acf^2® MVS

The Access Control Facility Administrator's Guide

acf2®

The Access Control Facility

ADMINISTRATOR'S GUIDE

for

acf2/MVS Release 4.1 Installations

Base Manual Dated: January 31, 1986

Doc. Nr. ABP0005-04

Printed in U.S.A.

acf2/MVS is a proprietary product developed and maintained by:

SKK, Inc.
10400 West Higgins Road
Rosemont, Illinois 60018-3790

Business Office: (312) 635-1040
Product Support: (312) 635-3000
TELEX: 206-186 (SKK ROSM)

A 24 hour answering service on (312) 825-5150 is available
for emergency assistance outside of normal business hours.

ACF2 ADMINISTRATOR'S GUIDE

INTRODUCTION

PURPOSE OF MANUAL

The acf2/MVS Administrator's Guide provides basic information for system administrators on the components and functions of the Access Control Facility (ACF2).

An installation may want to provide system users with selected portions of this manual, as follows. The installation may need to alter these recommendations according to the responsibilities of the various individuals:

Chapter	Security Officer	Account Manager	EDP Auditor	Other Users
Overview	X	X	X	X
System Entry Procedures Under ACF2	X	X	X	X
Introduction to the Logonid Record	X	X	X	X
ACF Command	X	X	X	X
LID Setting	X	X	X	
RULE Setting	X	X	X	X
ACF Setting	X	X	X	
RESOURCE Setting	X		X	
ENTRY Setting	X		X	
SCOPE Setting	X		X	
SHIFT Setting	X		X	
CONTROL Setting	X		X	
Utilities	X	X	X	
Console Operator Commands	X		X	
Special ACF2 Procedures	X		X	

This manual is intended for users of acf2/MVS and assumes that the user has a basic knowledge of fundamental data processing concepts.

The security-oriented environment achieved by the installation of the ACF2 system is important to all users. Therefore, users of this security system are encouraged to read and study this guide as a source of basic information. Various other manuals supplied with the ACF2 package, which provide more detailed information for selected areas, are described below.

DIRECTORY OF OTHER DOCUMENTATION

Numerous manuals in addition to this Administrator's Guide are provided with the ACF2 system package. The brief descriptions that follow are provided for the general ACF2 system user. Of course, special users, such as security officers, auditors, and installation team members will want to refer to many of these other manuals for more detailed information and instructions.

Overview
The acf2/MVS Overview is a short introduction to ACF2 and some of its basic components. It is designed as a management overview of the ACF2 product, but should be the first document read by all users.

General Information Manual
The GIM is the basic reference manual for ACF2. It describes the features and functions of the ACF2 system, most of which are covered in greater detail in other specific manuals. Turn to the GIM when you need to find a general definition or a description of a certain process (such as system or data access control).

Utilities Manual
Refer to this document whenever you wish to use an ACF2 utility, batch program, and/or an ACF2 report generator. Field and parameter descriptions, sample JCL, and sample reports are provided here.

Auditor's Guide
This is a specialized manual developed for the needs of your company's auditor(s). Yet, it is a useful source of information for general users and security officers as well. Critical Logonid fields and system option choices are discussed, as well as in-depth profiles of the special ACF2 privileges.

System Programmer's Guide
Your system programming staff will use this manual for various aspects of installation and maintenance of the ACF2 system. It describes the ACF2 databases, records, macros, exits, control blocks, subroutines, etc., from a technical standpoint.

Messages Manual

This reference document contains explanations for all ACF2-initiated messages. Use this manual when you have questions regarding any ACF2 message.

CICS Support Manual

This manual provides information regarding the ACF2/CICS subsystem. This manual also contains a customization guide, information on defining user resources, and worksheets to calculate storage requirements.

IMS Support Manual

This document describes the ACF2 interface to IMS, such as option selection, Logonid record considerations, and storage estimates.

IDMS Support Manual

The ACF2/IDMS interface is described in this manual. Topics include parameter selection, macro definitions, and Logonid record considerations.

