS	Communicates with CA-Top Secret by using the standard IBM Security Server macros
Module TOPSPARM,	Can be linked with Model 204, or can be loaded dynamically at initialization

Each component is described later in this chapter.

Figure 5-1. CA-Top Secret Interface



Brief introduction to CA-Top Secret

This section provides a brief summary of key CA-Top Secret features that are used in the discussion of the Model 204 CA-Top Secret Interface.

For more information about CA-Top Secret, see the documentation associated with that product.

CA-Top Secret processing

CA-Top Secret implements security throughout a data processing community by:

- Identifying users allowed to access the system
- Controlling access to system facilities (for example, BATCH, TSO, Model 204)
- Restricting use of system resources (for example, files, programs)

This extremely flexible system can be tailored to meet the precise security requirements and operational characteristics of an installation.

An overview of the structure and components of CA-Top Secret is in the following sections.

Accessor ID (ACID)

An Accessor ID (ACID) is the fundamental key that CA-Top Secret uses to identify who has access to what resources. A set of resource access authorizations or CA-Top Secret administrative authorizations or both is associated with each unique ACID. This set of authorizations is maintained in one or more Security Records in the Security File whose primary access key is the designated ACID.

An ACID can be a maximum of eight alphanumeric characters. If the user is using TSO, the ACID is limited to seven characters because of TSO restrictions. A user's ACID and user ID must be the same. A user can have multiple ACIDs, one for each facility (TSO, CICS) used.

A user with an ACID is considered defined to CA-Top Secret. Conversely, a user without an ACID is considered undefined by CA-Top Secret.

CA-Top Secret recognizes several different types of ACIDs. For example, each of the various system organizational elements described in "System organizational elements" on page 100 possesses its own ACID type. The type of ACID indicates what administrative authorities it can possess.

System organizational elements

CA-Top Secret architecture is user-oriented, rather than resource-oriented. CA-Top Secret associates a set of specific resource access authorizations with each user, rather than each resource with a set of users who are allowed various types of access to it. As a result of this approach, CA-Top Secret's processing requires only one I/O operation, via a cross-memory call to the CA-Top Secret address space, when a batch job or Online session initiates. Every accessed resource need not be independently checked, which contributes to system efficiency.

CA-Top Secret architecture is constructed from the following basic elements:

- Users
- Profiles
- Departments
- Divisions

The function of the CA-Top Secret organizational elements is to serve as a coordinating framework for security implementation and maintenance. Therefore, a CA-Top Secret security system should be designed to follow the actual organizational structure at an installation.

User element

The *user* element is the lowest level in a CA-Top Secret organizational hierarchy. It associates a specific employee with a specific department. Every user must be associated with a single department. In a broader sense, a user can refer to any discrete entity whose access to resources is defined and specifically restricted. For example, a batch job authorized to read data from only a specific file can be identified to CA-Top Secret as a user. An operating system started task (STC) also can be identified to CA-Top Secret as a user.

A user element is identified to CA-Top Secret by a unique user ACID.

Profile element

When a group of users uses a set of identical resources in the same way (that is, they perform the same job function), it is easier to define the set of access authorizations once and associate the entire set to each of the users in the group. In CA-Top Secret this set of common resource access characteristics is called a *profile*. Every profile is assigned a unique profile ACID.

A defined profile can be associated with any number of users, thereby eliminating the need to define each resource and resource access separately for each user. There is no limit to the number of profiles that can be defined. However, a single user can be associated with no more than 254 profiles.

Every profile must be associated with and defined within a single department.

Department element

At an installation, users typically work for a particular department. CA-Top Secret recognizes this and associates user ACIDs with *departments*. Every department is assigned a unique department ACID. Resources can be assigned to a department just as they can be assigned to a user or a profile.

Every user ACID must be associated with only one departmental ACID.

Division element

CA-Top Secret allows the user to define multiple divisions within the corporate security structure. Each *division* must include one or more departments. However, a department need not be associated with a division. Every division is assigned a unique division ACID, and resources can be assigned to a division just as they can be assigned to a department, profile, or user.

Facilities

A *facility* is any z/OS subsystem (IBM or vendor, provided it is managed by z/OS) that processes work on behalf of users or jobs (for example, Model 204, BATCH, TSO). CA-Top Secret can restrict users to only those specific facilities required to perform their jobs.

Because CA-Top Secret is driven by the Standard z/OS Security Interface, it is compatible with any z/OS subsystem facility or vendor facility that uses the Standard Security Interface including batch processing, STC, TSO, CICS, IMS, NCCF, ROSCOE, WYLBUR, ACEP, TONE, SNA Communications Server (formerly VTAM)/SPF, and COM-LETE. CA-Top Secret checks the set of access authorizations of a particular ACID to determine whether or not it should be allowed to access this facility.

A single ACID can be allowed access to several different facilities, and multiple concurrent logins can be permitted.

Resources

CA-Top Secret protects a great variety of computer resources including data sets (VSAM and non-VSAM), programs, DASD and tape volumes, catalogs, terminals, readers, transactions, and jobs. Even user-defined, nonstandard resources that normally do not activate z/OS security interfaces (such as database fields and account codes) can be defined to CA-Top Secret and protected by them.

A site can choose whether or not to protect a particular class of resources by default (that is, determine whether or not a user must be specifically authorized to gain access to any resource protected in this manner). Resource types that are not protected by default must be explicitly defined for protection. Otherwise, they remain globally accessible to both defined and undefined users.

The techniques for defining resources are discussed in the following sections.

Prefixes

Resources are defined to CA-Top Secret either by their full name or through a generic prefix. Prefixing allows a group of resources of the same type to be defined to CA-Top Secret simultaneously. For example, the program type prefix IEH encompasses IEHPROGM, IEHINIT, IEHLIST, and so on. Using prefixing,

the user ADM10MK can be granted access authorizations to all data sets whose first qualifier is ADM10 with a single CA-Top Secret entry.

Prefixing can be applied to any type of resource.

Data set masking

Another technique for significantly reducing the number of definitions required for CA-Top Secret is data set masking. Masking defines a group of data sets with similar naming characteristics, substituting special characters for the dissimilar characters. For example, the mask *.PROD ignores the high-level index to match it with APC001.PRODLIB.

Access controls

CA-Top Secret controls access to systems and system data in the following four stages:

- Performs the appropriate signon and password validation, and checks the ACID for validity and for whether or not it has been suspended or expired
- Determines whether or not a user is allowed access to the requested facility, forces the user to access the system through a specific source (for example, a particular terminal or to use a specific CPU)
- Checks whether or not the user is authorized to use a requested resource
- Checks exactly how and when the user is authorized to use the resource and whether or not a request is permitted by that authorization

CA-Top Secret allows great specificity in defining what restrictions are placed on the use of a resource by a user or profile group. (Facility and resource restrictions cannot be made at the divisional or departmental levels.) For example, access to a resource can be restricted by expiration date, day of week or time of day, or by facility (for example, this ACID may only access this volume through TSO). For sensitive data sets, users can be restricted to privileged path access (that is, only through a particular program that also may have been forced to load from a specific library).

Access levels

A user either does or does not have access to resources such as terminals. However, other resources such as data sets, volumes, and CICS data (for example, DCT, FCT) can be restricted for specific uses for a specific user. In CA-Top Secret, an access level is defined as the manner in which a resource can be used. CA-Top Secret allows for specificity in defining restrictions on the manner in which a resource can be used by a user or profile group. For example, an ACID might be limited to read, write, update, create, scratch, or fetch access to a particular data set, a combination of these access levels, or granted total access.

Specific resources also can be made globally accessible to all users. Thus, system data sets, libraries, DASD storage, and work volumes that should be made available to all system users can be defined through a few simple CA-Top Secret entries. If access level restrictions are appropriate for one of the shared system resources (for example, SYS1.BROADCAST should be globally accessible, but for system-initiated update access only), such restrictions can be implemented.

Note: Global access restrictions apply to both defined and undefined users.

Ownership and authorization

A particular resource can be owned by only one ACID. In general, ownership of the resource gives full access automatically to that resource. Nonowners must be authorized to access resources and their access levels can be restricted.

When a resource is assigned an owner, it is defined to CA-Top Secret. This is important for resource types that are extended security protection only if they have been defined to the system (for example, programs, terminals, database fields). Any user can access a resource that is not owned.

To simplify security administration, resources should be owned by departments and divisions rather than users. Because ownership of a resource by a department does not imply automatic access to that resource for all users in that department (user ACIDs associated with that department ACID), explicit authorizations are required to restrict access to only the resources a user needs to perform job functions. An exception to this guideline allows users to own all files with data set name prefixes matching their ACID.

Division, department, profile, and user ACIDs can all own resources. Of these ACID types, however, only users and profiles (groups of users) can be authorized to access resources.

The use of profile ACIDs and generic prefixing greatly reduces the number of CA-Top Secret entries required to define an installation's resource owners and resource access authorizations. A group of users performing the same job function usually need to access the same resources. Each of the individual users can be associated with a single profile authorized access to these required resources. If company standardized conventions exist, often these resources possess a family relationship that can be described as a generic prefix (for example, the same highest level qualifier shared among a group of data set names).

Access to facilities is controlled by using authorizations; facilities cannot be owned.

Model 204 user privileges and other authorization items are defined as *pseudo data set resources*. Local installation profiles are written to control who can use those data sets. The Model 204 system manager and the TSS administrator

write these rules, which are described in “Sample definition of Model 204 pseudo data set names” on page 118.

Running the CA-Top Secret Interface with Model 204

When running either as an Online or in batch mode, Model 204 checks CA-Top Secret for the system resources a user can access. Model 204 manages the authorization process, because, in multiuser mode, only Model 204 can identify the user requesting a resource. Model 204 asks CA-Top Secret to validate those requests and responds to a user who has been denied a request for a resource.

Model 204 interfaces with CA-Top Secret in the following distinct areas:

- User logins
 - LOGIN/LOGON command
 - IFSTRT function
 - IFLOG function
- User resource requests
 - Dynamic allocation (ALLOCATE command)
 - Job submission (USE \$JOB command)
 - Sequential data set handling (Record I/O)
 - VSAM data set handling (External I/O)

Note: Because Model 204 databases are not validated on behalf of the user by the security interface, CA-Top Secret must grant the owner of an address space permission to open Model 204 file data sets for update, regardless of whether the owner has write or read-only privileges. This allows Model 204 to write back updates to the FPL (File Parameter List) page, as required by the database management system, regardless of the Model 204 file open privileges.

- User logouts
 - LOGOUT/LOGOFF command
 - IFFNSH/IFDTHRD function

Managing the authorization process

The CA-Top Secret Interface uses the standard IBM Security Server macros to validate user identities and user requests for Model 204 resources. No source code modifications or exits to the CA-Top Secret software are necessary. However, the installation can define additional CA-Top Secret control ACIDs for nonshared resources as well as resources shared among multiple copies of Model 204. An installation *must* define pseudo data sets with appropriate

PERMITs as described in “Sample definition of Model 204 pseudo data set names” on page 118.

Using the CA-Top Secret Interface

A user notices very little change, when running Model 204 with CA-Top Secret. These changes include:

- LOGIN or LOGON
- IFSTRT function
- IFLOG function with IFAM1
- Dynamic allocation
- Job submission
- Data set handling

LOGIN or LOGON command

The LOGIN/LOGON command allows you to gain access to Model 204. After you are connected to Model 204, and if the system manager has set the Model 204 option to require logins, any commands entered by the user (other than LOGIN or LOGON) produces the following message:

```
*** PLEASE LOGIN
```

To log in, enter:

```
LOGIN userid [account]
```

or

```
LOGON userid [account]
```

where:

<i>userid</i>	Is a character string that identifies you
<i>account</i>	Is an optional character string that identifies the account under which you log in

If Model 204 is providing security authorization checks, every Model 204 user has been assigned a user ID, password, and user privileges by the system manager consisting of:

- User ID that identifies the user to Model 204
- Password that provides the user access to the system
- User privileges associated with that user ID and password to define the particular type of access that the user is allowed

- Default priority class assigned to the user ID

When native Model 204 security is in effect, this user ID information is stored in a record in the system file CCASTAT. This record is deleted from CCASTAT when the system manager moves the user into CA-Top Secret security mode. The TOPSPARM module (described in “Preparing a TOPSPARM parameter module with TOPSGEN” on page 121) supplies a default Login ID for CCASTAT Login IDs to do validation requests to CA-Top Secret.

When CA-Top Secret is performing login validation, the user ID (that is, the ACID) is limited to a maximum length of eight characters. The ACID is verified by CA-Top Secret before the user can log in to Model 204.

If the installation chooses to perform CA-Top Secret account validation, any value entered in the account field is verified via CA-Top Secret services. When CA-Top Secret is performing account validation, the account is limited to a maximum length of 8 characters. If no account is entered, the default account becomes the user’s ACID.

For more information about login processing, see page 109.

If you have the appropriate authority, you can change your CA-Top Secret user ID password when you log in to Model 204. For more information, refer to the *Rocket Model 204 Parameter and Command Reference Manual*.

IFSTRT function

In the IFAM1 and IFAM4 environments, the IFSTRT function is used in a Host Language program to allocate a Host Language Interface thread. IFSTRT also establishes the calling convention, performs a user login, and determines whether or not the thread is allowed updating privileges.

When processing the login argument of the IFSTRT call, the user login process follows the rules described in “User 0 login” on page 111.

IFLOG function within IFAM1

The IFLOG function is used only in the IFAM1 environment. It is called following IFSTRT to identify the user when a login is required. This function is necessary in any IFAM1 program where user authorization is validated by CA-Top Secret.

As with the IFSTRT function, the user login process is the same as the process for User 0 login (refer to “User 0 login” on page 111).

Dynamic allocation considerations

Dynamic allocation services in a CA-Top Secret environment are subject to CA-Top Secret system rules for data set validation. For example, data set allocation can be restricted to allow the creation of data sets with certain name patterns. The rules for data set access are determined by the TSS administrator.

To dynamically allocate a *new* data set, the user must be defined to CA-Top Secret as having ALL or CREATE authority. Without this authority, a user issuing the ALLOCATE command receives a Model 204 error message and the ALLOCATE fails.

Allocation of an existing data set to the Online is not validated at allocation time, it is validated when the data set is opened.

If you log in to Model 204 via CCASTAT, your allocation privileges are determined by the CA-Top Secret default ACID.

Note: When you open a sequential or a VSAM data set, either new or existing, CA-Top Secret verifies that you have the appropriate access authority. CA-Top Secret does not perform access authorization on Model 204 data sets, because these data sets are currently protected under Model 204

Job submission considerations

When an Online user issues the USE \$JOB command to submit a batch job, Model 204 identifies the user who submitted the job so that CA-Top Secret can properly determine the authorization for that execution. If the submitter is IFAM1/IFAM4 or User 0, no additional processing occurs. In these cases, the submitted job inherits the privileges that existed when the job doing the submitting was started, because they have already been authorized to run by CA-Top Secret.

When the Model 204 Online is running with an active CA-Top Secret interface, either the user's ACID address or the default user's ACID address is entered into the appropriate location within the JOB statement. Any attempt to circumvent this is denied.

Sequential and VSAM data set considerations

Any attempt to use a sequential or VSAM data set results in an authorization check of:

- CA-Top Secret all, control, or write privileges to open a deferred update data set or to issue the DUMP or USE data set commands
- CA-Top Secret all or create privileges to dynamically allocate a new sequential data set
- CA-Top Secret read, all, or update privileges to read sequential files from User Language or to read the file specified in a RESTORE command
- CA-Top Secret read, all, or update privileges to read external VSAM files from User Language

If a Model 204 user issues a sequential or VSAM data set OPEN command that fails for CA-Top Secret reasons, a Model 204 error message is displayed and the operation fails.

For those users who logged in by being on CCASTAT, authorization is determined by the default user's authorization limits.

Model 204 sequential file data sets (such as CCASTAT, CCAAUDIT, CCAJRNL, or CCAJLOG) are also checked to determine if the owner of the address space has the authority to write to them.

CA-Top Secret directly checks the following Model 204 commands:

- DUMP and DUMPG
- RESTORE and RESTOREG
- USE OUTXXX
- ALLOCATE a new data set
- OPEN DATASET
- OPEN *filename* with deferred update

Login processing

The system manager is responsible for defining security processing in a Model 204 installation in which CA-Top Secret supplies authorization services. This section describes the login processing that the CA-Top Secret Interface performs. AUTHCTL, the Model 204 command used by the system manager to delete and view CA-Top Secret control information, is also described.

Login processing

When a user logs in, Model 204 acquires the user ID and account and searches CCASTAT to determine whether or not the user ID exists. If the user ID is found, Model 204 queries the user for a password and proceeds with authorization processing. If the TOPSPARM module indicates that CCASTAT defined users cannot log in, the login fails.

If the user ID is not found in CCASTAT and CA-Top Secret security is in effect, Model 204 uses CA-Top Secret facilities to authorize the user login. Model 204 passes to CA-Top Secret the login user ID (that is, the ACID), the password and any new password, and finally the terminal ID or user source.

CA-Top Secret makes its own validation checks at this point, which include but are not limited to:

Validation check	Comments
ACID identification	ACID must be defined and must not be suspended or expired
Password checking	Password must be correct for the ACID
New password check	User can be prevented from specifying a new password, or might require a new password

Validation check	Comments
FACILITY checking	Ensures that the user is authorized to use the Model 204 Facility and that the user is allowed to use Model 204 during the specified time
Terminal security	Ensures that the user is allowed to use this terminal. Refer to the section on terminal security for non-SNA Communications Server terminal identification by Model 204
Duplicate signon check	Ensures that each user is logged in only once. This check is independent of the Model 204 duplicate signon check, and its scope is wider. For example, you can ensure that a user cannot log in to Model 204 if they are also logged in to IMS.

Messages

At login, CA-Top Secret can issue messages to users advising them of their status, or why system access was denied. Model 204 passes any messages back to the user via an M204.1500 message.

Note: Model 204 passes back the entire CA-Top Secret error message series following the M204.1500 prefix, which might exceed the terminal line size. Technical Support recommends using the following MSGCTL command in the User 0 input stream to remove the Model 204 message prefix:

```
MSGCTL M204.1500 NOPREFIX
```

After the previous checks, CA-Top Secret indicates whether or not the user can enter into the system, and Model 204 acts accordingly.

- If the login is successful, account processing occurs.
- If the installation has chosen to verify the account, the value entered is checked against the CA-Top Secret profile for comacid.ACCOUNT.account.
- If the value entered is PERMIT READ for that user's ACID, the user is allowed in to Model 204.

After successfully logging in, the user is granted Model 204 privileges of X'00'. Any additional privileges are determined at the time a user command requires a specific privilege.

Users who do not log in directly to CA-Top Secret are automatically restricted in what they can do by the privileges associated with the default ACID. Refer to the discussion of the default ACID in "Defining the Model 204 default user ACID" on page 116.

All system manager commands concerning CCASTAT maintenance functions work as usual. ZBLDTAB also functions as usual when creating the initial CCASTAT.

File, group, and subsystem security functions are defined and utilized as described in the Model 204 documentation.

User 0 login

When the CA-Top Secret Interface is active, User 0 is validated as the owner of the Model 204 run, regardless of whether or not the execution is Online, BATCH204, IFAM1 or IFAM4, because User 0 has a higher ranking than an ordinary user.

Model 204 always attempts to log in User 0 automatically and verifies that any user ID supplied matches the ACID of the owner of the address space. It is not necessary to supply a user ID on the LOGIN command for User 0, because Model 204 determines the owner ACID and supplies it automatically.