Implementation Planning Guide

Your site's ACF2 planning team will use this manual during the initial phases of planning for and implementing ACF2. This manual contains general technical and non-technical considerations for implementing ACF2. It also includes sample checklists and timetables that you can use and modify as necessary during the implementation phase.

Other Products Manual

The OPM provides more detailed information on various software products on the market and their uses/interfaces to ACF2. The products are listed alphabetically. OPM references may also be found in the Composite Index.

Composite Index

The acf2/MVS Composite Index is a cross-reference tool, combining the index entries of the other acf2/MVS manuals. Manual codes (such as GIM for General Information Manual and AUDIT for Auditor's Guide) appear after each entry indicating the manual and page that contains the relevant information.

Command Summary

The Reference Card shows the syntax of the ACF subcommands, ACF2 operator commands, access rules, and generalized resource rules. It also contains the names of fields in a system user's record.

Customer Education Catalog

This catalog contains descriptions of the standard acf2/MVS (including acf2/VS1) and acf2/VM training classes, acf2/VM short class, Independent Study Program (ISP), and models of onsite ACF2 training classes that can be tailored specifically to the needs of an installation. This catalog also contains enrollment and scheduling information. Available upon request.

SYMBOLS FOR SYNTAX REPRESENTATION

Throughout this manual, the syntax of subcommands provided by ACF2 will be represented in a standard fashion. For example:

```
LIST {*/record-name/LIKE(record-name-mask)}
```

This syntax can be interpreted through the following standards:

Slashes /

Separate each of the alternative items. You can enter one and only one of the items. You do not enter any slash itself. In the example, only an asterisk, a record-name, or the LIKE keyword and record-name-mask must be entered. You cannot enter both an asterisk and record-name.

Braces {}

Enclose a group of alternative items that must be entered. You enter one and only one of the items. You do not enter the braces themselves. In the example, an asterisk, a record-name, or the LIKE keyword and record-name-mask must be entered.

Brackets [ ]

Indicate a single or group of optional items. You do not enter the brackets themselves. The example contains no optional items. If an item is not within brackets, then it is required.

Uppercase letters

Represent keywords that must be entered literally, such as a particular subcommand or parameter name. In the example, the name LIST must be literally entered when issuing this subcommand. The name LIKE must be entered if that particular parameter will be used when issuing the subcommand. For example:

```
list like(week-)
```

Lowercase letters

Describe the data that must be entered for a particular parameter. In the above example, record-name indicates that the actual name of a particular record to be listed. For example:

```
list weekdays
```

Asterisks *

Are taken literally if they appear as part of the syntax. For example:

```
list *
```

They indicate the name(s) of the previous record(s) processed.

Underscoring ________

Indicates a default. If an item is underscored, then it is the default value used when no item has been specified.

Apostrophes and Quotes ", '

Act as delimiters when used in the middle of an entry. For example, when entering a user name on a Logonid such as O'Henry, enclose the information in arbitrary delimiters. This applies to rules, Logonids or any free form data stored on the ACF2 databases.

```
     insert ssdth name("Tom O'Henry")
```

OVERVIEW OF ACF2

ACF2, the Access Control Facility, provides protection against unauthorized destruction, disclosure, or modification of data and resources at your installation. ACF2 operates as an extension of your operating system.

ACF2 protects all data by default. That is, a user who is not the owner of the data can access the data only if the owner or the security officer has explicitly authorized such access.

MAJOR COMPONENTS OF ACF2

From an administrative standpoint, ACF2 is tailored to individual users, datasets, and resources through:

1. Logonid Records, which define each system user in terms of general identification, status, privileges, access history, attributes related to TSO, CICS, IMS, IDMS, and VM, violation statistics, etc.

2. Access Rules, which describe the conditions (environment) for accessing particular datasets, and which determine whether access will be allowed or prevented for a user or group of users.