If a user ID is supplied on the LOGIN command and the user ID does not exist on CCASTAT, it must match the ACID of the owner of the address space or the login fails. If the user ID is found on CCASTAT and CCASTAT users are allowed to log in, all further authorization checking is based on a default ACID specified by the installation (must be APF-authorized).

Technical Support recommends that no user ID be specified in the login for User 0, so jobs can be easily migrated from test to production without having to change the login command.

Whether or not a user ID is supplied for User 0, no password is necessary and Model 204 does not prompt for one. If a password statement is coded in the input stream, it is treated as an invalid Model 204 command.

If the user ID is on CCASTAT, Model 204 prompts for a password. If the password is correct, the login succeeds but all future data set authorization checking is based on the ACID of the address space or the default ACID.

AUTHCTL command

With the AUTHCTL command, you can view the control information Model 204 needs to interface with CA-Top Secret.

For information about setting control information, see “Preparing a TOPSPARM parameter module with TOPSGEN” on page 121.

The format of the AUTHCTL command is as follows:

Syntax AUTHCTL VIEW TOPSECRET

where:

<i>VIEW</i>	Displays the interface options in effect for the interface currently in operation.
-------------	--

For example, the system manager enters:

AUTHCTL VIEW

The following list is displayed (assuming that the above information was used during initialization):

```
TOP SECRET INTERFACE OPTIONS
ACID          M204TEST  TOPSECRET CONTROL ACID NAME
COMACID       M204COM   TOPSECRET COMMON CONTROL ACID
                           NAME
VALIDATE      ACCOUNT  VALIDATION OPTION IN EFFECT
PRIORITY      LOW      PRIORITY DEFAULT
DLMCHECK      M204USR   USE $JOB DLM CHECKING OPTION
                           DEFAULT ACID
```

The additional line displayed indicates the current default ACID in use. For more information about defaults, refer to “Defining the Model 204 default user ACID” on page 116.

Using trusted login for CRAM users

If your site uses CRAM, you can use the trusted login feature, which allows users to issue login commands or calls that do not include a user ID and password. For a user to log in as trusted, the user ID must be defined to CA-Top Secret. User IDs defined to CCASTAT are not allowed to log in as trusted users. CICS users must be using the CA-Top Secret interface for CICS. Only CICS user IDs that log in through CA-ACF2 can log in as trusted users.

Trusted login can be used with the following CRAM thread IODEV types:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

To set up trusted login for a user, set the SECRTLOG user parameter, as discussed in “Setting the SECTRLOG parameter for trusted login” on page 124.

Login processing for trusted login

For users connecting with a CRAM thread that allows trusted login, the user login processing routines are changed to handle a LOGIN command or IFSTRT call without a user ID and password:

- CRFS and CRIO channel threads handle User Language statements. These users ordinarily issue a LOGIN or LOGON request with the following format:

```
LOGIN userid [account];password [:new password];apsyn-  
ame
```

For trusted logins, the format is:

```
LOGIN;;apsynname
```

- The IFAM channel threads handle IFAM2 statements. These users ordinarily issue an IFSTRT or IFSTRTN call with the following format:

```
IFSTRT (RETCODE, LANG_IND, LOGIN, THRD_TYP, THRD_ID)
```

```
IFSTRTN (RETCODE, LANG_IND, LOGIN, THRD_TYP, THRD_ID, CHAN)
```

The LOGIN parameter is required and supplies the user ID and password as a character string using the following format:

```
`userid [account];password [:new password];`
```

For trusted logins, the statement format is the same but the login character string is a semicolon surrounded by single quotes (';').

Trusted login errors

When using trusted login, you might receive one of the following errors:

- Message below is generated when either:
 - a. Model 204 job requesting the trusted login feature is running without a Model 204 Security Interface (CA-ACF2, Security Server, or CA-Top Secret).
 - b. Model 204 address space does not have enough storage to allocate an internal work area for the trusted login feature. In this situation, the Model204 job is not initialized, because it still has to allocate storage for the Model 204 file buffers.

```
M204.2378: SECURITY TRUSTED LOGIN FEATURE DISABLED
```

- The following message is issued when the trusted user ID passed by CRAM is not between one and eight bytes long:

```
M204.2379: INVALID TRUSTED USERID LENGTH = length.
```

Maintaining the CA-Top Secret Interface

This section discusses the user site requirements that must be met in order to activate the CA-Top Secret Interface.

At each CA-Top Secret site, a TSS administrator performs system-wide maintenance functions within CA-Top Secret. A TSS administrator is

responsible for defining and maintaining the CA-Top Secret pseudo data sets and PERMITs for authorization that are eventually used by Model 204. The following discussions explain the Model 204 data that TSS administrators maintain within their scope of control.

All the CA-Top Secret specific functions are administered by the TSS administrator using the TSS command. The TSS command is described fully in the CA-Top Secret documentation.

Defining Model 204 to CA-Top Secret

A Model 204 Online is defined to CA-Top Secret using the Facilities Matrix, because the Online must run as a multiuser address space. Model 204 can be initiated as a started task or as a batch job, depending on local security requirements.

The definition of Model 204 can be done through Online TSS services or via a batch job. The following sample batch execution can be used to define Model 204 (assuming the USER1 entry in the Facilities Matrix is available). The sample entries related to multiuser address spaces are required; however, the other sample options are merely recommended. Different options, as described in the CA-Top Secret documentation, might apply depending on local security rules.

```
//JOBNAME JOB (other information).....
//*-----*
//*          ADD THE MODEL 204 FACILITY TO TOP SECRET.      *
//*-----*
//BATHTSO   EXEC   PGM=IKJEFT01,REGION=300K
//SYSPRINT  DD     SYSOUT=A
//SYSTSPRT  DD     SYSOUT=A
//SYSTSIN   DD     *
    TSS MODIFY('FAC(USER1=NAME=MODEL204)')
    TSS MODIFY('FAC(MODEL204=NOABEND,ASUBM,AUTHINIT)')
    TSS MODIFY('FAC(MODEL204=ID=M)')
    TSS MODIFY('FAC(MODEL204=NORNDPW)')
    TSS MODIFY('FAC(MODEL204=ACTIVE,LCFCMD,LUMSG)')
    TSS MODIFY('FAC(MODEL204=STMSG,MULTIUSER,PGM=ONL)')
    TSS MODIFY('FAC(MODEL204=RES,SIGN(M),SHRPRF,NOTSOC)')
    TSS MODIFY('FAC(MODEL204=WARNPW,XDEF)')
    TSS MODIFY('FAC(MODEL204=DEFACID(M204USR))')
//
```

Note the following information regarding the above entries:

1. *USER1* entry can be changed to any open Facility Matrix entry available at the installation. The name specified for the *NAME=* entry can be changed to any name if the ACID used to define Model 204 uses this same name as its Master Facility.
2. *NOABEND* and *AUTHINIT* are required entries to prevent abends due to

user errors and security abends because Model 204 issues RACINIT commands. The ASUBM option is used if job submission is allowed.

3. *ID=M* entry is variable. It is used in CA-Top Secret reporting as a one-character facility identifier.
4. *MULTIUSER* entry is required to support Online users. *PGM=ONL* is a required entry to authorize a Model 204 use of security commands, and identifies the first three characters of the actual program name.
5. *DEFACID(M204USR)* entry defines the ACID to be assigned to users that cannot be defined. This entry is not normally used, but might be in the future, so this should correspond to the Model 204 default ACID that is defined in TOPSPARM.

Defining the Model 204 ACID

Because facilities cannot own other objects, an ACID must be defined to be used as the main user when starting the facility. The name of this ACID is irrelevant to Model 204, but it might be useful to use the same name as the control ACID or common control ACID used for the pseudo data set names that Model 204 uses.

For example, the following TSS command can be used from a TSO terminal to define the ACID M204TOPS:

```
TSS CREATE (M204TOPS) TYPE (USER) FAC (STC,BATCH) PASS (NOPW)
      DEPT (department) NOSUBCHK MASTFAC (MODEL204)
      NAME ('MODEL 204 ACID')
```

where the following options are the minimum required to be assigned to this ACID:

- *CREATE(M204TOPS)* identifies the ACID to be created.
- *TYPE(USER)* specifies that this is a user type ACID.
- *FAC(STC,BATCH)* indicates Model 204 can start as a batch job or started task.
- *PASS(NOPW)* specifies there is no password associated with this ACID. This should be changed in a secured environment.
- *DEPT(department)* specifies the department that owns and is responsible for this ACID.
- *NOSUBCHK* allows this ACID to bypass job submission checking, allowing this user to submit jobs. The submitted jobs must contain the *USER=* and *PASSWORD=* parameters.
- *MASTFAC(MODEL204)* indicates that this ACID's master facility is MODEL204 (assuming MODEL204 was the name assigned to the facility).

- *NAME('MODEL 204 ACID')* simply assigns a name to the ACID.

This entry can be duplicated with a different ACID name if a common ACID is defined for pseudo data set name ownership that is shared between several copies of Model 204.

Defining the Model 204 default user ACID

Define a default ACID for Model 204. This ACID is named in the TOPSGEN MACRO, and is used to limit the authorizations for Model 204 users that log in while still on CCASTAT. The ACID probably has minimal authority if used at all.

For example, the following TSS command can be used from a TSO terminal to define the ACID M204USR (that is, the default used by Model 204):

```
TSS  CREATE(M204USR)  TYPE(USER)  FAC(MODEL204)
      DEPT(department)  PASSWORD(M204PASS,0)
      NOPWCHG NAME('M204 DEFAULT USER')
```

where:

- *CREATE(M204USR)* names the ACID to be created. This name must match the name in the TOPSPARM parameter module or must be setup as M204USR, which is the default. Model 204 attempts to log in this user during initialization, and if unsuccessful, users on CCASTAT are denied login privileges.
- *TYPE(USER)* means that this is a user type ACID.
- *FAC(MODEL204)* indicates that this user can use the Model 204 facility.
- *PASSWORD(M204PASS,0)* establishes the password for this user (the default or it must match the DEFPASS entry in TOPSPARM). The 0 entry indicates that this password never expires.
- *DEPT(department)* names the department that owns and is responsible for this ACID.
- *NOPWCHG* indicates that this user is not allowed to change the password. This prevents a user from logging in to Model 204 and accidentally changing the password, which would then cause a login failure for M204USR the next time Model 204 is initialized.
- *NAME('M204 DEFAULT USER')* assigns a name to the ACID.

Allowing users to log in to Model 204

The following TSS command is required to allow a user access to the Model 204 facility:

```
TSS PERMIT(acid)  FAC(MODEL204)
```


where:

FAC(MODEL204)	Identifies the facility name under which Model 204 was set up.
---------------	--

Defining user privileges

The privilege names that Model 204 uses are based on pseudo data set profiles. The CA-Top Secret CREATE and ADD commands are used just as if a data set were being defined, but it really describes a resource profile for Model 204. This method provides the simplest way to describe resources.

In the following examples, the simplest set of conditions is described. The system manager can use all CA-Top Secret options (for example, auditing when a resource is used, changing the '*ALL*' authority, or naming a specific owner of the pseudo data sets). The system manager also uses security access levels for both the profiles and the PERMITs or REVOKEs for users of those profiles.

The interface requires that users authorized for pseudo data set resources are permitted read access.

One of the tasks that the TSS administrator must perform is to identify user privilege names as CA-Top Secret pseudo data set names.

The user privilege names listed in Figure 5-Table 5-1. define the Model 204 privilege types for a user. These names are further qualified by the control ACID name entered in TOPSPARM CSECT.

Table 5-1. User privilege names and privileges

This user privilege...	Defines a user...	Corresponding LOGCTL setting
'acid.PRIV.SUPER .USER'	With superuser privileges. A user with read access to this profile can create a file with the Model 204 CREATE command.	X'80"
'acid.PRIV.SYSTEM .ADMIN'	With Model 204 system administrator privileges. A user with READ access to this profile can issue commands such as LOGWHO, MONITOR, PRIORITY, and WARN.	X'40'
'acid.PRIV.CHANGE .FILE.PASSWORD'	Who can change the file password that is used to open a file. This privilege is valid only if the file being opened is secured through Model 204 facilities (that is, the file entry is stored in CCASTAT).	X'20'
'acid.PRIV.SYSTEM .MANAGER'	With system manager privileges. A user with read access to this profile can issue all system administrator commands along with certain other privileged commands (for example, LOGCTL, AUTHCTL, DUMPG).	X'08'
'acid.PRIV.OVERRIDE .RECORD.SECURITY'	Who can override record security. A user with read access to this profile can do so.	X'04'

Sample definition of Model 204 pseudo data set names

CA-Top Secret pseudo data set names, which correspond to the defined privileges, must be defined to CA-Top Secret. This step is necessary, because an ACID must be assigned ownership of the pseudo data sets before permission can be granted to use them.

The CA-Top Secret statements used to define the standard Model 204 privileges and other resources to CA-Top Secret in a batch TSO execution are shown in the following example, which assumes M204TOPS as both the control and common control ACIDS:

```
//JOBNAME      JOB      (other information)
//DOIT         EXEC     PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT     DD      SYSOUT=A
//SYSTSIN      DD      *
TSS ADD(M204TOPS) DSN('acid.PRIV.SUPER.USER')
TSS ADD(M204TOPS) DSN('acid.PRIV.SYSTEM.ADMIN')
TSS ADD(M204TOPS) DSN('acid.PRIV.CHANGE.FILE.PASSWORD')
TSS ADD(M204TOPS) DSN('acid.PRIV.SYSTEM.MANAGER')
TSS ADD(M204TOPS) DSN('acid.PRIV.OVERRIDE.RECORD.SECURITY')
TSS ADD(M204TOPS) DSN('comacid.PRIORITY.HIGH')
TSS ADD(M204TOPS) DSN('comacid.PRIORITY.STANDARD')
TSS ADD(M204TOPS) DSN('comacid.PRIORITY.LOW')
TSS ADD(M204TOPS) DSN('comacid.ACCOUNT.nnnnnnnnn')
```

where:

<i>nnnnnnnn</i>	Represents an account number; there are as many as required.
-----------------	--

The DSN prefix is irrelevant, but probably is the same as the control ACID for simplicity's sake.

Next permission to have these privileges would be granted to specific ACIDs or groups of ACIDs via profiles. The following sample TSS PERMIT gives a user Model 204 system manager authority:

```
TSS PER(useracid) DSN('acid.PRIV.SYSTEM.MANAGER')
ACCESS(READ)
```

where:

<i>acid.PRIV.privilege</i>	Specifies one of the fixed types that Model 204 uses to build CA-Top Secret search keys for the rules. This is the CA-Top Secret control ACID defined to Model 204 by the AUTHCTL A TOPSECRET ACID command or provided as an override in TOPSPARM.
----------------------------	--

Another sample TSS PERMIT assigns a user all Model 204 privileges:

```
TSS PER(useracid) DSN('acid.PRIV.-') ACCESS(READ)
```

This CA-Top Secret PERMIT statement permits the specified user ACIDs to access the named resources. Rules for determining the authorized user are described in the CA-Top Secret documentation.

Privilege rules are tested whenever a user issues a command that requires a specific privilege. If the user is authorized to have that privilege, the command succeeds. If not, the user typically receives a Model 204 error message and the attempt might be logged as a CA-Top Secret violation.

CA-Top Secret default ACID

The default ACID limits the authorization of users that have not logged in to CA-Top Secret directly (that is, users still defined in CCASTAT). Each user not logged directly in to CA-Top Secret is assigned the authorization provided by CA-Top Secret for the default ACID. The authorization includes all data set access authorization as well as job submission, and so on.

The default ACID automatically is assigned the ACID M204USR with the password M204PASS. An ACID of this name must be defined to CA-Top Secret for CCASTAT-defined users to be able to log in.

For security purposes, the system manager cannot modify this ACID and password. If different data is required or if the installation would like to change defaults, the parameter module TOPSPARM must be modified by the TSS administrator or systems programmer or both.

TOPSPARM allows you to override the default ACID and password. This module can be made available during the Model 204 link-edit. Otherwise, Model 204 loads it when the interface is initialized. If the module is loaded, it must be made available in the STEPLIB containing Model 204 or in a system load library (be sure it is APF-authorized so that it does not cause the Model 204 LOAD module to lose its APF authorization). The TOPSPARM module is generated with the TOPSGEN macro.

Terminal security considerations

CA-Top Secret can enforce terminal security when a user logs in. At that time, the source of the login is passed to CA-Top Secret. For systems running Model 204 with the SNA Communications Server Interface, this poses no problem, because Model 204 recognizes the terminal name and stores it for use.

Configurations using CRAM (TSFS, CICS, IFDIAL) must use the CRAM channel name instead of the terminal ID as the user source, because no secured mechanism in CRAM identifies the source of the user. Because CRAM channel names can be identified as valid sources, the ACIDs allowed to use those sources can be validated. Using the CRAM channel name temporarily ensures that entry to Model 204 is from an approved source.

Installing the CA-Top Secret Interface

CA-Top Secret can run in the following modes:

This mode...	Indicates that CA-TOP Secret is...
DORM	Installed but not actively validating activity.
WARN	Active but violations do not result in a failed request. WARN mode provides TSS administrators with the ability to tune definitions without affecting current activity. Violation messages are usually sent to users in this mode.
IMPL	Active and fails any unauthorized requests from defined ACIDs. Users not defined to CA-Top Secret execute normally, unless they access a protected resource.
FAIL	In full control of access requests. All users are defined and all resources are protected.

These modes can apply on a system-wide basis or to individual ACIDS. Model 204 works in all modes of operation.

Unless the ACIDs or the system are in FAIL or IMPL mode, access to Model 204 cannot be controlled, and access checks for data sets are ineffective. However, any WARN mode messages are issued to the user.

Unless ACIDs using Model 204 operate in FAIL or IMPL mode, users can log in and receive very high Model 204 privileges. Because privilege checks consist of asking CA-Top Secret if the user has read access to a pseudo data set name, CA-Top Secret indicates that access is authorized, and the user receives all privileges.

Decrypting the CA-Top Secret Interface

As part of INS204, the M204DECR job is generated with all the steps needed to decrypt CA-Top Secret. For more information, refer to the *Model 204 Installation Guide for IBM z/OS*.

SECPLIST parameter

The SECPLIST User 0 parameter in CCAIN allows you to specify the name of the TOPSGEN argument set with which to initialize the interface. The name is defined by the assembler label name of the TOPSGEN macro.

If the SECPLIST parameter is not in CCAIN, TOPSPARM is used as the default name of the argument set. If no match is found for this name and the name in the TOPSPARM module, the interface is initialized using the default parameters precoded in the CA-Top Secret Interface. In this case, Account and Priority validation are not performed.

The following TOPSPARM module contains two sets of CA-Top Secret arguments. In the first set, the name is LOG1 and account security is in effect. In the second set, the name is LOG2 and both account and priority security are in effect.

```

                TITLE 'TOPSPARM MODULE'
LOG1            TOPSGEN TYPE=CSECT,                      X
                ACID=M204PROD,                          X
                COMACID=M204TOPS,                        X
                PRTY=H,                                  X
                VALIDAT=ACCOUNT,                         X
                DEFACID=,                                X
                DEFPASS=,                                X
LOG2            TOPSGEN TYPE=CSECT,                      X
                ACID=M204TEST,                          X
                COMACID=M204TOPS,                        X
                PRTY=S,                                  X
                VALIDAT= (ACCOUNT, PRIORITY) ,           X
                DEFACID=M204ACID,                       X
                DEFPASS=DEFPASS,                        X
                TOPSGEN TYPE=END
                END

```

Preparing a TOPSPARM parameter module with TOPSGEN

The TOPSGEN macro of the TOPSPARM module allows the system manager, TSS administrator, or system programmer to generate a set of arguments that govern login and other security processes. There can be multiple argument sets in a TOPSPARM module. The TOPSPARM parameter module can be linked with Model 204 or dynamically loaded when the CA-Top Secret Interface is initialized. Dynamic loading allows for more flexibility in making TOPSGEN changes, but you must remember that it is APF-authorized.