3. Generalized Resource Rules, which specifically allow or prevent a user or group of users to access generalized resources. Resources may include: TSO accounts; TSO procedures; IMS applications and transactions; CICS files, programs, transactions, transient data, temporary storage, and DL/I calls; or any other resource an installation wishes to define.

4. Entry Records, which allow an installation to specify access only from specific input sources or groups of input sources. Another type of entry record helps make Operator Identification (OID) card logon validation possible.

5. Scope Lists, which limit the authority a specially privileged user has over Logonid records, access rules, and other ACF2 records.

6. Shift Records, which define periods of time when access is allowed or prevented. Zone records offset the user's local time from the executing CPU time.

7. GSO Records, which specify an installation's ACF2 Global System Options.

Each of these components can be updated dynamically through the use of ACF2 TSO commands, ISPF screens, or batch utilities. Each is explained in detail in a separate chapter of this manual.

OTHER COMPONENTS OF ACF2

The components above are all contained on the ACF2 databases. Other components include these databases as well as the following items:

1. Three Databases contain the components previously described:

   * Logonid database contains Logonid records for all users on the system.

   * Rules database contains all dataset access rules.

   * Infostorage database contains generalized resource rules, entry records, scope lists, shift/zone records, and Global System Option (GSO) records.

2. ACF Command and Subcommands, under TSO, allow you to create and maintain the major components of ACF2. These subcommands are also available for ACF2 batch processing. A HELP command is available for providing both instructions on the use of commands, and descriptions of various fields.

3. Report Generators and Utilities assist with security maintenance, administration, and auditing. The ACF2 report generators produce reports and audit trails. An installation can use these reports and audit trails to implement and maintain security, and to monitor certain access and security violations. Most reports use data produced by ACF2 and recorded via IBM's SMF (System Management Facility).

   ACF2 utilities provide tools for maintaining and enhancing security functions at your installation. These report generators and utilities are listed in the chapter "Reports and Utilities" and are described in detail in the acf2/MVS Utilities Manual.

4. Your Password is a unique string of characters. You enter it in addition to the Logonid to prove your identity to the system. Once entered, the password is one-way encrypted so that it is not stored as it was entered. ACF2, however, cannot protect passwords outside the computer; such controls are the responsibility of the user.

5. A User Identification (UID) String exists for each user of your system. It is used during ACF2 access validation, since it usually contains not only the Logonid but also other information about the user. Such information may allow ACF2 to grant access to data and resources according to company name, department code,

site or branch, or job responsibility code. Your installation has the option of determining the type and quantity of information contained in the UID string.

6. The ACF2 Field Definition Record (ACFFDR) is made up of Assembler language macros that:

   * Define and establish controls for each field of data in the Logonid record. The Logonid record contains the same fields for all users.

   * Specify system options related to the Logonid record and to the operation of ACF2.

   Changes to this information are made only periodically and require reassembly of the ACFFDR and a system IPL. The ACFFDR is described in detail in the acf2/MVS System Programmer's Guide.

7. SPF Screens, for MVS (TSO/ISPF) installations, provide users with menus to conveniently select and then perform the following ACF2 administration online:

   * Adding, changing, deleting, testing, and listing access and generalized resource rules.

   * Adding, changing, deleting, and listing Logonid records.

   * Displaying ACF2 system processing options.

   * Creating ACF2 reports at the terminal.

   * Executing ACF2 utility programs at the terminal.

   * Adding, changing, deleting, and listing ACF2 Global System Options.

SYSTEM ENTRY PROCEDURES UNDER ACF2

This chapter contains procedures for:

1. TSO logon
2. TSO fullscreen logon
3. TSO User Authentication Logon Support
4. CICS sign-on
5. IMS sign-on
6. IDMS sign-on
7. Batch system entry (JES2, JES3, RJE, and NJE)

The logon procedures for ROSCOE, WYLBUR, and other online systems are similar to those for TSO. Check with your installation for exact procedures.