Note: AUTHCTL might be deleted from the CA-Top Secret Interface in a future release and all control information will be generated by TOPSGEN.

The following is a sample TOPSGEN macro:

```

TITLE 'GENERATE A TOP SECRET PARAMETER MODULE'

NAME TOPSGEN ACID='M204TOPS' , Control ACIDX
COMACID='M204TOPS' , Common control ACIDX
PRTY=S, PriorityX
VALIDAT=, Validation itemsX
DEFACID=, Default ACIDX
DEFPASS='M204PASS' , Default ACID passwordX
DLMCHECK/NODLMCHECK DLM= checkingX
TOPSGEN TYPE=END
END

```

where:

- *NAME* defines the name of this set of CA-Top Secret arguments. Because there can be any number of argument sets in the TOPSPARM module, each set must be given a unique name. There is no default name, but Technical Support recommends that you name one TOPSPARM for jobs that do not specify a SECPLIST value in CCAIN.
- *ACID* option specifies the 1- to 8-character CA-Top Secret control ACID name selected by the installation. The rules for Model 204 privileges are defined and owned by this ACID and stored in CA-Top Secret. The ACID name must match the pseudo data set profile high-level qualifier defined to CA-Top Secret in the CREATE statements for the pseudo data sets, all of which take the form of *acid.keyword.variable.data*.

The default is M204TOPS.

This control ACID is used to create a separate set of privilege definitions for an individual copy of Model 204, allowing an installation to have differing privileges for a test and production version.

- *COMACID* option specifies the 1- to 8-character common ACID name selected by the installation. The rules for the VALIDATE ACCOUNT and PRIORITY options are defined and grouped by this ACID and stored in CA-Top Secret. The common ACID name must match the pseudo data set high-level qualifier defined to CA-Top Secret in the CREATE statements for the pseudo data sets, all of which take the form of *comacid.keyword.variable.data*.

The default is M204TOPS, if an ACID name is not specified.

Note: This option is used to create a common set of privilege definitions shared between copies of Model 204.

- *PRTY* argument specifies a default priority for any user who successfully logs in through CA-Top Secret. Options are H (high), S (standard), L (low), or N (none). The default priority is S, if the PRIORITY keyword is not entered. For more information about the PRIORITY parameter, refer to the *Model 204 System Manager's Guide*.
- *VALIDAT* argument specifies validation options for execution at login time. Any operands supplied for this macro must be enclosed in parentheses and separated by commas. Multiple validation types can be specified for the interface.

VALIDAT argument options are as follows:

- *ACCOUNT* option specifies that any account entered by the user during the login process is validated by CA-Top Secret. The VALIDAT ACCOUNT option verifies with CA-Top Secret that the account value entered by the user is permitted. If so, the user is allowed in to Model 204. If not, the login fails.

This validation uses the *comacid.ACCOUNT.account#* pseudo data set

- *PRIORITY* option verifies what priority the user has. This option can be specified in addition to a standard priority. If priority validation fails, the standard priority is the default.

This validation uses the following pseudo data sets:

comacid.PRIORITY.HIGH

comacid.PRIORITY.STANDARD

comacid.PRIORITY.LOW

comacid.PRIORITY.NONE

Validation is checked from highest to lowest priority. If a user is permitted read access to one of these priorities, the value is assigned. If no PERMIT is found and a standard priority is not specified, the user's priority is set to standard.

- *DEFACID* argument defines the default ACID if the user is not directly logged in through CA-Top Secret. Refer to the discussion on CCASTAT user logins for a description of how the CA-Top Secret ACID is used.
- *DEFPASS* argument defines the default PASSWORD for the default ACID if the user is not directly logged in through CA-Top Secret.
- *DLMCHECK/NODLMCHECK* argument specifies DLM processing options for jobs submitted through the internal reader using the USE \$JOB command. The DLM parameter on a DD * or DD DATA statement allows a job to be submitted that can itself submit other jobs. This is a potential security compromise. The DLMCHECK/NODLMCHECK argument allows you to enforce certain rules when coding the DLM= parameter.

DLM processing options are as follows:

- *DLMCHECK* enforces the rule that if a DLM= parameter is used in a JCL stream, it must be the only parameter supplied. In this case, only the following forms of these statements are correct:

```
//DDNAME DD *,DLM=' ; ; '
```

or:

```
//DDNAME DD DATA,DLM=' &&&& '
```

In these statements, the DLM value follows the rules described in IBM JCL publications; any other parameters supplied result in an error. If there is a job statement following the offending statement, Model 204 appends USER=...,PASSWORD= to the job statement set as if it is an independent job.

- *NODLMCHECK* checks only the validity of the DLM= parameter and not the other optional parameters that can be specified on the JCL statement. All JCL statements after the DLM= parameter are sent to the internal reader without being checked. This argument does not guarantee that an error on the statement with the DLM= parameter is caught before submission.

DLMCHECK is the default.

The result of the TOPSGEN macro assembly is link-edited to a library that is included in a Model 204 link-edit, or it can be optionally linked into a separate library and loaded at execution time.

If you link TOPSPARM as a separate load module, use the SECRLINK job in the JCL library. Modify the job according to the comments.

If you link TOPSPARM with the Model 204 configuration, add the following line in SYSLIN for the link-edit steps of ONLINE, BATCH204, IFAM1, and IFAM4.

```
INCLUDE OBJLIB(TOPSPARM)
```

Model 204 link-editing requirements

To support multiple Model 204 logins for different ACIDs, Model 204 must be linked to an authorized library using the appropriate link-edit services. This means that any Online multiuser configuration must be linked to an authorized library.

BATCH204 or IFAM can be linked to a nonauthorized library, provided that the user logging in is the same user who starts the job or that the user ID exists on CCASTAT. Otherwise, the login fails.

Setting the SECTRLOG parameter for trusted login

The SECTRLOG user parameter defines which CRAM thread applications are allowed to log in to Model 204 with a trusted user ID. The CRAM threads for which trusted login applications are allowed are:

- IODEV11 (CRFSCHNL)
- IODEV29 (CRIOCHNL)
- IODEV23 (IFAMCHNL)

SECTRLOG must be set on the first IODEV line for each trusted CRAM thread since it is picked up during the initialization of each CRAM channel. The following settings are valid:

Setting	Meaning
X'00'	Trusted Login <i>not</i> allowed (default)
X'01'	CICS applications (CRFS, CRIO, IFAM)
X'02'	TSO applications (CRIO, CRFS)
X'04'	Batch applications (CRIO, IFAM)

Conversion tasks and considerations

The following list identifies the general tasks that must be completed to install and activate the CA-Top Secret Interface.

Table 5-2. Conversion checklist for CA-Top Secret

Step	Task
1.	Define Model 204 as a CA-Top Secret facility.
2.	Define the Model 204 control ACID (and common ACID if used).
3.	Define the Model 204 default user ACID if you allow CCASTAT IDs to log in.
4.	Define the Model 204 user privilege pseudo data sets.
5.	Permit users to use the Model 204 facility.
6.	Permit users to have appropriate Model 204 privileges.
7.	Use TOPSGEN to create a TOPSPARM parameter module.
8.	Link-edit TOPSPARM with SECRLINK for dynamic link with Model 204 during initialization. or Link-edit Model 204 with TOPSPARM.
9.	ADD the SECPLIST parameter in user zero input in CCAIN.
10.	Move the Model 204 facility to WARN mode for testing.
11.	Move the Model 204 facility to FAIL mode for execution.

Note: If other Rocket Software products are to be installed, it is essential that the default ACID be able to log in and have the appropriate access privileges to data sets that might be read or written during installation. For example, if the default user cannot log in, all installation JCL that is supplied must be modified to provide valid login statements for valid CA-Top Secret ACIDs.

6

SQL Security Exits

In this chapter

- Overview
- DDLPRIV exit
- DMLPRIV exit
- Return codes for DDLPRIV and DMLPRIV security exits
- Installing the SQL security exits

Overview

The SQL security exit feature allows you to replace the security provided by the Model 204 SQL catalog (CCACAT) with the security of your external package to control the use of SQL data definition language (DDL) and access to SQL objects via data manipulation language (DML) statements. Instead of using CCACAT to check privileges for DDL usage or SQL object access, you can write external security exits, called DDLPRIV and DMLPRIV, that are called every time a user tries to access the Model 204 SQL catalog.

The DDLPRIV and DMLPRIV SQL security exits are called only if an external security package is specified as being active for SQL. Otherwise, standard Model 204 SQL privilege checking is performed; Model 204 never performs both types of checking.

For information about the DDLPRIV exit, see “DDLPRIV exit” on page 130. For information about DMLPRIV, see “DMLPRIV exit” on page 132.

Requirements

To use the SQL security exit feature, your site must use the Model 204 SQL product, Connect★, and one of the external security packages discussed earlier in this manual.

Security exit functions

The SQL security exit feature is designed to:

- Check access privileges down to the column level.
- Recognize all the standard types of SQL access (SELECT, INSERT, DELETE, UPDATE).
- Control a user's access to all DDL statements via the security package already in place at your site.

If access to a DDL or DML statement is rejected by the security exit, Model 204 terminates the request (the processing of an SQL statement) but not the user's session.

Your role as security officer

As the security officer for your site, you can write two SQL security exits: one for DDL statements, DDLPRIV and one for DML statements, DMLPRIV. Within the security exits need to:

- Determine what rules to use for your site.
- Set up any necessary parameters for the appropriate validation request to the external security package.
- Return back to the Model 204 SQL Server the results of the security validation request so that Model 204 can either allow processing of the user's Model 204 request or terminate the request with an error message.

SQL security exit restrictions

SQL security exits *must* be written in IBM 370 Assembly language and use Model 204 register conventions and the Model 204 Assembly macros for subroutine entry and exit.

FIPS compliance

If your site is using a security exit, Model 204 SQL does not comply with Federal Information Processing Standard (FIPS) 127-1 (level 2 of ANSI X3.135-1989) for SQL. However, if you are not using a security exit, Model 204 SQL continues to comply with the standard.

Register conventions for IBM and Model 204

IBM and Model 204 both use the register conventions listed in Table 6-1, which you might need to know when writing security exits.

Table 6-1. Register conventions for IBM and Model 204

IBM register (REG) name	Model 204 register name
0	X0
1	X1
2	T1
3	T2
4	T3
5	T4
6	T5
7	RC
8	R6
9	R5
10 (X'a')	R4
11 (X'b')	R3
12 (X'c')	R2
13 (X'd')	R1
14 (X'e')	PD
15 (X'f')	RL

When writing a security exit, do not use the CSAVE and CRSTR macros, which save and restore (IBM) Registers 13, 14, and 15. The security exits use IBM Register 13 as a work register to pass the return code back to the caller.

Using the CDTB module

When you write SQL security exit routines, store them in the Model 204 CDTB module. If your site chooses to use an external security manager (Security Server (formerly RACF), CA-Top Secret, or CA-ACF2) and SQL security exits to replace Model 204's SQL security, you are responsible for managing this module and maintaining its contents. Information about managing the CDTB module is available in the *Model 204 Installation Guide* for your operating system.

Within the CDTB module, Model 204 supplies the dummy SQL security exit routines DDLPRIV and DMLPRIV. The combination of these two routines provides a model for the SQL security exits. That is, Model 204 passes all

necessary information to the SQL security exit so that your exit can issue the appropriate validation requests to the external security manager (ESM).

The dummy routines contained in the module serve the following purposes:

- Prevent unresolved references when the Model 204 Online is linked (you can also avoid unresolved references by defining the routine names as weak external symbols using the WXTRN instruction).
- Include the Assembler code required to accept the necessary parameter values passed by Model 204 SQL and return an appropriate return code to Model 204 based on the validation request of the external security manager.

Note: The dummy exits supplied by Model 204 are written with a return code of 9, which indicates that the exits are not active. For more information about the available return codes, see “Return codes for DDLPRIV and DMLPRIV security exits” on page 134.

- Help to illustrate the use of Model 204 register conventions and the ENTER and EXIT macros used in Model 204 subroutines.

Even if your site does not use an external security interface, you need to link CDTB into your Model 204 Online to avoid unresolved references.

DDLPRIV exit

Currently Model 204's SQL processor checks a user's privileges to access SQL objects when compiling an SQL query for that user. To check privileges, the compiler calls a dictionary routine that has access to privilege data in CCACAT. You can instead use the DDLPRIV exit to check DDL statements before processing.

One kind of privilege checking used in DDL processing is to make sure that the user has Model 204 system manager privileges and that the user's Model 204 user ID (called the authorization ID in SQL) matches the user ID associated with the SQL object that the DDL statement is acting upon.

When checking system manager's privileges for, for example, CA-Top Secret, the user ID corresponds to an accessor ID (ACID), and the DDL statement corresponds to a nonstandard resource. The SQL security exit does not check for Model 204 system manager privileges as such, but checks for the set of privileges that you create, corresponding to a system manager's role.

In a different example, a user named JOAN creates a schema and another user, ADRIAN, attempts to drop it. ADRIAN is not be allowed to do so, because the user ID associated with the schema (JOAN) does not match the user ID of the user trying to drop the schema (ADRIAN).

Security for DDL statements

All DDL statements in Model 204's SQL processing are protected by some level of security. The DDL statements supported in Connect★ are:

- GRANT
- REVOKE
- CREATE SCHEMA
- CREATE VIEW
- CREATE TABLE
- DROP SCHEMA
- DROP VIEW
- DROP TABLE
- ALTER TABLE
- SET SCHEMA
- SET USER

Parameters passed to SQL

The parameters passed to DDLPRIV from Model 204 are:

DDLPRIV *userid*, *authid*, *ddl_statement*, PRIVOUT

where:

<i>userid</i>	Is the address of a character string containing the user's Model 204 user ID. The string is terminated with a binary zero. The user ID parameter is passed in T1.
<i>authid</i>	Is the address of a character string containing the Model 204 user ID associated with the SQL object being operated on by the DDL statement. The string is terminated with a binary zero. The authid parameter is passed in T2.
<i>ddl_statement</i>	Is an integer code describing the type of DDL statement. The <i>ddl_statement</i> parameter is passed in T3. The codes are listed in Table 6-2.
PRIVOUT	Is the return code from the call passed back to Model 204 in R1. The return codes are described on page 134.

Table 6-2. Statement codes and DDL statement types

Statement code	DDL statement type
40	CREATE SCHEMA
41	CREATE VIEW
42	CREATE TABLE
53	SET USER
54	SET SCHEMA
55	GRANT
56	REVOKE
57	DROP SCHEMA
58	DROP TABLE
59	DROP VIEW
81	ALTER TABLE

Disallowing access to SQL statements

If your site does not, for example, allow users access to GRANT and REVOKE statements, DDLPRIV can be coded to return an error return code without invoking any security package services; you can also do this with any other DDL statement that you do not want to make available to your users.

CREATE VIEW checking

To use the CREATE VIEW statement, the user must have both the correct privileges to use the statement and at least SELECT access to the SQL objects referenced in the VIEW.

In this case Model 204 calls DDLPRIV to check the user's DDL privilege, and then calls DMLPRIV an appropriate number of times to validate the user's access to SQL objects.

DMLPRIV exit

When compiling an SQL query for a user, the Model 204 SQL processor checks the user's privileges to access SQL objects. To check privileges, the compiler calls a dictionary routine that has access to privilege data in CCACAT. The arguments to the routine are the user's ID, the full name of the SQL object, and the type of access the user requires to run the query.

You can also use the DMLPRIV security exit to check user privileges. The types of SQL objects for which privileges are checked are TABLE, VIEW, and COLUMN. The types of access for which privilege checking is done are:

- INSERT
- UPDATE
- SELECT
- DELETE

External security packages do not have access privileges named SELECT, INSERT, UPDATE, or DELETE. Rocket Software, therefore, suggests that you define your SQL objects as pseudo data set names and then map the SQL privileges to those of the external security package. For appropriate access codes to use when validating pseudo data set names, see Table 6-3 on page 134.

Column level checking

SQL UPDATE access type checking can be done at the individual column level. For UPDATE, DMLPRIV has an additional argument, which is an array of column names.

If the user does not have the appropriate privileges required by the query, then further processing of the query is terminated and Model 204 issues an error message.

View checking

A VIEW is regarded as an independent SQL object for the purposes of checking access privileges. Once a user's access to a view has been established no further check of the objects that compose the view is made. That is, for DML, no differentiation is made between a view or a base table.

Parameters passed to SQL

The parameters passed to DMLPRIV from Model 204 are:

DMLPRIV userid, schema, table, cols, priv, retcode

where:

<i>userid</i>	Is the address of a character string containing the user's Model 204 user ID. The string is terminated with a binary zero. The user ID parameter is passed in T1.
<i>schema</i>	Is the address of a character string containing the schema name. The string is terminated with a binary zero. The schema name parameter is required and is passed in T2.

<i>table</i>	<p>Is the address of a character string containing the view or table name. The string is terminated with a binary zero. The table parameter is passed in T3.</p> <p>The schema and table names cannot be concatenated, because the length of the combined name might be greater than your security package can tolerate for a resource name. Specifying the schema and table names separately also provides greater flexibility in defining the rules for your site.</p>
<i>cols</i>	<p>Points to an array of column names (that is, <i>cols</i> is a pointer to a list of pointers). The array is terminated by a binary zero. The <i>cols</i> parameter is passed in T4.</p> <p>If privilege checking is not required at the column level, this is a null value.</p>
<i>priv</i>	Is an integer code describing the type of access. The <i>priv</i> parameter is passed in T5. The access codes are listed in Table 6-4.
<i>retcode</i>	Is the return code from the call passed back to Model 204 in R1. The return codes are described on page 134.

Table 6-3. Access codes and pseudo data set names for SQL privileges

Access code	SQL privileges	CA-ACF2	Security Server	CA-Top Secret
0	SELECT	execute	execute	execute
1	INSERT	read	read	read
2	UPDATE	write	write	write
3	DELETE	allocate	alter	create

Return codes for DDLPRIV and DMLPRIV security exits

Table 6-4 lists the values of and meanings of return codes for DDLPRIV and DMLPRIV security exits.

If, for example, your site does not use the AUTHID in the rules for checking DDL privileges, you can ignore the parameter in your SQL security exit. As a result, DDLPRIV never returns a return code of 8.

Table 6-4. Return codes for DDLPRIV and DMLPRIV security exits

Value	Meaning
0	Success. The DDL or DML statement will be executed.
1	The user's privileges are not sufficient.

Table 6-4. Return codes for DDLPRIV and DMLPRIV security exits

Value	Meaning
2	Model 204 does not recognize the user ID.
3	Model 204 does not recognize the access type.
4	Model 204 does not recognize the schema name.
5	Model 204 does not recognize the table name.
6	Model 204 does not recognize the column name.
7	Model 204 does not recognize the DDL statement type.
8	Model 204 does not recognize the AUTHID.
9	User exits are inactive: use normal security.

Installing the SQL security exits

DMLPRIV and DDLPRIV must be included as part of the Model 204 CDTB module. You can then assemble and link CDTB following the instructions in the *Model 204 Installation Guide* for your operating system.

Index

Symbols

- @CFDE entries in ACFFDR, creating
 - CA-ACF2 MVS 28
 - CA-ACF2 VM 64
- @MUSASS macro, modifying, CA-ACF2 MVS 26
- @RESTYPE macro, updating, CA-ACF2 VM 64
- @SRF macro, updating, CA-ACF2 VM 64

A

- access controls, CA-Top Secret 103
- access levels, CA-Top Secret 103
- Accessor environment element (ACEE), Security Server 74
- Accessor ID (ACID), CA-Top Secret 100
 - default ACID 119
 - defining Model 204 115
 - login validation 107
 - overriding default 119
- ACCOUNT argument, ACF2GEN
 - CA-ACF2 MVS 21
 - CA-ACF2 VM 54
- ACCOUNT option, VALIDAT argument
 - TOPSGEN 122
- ACCOUNT option, VALIDAT argument, RACFGEN 91
- account validation
 - CA-ACF2 VM 43, 68
 - Security Server (formerly RACF) 79
- account, login
 - CA-ACF2 MVS 10, 13
 - CA-ACF2 VM 42
 - CA-Top Secret 106
 - Security Server (formerly RACF) 78
- ACF command, CA-ACF2 MVS 18
- ACF2 MVS Interface see CA-ACF2 MVS
- ACF2 VM Interface see CA-ACF2 VM
- ACF2CMS module, CA-ACF2 VM 41
- ACF2GEN
 - CA-ACF2 MVS 19
 - CA-ACF2 VM 44, 52
 - NOPASSWORD argument
 - CA-ACF2 VM 42

- ACF2PARM module
 - CA-ACF2 MVS 10, 19
 - assembling 25
 - Logonid record and 29
 - sample 23
 - CA-ACF2 VM 52
 - creating 61
 - sample 56
- ACFDIAG macro, CA-ACF2 VM 41
- ACFDIAGC, applying for CA-ACF2 VM 61
- ACFFDR (ACF2 Field Definition Record)
 - assembling, CA-ACF2 VM 66
 - modifying
 - CA-ACF2 MVS 27
 - CA-ACF2 VM 64
- ACID argument, TOPSGEN 122
- ACID see Accessor ID (ACID), CA-Top Secret
- ADDSD statement, Security Server (formerly RACF) 75, 91
- ALTER authority, Security Server (formerly RACF) 79
- AUTHCTL command
 - CA-ACF2 MVS 14
 - CA-Top Secret 112
 - Security Server (formerly RACF) 84
- authorization checking, Security Server (formerly RACF) 74
- authorization, for CA-Top Secret 106

C

- CA-ACF2 MVS
 - ACF2PARM parameter 19
 - ACF2PARM sample 23
 - AUTHCTL command 14
 - decrypting 24
 - defining Generalized Resource Rules 18
 - defining Logonid fields 27
 - defining user privileges 17
 - dynamic allocation with 11
 - HLI with 10
 - installation checklist 31
 - installing 24
 - logging in 9

- login processing 13
- Logonid record for 29
- maintenance 17
- MUSASS, defining 26
- overview 3
- privilege rules 19
- sequential and VSAM data sets 12
- storage requirements 30
- submitting jobs 11
- terminal security 24
- trusted login feature 15
- TSO with 12
- UID (user ID) 19
- User 0 login 14
- with CRAM 12
- CA-ACF2 VM
 - account validation 43, 68
 - ACF2GEN macro 52
 - ACFFDR, updating 64
 - configurations 35
 - decrypting 60
 - diagnoses 62
 - execute-only rules, writing 63
 - Generalized Resource Rules for 67
 - HLI with 43
 - installing 56
 - LIDREC dsect, updating 65
 - logging in 42
 - login validation 68
 - Logonid record 44
 - M204SCTY module, generating 61
 - maintenance 47
 - modules, generating 60
 - overview 34
 - sample rules 63
 - saved segments, creating 60
 - SECPLIST rules 67
 - SECUR204 machine VM directory entry 58
 - security module, generating 61
 - service machine 35
 - service machine commands 49
 - single-user Onlines 47
 - source entry rules, writing 68
 - source validation 43
 - statistics 51
 - User 0 login 46
 - user ID string 39
 - user privileges, defining 67
 - VM directory entry, adding 58
- CA-Top Secret
 - access controls 103
 - Accessor ID (ACID) 100
 - decrypting 120
 - default ACID 119
 - defining Model 204 to 114
 - department element 101
 - division element 102
 - dynamic allocation 108
 - facilities 102
 - Facilities Matrix 114
 - installation checklist 125
 - installing 120
 - logging in 106
 - login messages 110
 - maintaining 113
 - overview 97
 - processing 100
 - profile element 101
 - SECPLIST parameter 120
 - submitting jobs 108
 - terminal security 119
 - TOPSGEN macro 121
 - trusted login feature 112
 - User 0 login 111
 - user element 101
 - user privileges, defining 117
- CCAIN files, CA-ACF2 VM, updating 69
- CCASTAT
 - CA-ACF2 MVS 10, 14
 - CA-ACF2 VM 42, 45
 - CA-Top Secret 107
 - RACF 78
 - setting user privileges
 - CA-ACF2 MVS 18
 - CA-ACF2 VM 48
- COMACID argument, TOPSGEN 122
- Comgroup.PRIORITY profile 92
- COMGRUP argument, RACFGEN 91
- CRAM
 - considerations
 - CA-ACF2 MVS 24
 - RACF 89
 - trusted login feature 84
 - CA-ACF2 MVS 15
 - CA-Top Secret 112
 - with CA-ACF2 MVS 12
- creating saved segments, CA-ACF2 VM 60
- CRFS channel thread
 - CA-ACF2 MVS 16
 - CA-Top Secret 113
 - RACF 85
- CRIO channel thread
 - CA-ACF2 MVS 16
 - CA-Top Secret 113
 - RACF 85

D

- D option, AUTHCTL command
 - CA-ACF2 MVS 15
 - RACF 84
- data sets
 - pseudo data set names, CA-Top Secret 118
 - sequential or VSAM
 - CA-ACF2 MVS 12
 - CA-Top Secret 108
 - RACF 81
- DDL statements
 - security for 131
 - statement codes 132
- DDLPRIV exit 130
 - installing 135
 - parameters 131
 - return codes 134
- decrypting
 - CA-ACF2 MVS 24
 - CA-ACF2 VM 60
 - CA-Top Secret 120
 - RACF 89
- DEFACID argument, TOPSGEN 123
- default Logonid, CA-ACF2 MVS 29
- default user ID, defining
 - CA-Top Secret 116
 - RACF 94
- DEFPASS argument, CA-Top Secret 123
- DEFPASS argument, RACFGEN 92
- DEFUGRUP argument, RACFGEN 92
- DEFUSER argument, RACFGEN 92
- department element, CA-Top Secret 101
- diagnoses, CA-ACF2 VM 62
- discrete profiles, RACF 76
- division element, CA-Top Secret 102
- DLMCHECK argument
 - ACF2GEN 22
 - RACFGEN 93
 - TOPSGEN 123
- DMLPRIV exit 132
 - installing 135
 - parameters for 133
 - return codes 134
- dynamic allocation
 - CA-ACF2 MVS 11
 - CA-Top Secret 108
 - RACF 79

E

- elements, CA-Top Secret 100
- EOJ command, CA-ACF2 VM 51

F

- facilities, CA-Top Secret 102
- FORCE command, CA-ACF2 VM 50

G

- Generalized Resource Rules
 - CA-ACF2 MVS 8, 18
 - CA-ACF2 VM 40, 66
- generating modules, CA-ACF2 VM 60
- generic profiles, RACF 76
 - defining 88
- global access checking, RACF 75
- GROUP argument, RACFGEN 91

H

- HLI
 - CA-ACF2 VM 43
 - CA-Top Secret 107
 - RACF 79
 - trusted login with
 - CA-ACF2 MVS 16
 - CA-Top Secret 113
 - RACF 85

I

- IFAM2, trusted login with
 - CA-ACF2 MVS 16
 - CA-Top Secret 113
 - RACF 85
- IFLOG function
 - CA-ACF2 MVS 11
 - CA-ACF2 VM 43
 - CA-Top Secret 107
 - RACF 79
- IFOPEN function
 - CA-ACF2 VM 43
- IFSTRT function
 - CA-ACF2 MVS 10
 - CA-ACF2 VM 43
 - CA-Top Secret 107
 - RACF 79
 - with trusted login
 - CA-ACF2 MVS 16
 - CA-Top Secret 113
 - RACF 85
- installation exits, RACF 75
- installing
 - CA-ACF2 MVS 24, 31
 - CA-ACF2 VM 56

- CA-Top Secret 120, 125
- RACF 88, 95
- SQL security exits 135
- IODEV threads, trusted login
 - CA-ACF2 MVS 16
 - CA-Top Secret 112
 - RACF 85
- IUCV communications, CA-ACF2 VM 48

J

- job submission
 - CA-ACF2 MVS 11
 - CA-Top Secret 108
 - RACF 80

L

- LIDREC (Logonid record)
 - CA-ACF2 MVS 27
 - CA-ACF2 VM 39
 - updating 64
- LIDREC dsect, updating, CA-ACF2 VM 65
- link-editing
 - CA-Top Secret 124
 - TOPSGEN 124
- LIST option, AUTHCTL command
 - CA-ACF2 MVS 15
 - RACF 84
- LOGCTL command, password changes with
 - CA-ACF2 MVS 18
 - CA-ACF2 VM 48
- logging in
 - as User 0
 - CA-ACF2 MVS 14
 - CA-ACF2 VM 46
 - CA-Top Secret 111
 - RACF 83
 - CA-ACF2 MVS 13
 - CA-ACF2 VM 42
 - CA-Top Secret 106, 109
 - RACF 78
 - trusted login
 - CA-ACF2 MVS 15
 - CA-Top Secret 112
 - RACF 84
- LOGIN argument, ACF2GEN, CA-ACF2 MVS 21, 29
- LOGIN command
 - CA-ACF2 MVS 9
 - CA-ACF2 VM 42
 - CA-Top Secret 106
 - RACF 78

- with trusted login
 - CA-ACF2 MVS 15
 - CA-Top Secret 112
 - RACF 84
- login messages, CA-Top Secret 110
- login processing
 - CA-ACF2 MVS 13
 - CA-ACF2 VM 43
 - CA-Top Secret 109
 - trusted login
 - CA-ACF2 MVS 16
 - CA-Top Secret 112
 - RACF 85
- login validation
 - CA-ACF2 MVS 14
 - CA-ACF2 VM 68
 - CA-Top Secret 107
 - RACF 79
- LOGON command see LOGIN command
- LOGONID argument, ACF2GEN
 - CA-ACF2 MVS 22
 - CA-ACF2 VM 46, 56
- Logonid fields, defining, CA-ACF2 MVS 27
- Logonid record
 - CA-ACF2 MVS 6, 10, 13
 - default 29
 - CA-ACF2 VM 39, 44
 - updating fields in 66

M

- M204CRYP command, CA-ACF2 VM 60
- M204USR default user ID, RACF 94
- MAINT204 machine, CA-ACF2 VM 59
 - PROFILE EXEC for 59
- maintaining
 - CA-ACF2 MVS 17
 - CA-ACF2 VM 47
 - CA-Top Secret 113
 - RACF 86
- masking
 - data sets, CA-Top Secret 103
 - UID entries, CA-ACF2 VM 39
- Model 204
 - CA-Top Secret ACID, defining 115
 - CDTB module for SQL security exits 129
 - defining
 - to CA-ACF2 MVS 26
 - to CA-Top Secret 114
 - to RACF 94
 - generating modules, CA-ACF2 VM 60
 - link-editing
 - CA-ACF2 MVS 25

- CA-Top Secret 124
- RACF 93
- Logonid records, CA-ACF2 VM 62
- privileges, defining
 - CA-ACF2 MVS 29
 - CA-ACF2 VM 47
 - RACF 87
- saved segments, saving 61
- user ID
 - CA-ACF2 MVS 13
 - CA-ACF2 VM 44
 - CA-Top Secret 109
 - RACF 82
- Model 204 EXECs, updating, CA-ACF2 VM 69
- Model 204 privileges, defining
 - CA-Top Secret 118
- MODEL204 machines, CA-ACF2 VM 59
- PROFILE EXECs for 59
- MUSASS (Multiuser Single Address Space System)
 - CA-ACF2 MVS 8
 - defining 26
 - CA-ACF2 VM 38

N

- NAME argument
 - ACF2GEN
 - CA-ACF2 VM 54
 - RACFGEN 90
 - TOPSGEN 122
- NODLMCHECK argument
 - ACF2GEN 22
 - RACFGEN 93
 - TOPSGEN 123
- NON-CNCL attribute, @MUSASS, CA-ACF2 MVS 26
- NOPASSWORD argument, ACF2GEN, CA-ACF2 VM 42, 55
 - in IFAM2 44

P

- PASSWORD argument, ACF2GEN, CA-ACF2 VM 55
- password, ACF2 violation counter
 - CA-ACF2 MVS 14
 - CA-ACF2 VM 45
- password, defining
 - ACF2GEN, CA-ACF2 VM 55
 - RACFGEN 92
 - TOPSGEN 123
- PERMIT authority
 - CA-Top Secret 118, 123

- RACF 88
- PERMIT statement
 - CA-Top Secret 116
 - RACF 88
- PRIORITY option, VALIDAT argument
 - RACFGEN 92
 - TOPSGEN 123
- priority, defining
 - CA-ACF2 MVS 22
 - CA-ACF2 VM 55
 - CA-Top Secret 122
 - RACF 91
- privilege rules, CA-ACF2 MVS 19
- profile element, CA-Top Secret 101
- PROFILE EXECs, CA-ACF2 VM 59
- profiles, RACF 75
- PRTY argument
 - ACF2GEN
 - CA-ACF2 MVS 22
 - CA-ACF2 VM 45, 55
 - RACFGEN 91
 - TOPSGEN 122
- pseudo data sets, CA-Top Secret 118
- PURGE command, CA-ACF2 VM 50

R

- RACF
 - account validation 79
 - ADDSD statement 76
 - ALTER authority 79
 - AUTHCTL command 84
 - authorization checking 74
 - decrypting 89
 - global access checking 75
 - installation exits 75
 - installing 88, 95
 - logging in 78
 - login validation 79
 - maintenance 86
 - Model 204 users, defining 94
 - overview 71
 - PERMIT statements 88
 - priorities 91
 - processing 74
 - RACF group 80
 - SECPLIST parameter 89
 - terminal security 89
 - trusted login feature 84
 - User 0 login 83
- RACFGEN macro, RACF 90
- RACFPARM module
 - assembling and linking 93

- default user ID and 94
- record security key, defining, ACF2GEN
 - CA-ACF2 MVS 21
 - CA-ACF2 VM 55
- RECSCTY argument, ACF2GEN
 - CA-ACF2 MVS 21
 - CA-ACF2 VM 45, 55
- RESOURC argument, ACF2GEN
 - CA-ACF2 MVS 20
 - CA-ACF2 VM 54
- user privileges and
 - CA-ACF2 MVS 17
 - CA-ACF2 VM 48
- resource ownership, CA-Top Secret 104
- resources, CA-Top Secret 102
- rules
 - execute-only, CA-ACF2 VM 63
 - sample, CA-ACF2 VM 63

S

- saved segments, creating for CA-ACF2 VM 60
- SBA2OS module, assembling, CA-ACF2 MVS 25
- SECP LIST parameter, CCAIN
 - CA-ACF2 MVS 20
 - CA-ACF2 VM 52
 - writing rules for 67
 - CA-Top Secret 120
 - RACF 89
- SECTRLOG parameter, trusted login
 - CA-ACF2 MVS 30
 - CA-Top Secret 124
 - RACF 95
- SECUR204 machine, CA-ACF2 VM
 - PROFILE EXEC for 59
 - VM directory entry 58
- SECURID argument, ACF2GEN, CA-ACF2 VM 49, 54
- SECURITY command, CA-ACF2 VM 49
- security exits, SQL see SQL security exits
- security modules, creating for CA-ACF2 VM 61
- security service machine see service machine, CA-ACF2 VM
- sequential data sets
 - CA-ACF2 MVS 12
 - RACF considerations 81
- service machine, CA-ACF2 VM 35, 48
 - commands for 49
- single-user Onlines, CA-ACF2 VM 47
- source entry rules, CA-ACF2 VM 68
- source validation, CA-ACF2 VM 43
- SPCORE parameter, CA-ACF2 MVS 30
- SQL security exits

- DDLPRIV exit 130
- DMLPRIV exit 132
- installing 135
- overview 127
- requirements for 128
- return codes 134
- START command, CA-ACF2 VM 50
- statistics, CA-ACF2 VM 51
- STATUS command, CA-ACF2 VM 49
- STOP command, CA-ACF2 VM 50
- storage requirements, CA-ACF2 MVS 30
- SUBMIT option of VALIDAT argument, RACFGEN 92
- submitting jobs
 - CA-ACF2 MVS 11
 - CA-Top Secret 108
 - RACF 80

T

- TERMINAL, CA-ACF2 MVS 12
- terminal security
 - CA-ACF2 MVS 24
 - CA-Top Secret 119
 - RACF 89
- Top Secret see CA-Top Secret
- TOPSGEN macro 121, 124
 - sample 121
- TOPSPARM module 99
- TOPSPARM module, CA-Top Secret 121
- trusted login
 - CA-ACF2 MVS 15
 - CA-Top Secret 112
 - RACF 84
 - SECTRLOG parameter
 - CA-ACF2 MVS 30
 - CA-Top Secret 124
 - RACF 95
- TSO SUBMIT, CA-ACF2 MVS 12

U

- UID (user ID)
 - CA-ACF2 MVS 19
 - CA-ACF2 VM 39
- USE \$JOB command
 - CA-Top Secret 108
 - RACF with 80
- USE command, in CA-ACF2 MVS 11
- User 0 login
 - CA-ACF2 MVS 14
 - CA-ACF2 VM 46
 - CA-Top Secret 111

- RACF 83
- user element, CA-Top Secret 101
- user ID
 - CA-ACF2 VM 42
 - CA-ACF2-MVS 10
 - CA-Top Secret 106
 - Model 204
 - CA-ACF2 MVS 13
 - CA-ACF2 VM 44
 - CA-Top Secret 109
 - RACF 82
 - RACF 78, 94
 - submitting jobs 80
- User Identification (UID) field, CA-ACF2 MVS 7
- user privileges
 - defining
 - CA-ACF2 MVS 17
 - CA-ACF2 VM 47, 67
 - CA-Top Secret 117
 - RACF 86
 - names of
 - CA-ACF2 MVS 17
 - CA-ACF2 VM 47
 - RACF 86
 - RACF 88
- user profile record, RACF 78
- USERLID extension of LIDREC, CA-ACF2 MVS 28

V

- VALIDAT argument
 - ACF2GEN
 - CA-ACF2 MVS 22
 - CA-ACF2 VM 44, 54
 - RACFGEN 91
 - TOPSGEN 122
- VIEW option, AUTHCTL command
 - CA-ACF2 MVS 15
 - CA-Top Secret 112
 - RACF 84
- VM directory, modifying 58
- VSAM data sets
 - CA-ACF2 MVS 12
 - RACF requirements 81