TSO LOGON PROCEDURE

To access your system via TSO (IBM's Time Sharing Option), follow the steps outlined below. Note that in this example, system responses are capitalized. Information entered by the user is in lowercase.

Type:

```
logon your-logonid
```

An alternative logon syntax (VTAM) is:

```
logon 'your-logonid logon-operands'
```

Press:

ENTER key on your keyboard

System reply:

```
ACF82004 ACF2, ENTER PASSWORD -
```

Type:

your-password
(entered into a non-displayable field)

Press:

ENTER key on your keyboard

ACF2 completes its logon processing.

NOTE: A TSO logon can be aborted by entering a plus sign (+) in response to any ACF2 prompt.

Changing Your Password

To change your password, follow your usual logon procedure until the system prompts you with the message: ENTER PASSWORD-. Enter your old password, a slash, and then your new password (i.e., old-password/new-password). A password can be from 1 to 8 characters in length. The system will prompt:

```
ACF82020 ACF2, REENTER NEW PASSWORD FOR VERIFICATION -
```

Enter your new password again. If you reenter your new password successfully, the system displays:

```
ACF82007 ACF2, LOGON IN PROGRESS
ACF01129 PASSWORD SUCCESSFULLY ALTERED
```

The logon procedure continues as usual.

If you reenter your new password incorrectly, the system displays:

```
ACF82916 ACF2, VERIFICATION OF NEW PASSWORD FAILED
ACF82008 ACF2, ENTER NEW PASSWORD -
```

You must enter your new password, and then enter it again when the system asks you to reenter it for verification. If you successfully reenter your new password, the logon procedure will continue as usual.

Separate Entry of Logonid and Password Recommended

Your installation may optionally allow the LOGON command, Logonid, and password to be entered on one line such as:

Type:

```
logon logonid/password
```

You may also enter a new password along with your old password simply by typing a slash (/) after the old and then entering the new as follows:

```
logon logonid/old-password/new-password
```

However, entering all this information on one line is not recommended because in this way your password is visible on the screen, and can be read by any nearby observer. Your installation can force the use of separate entry by means of the NOQLOGON option in the GSO OPTS record.

TSO LOGON Operands

The TSO LOGON operands which can be specified are:

- ACCT(acct-number) specifies the account number to be associated with this TSO session (1-40 characters). User must have LGN-ACCT in order to specify ACCT.

- FORCE indicates that the user wishes to gain access to the system even though ACF2 was never started. The user must also be defined in the UADS dataset.

- INDEX(uads-pswd) specifies the password to be used for UADS validation when UADS=YES is specified in the GSO TSO record (1-8 characters). User must have LGN-INDX in order to specify INDEX.

- MAIL/NOMAIL indicates to TSO that the user does or does not want to see any messages from TSO at LOGON time.

- MSGCLASS(message-class) specifies the desired message class for this TSO session (1 character). User must have LGN-MSG in order to specify MSGCLASS.

- NOTICES/NONOTICES indicates to TSO that the user does or does not wish to receive TSO notices at LOGON time.

- PERFORM(perf-group) specifies the desired performance group for this TSO session (1-3 digits). User must have LGN-PERF in order to specify PERFORM.

* PROC(procedure) specifies the procedure to be used for this TSO session (1-8 characters). User must have LGN-PROC in order to specify PROC.

* RECONNECT specifies that the user wishes to re-establish a TSO session which was disconnected. R may be used instead of RECONNECT.

* RECOVER/NORECOVER indicates that the user wants the TSO RECOVER option on or off for this TSO session.

* SIZE(region-size) specifies the desired region size in K-bytes for this TSO session (1-5 digits). Any user can specify SIZE but the value must not exceed TSORGN value unless user has LGN-SIZE.

* TIME(session-time) specifies the desired CPU time for this TSO session. User must have LGN-TIME in order to specify TIME.

* UNIT(unit-name) specifies the desired default generic unit name for this TSO session (1-8 characters). User must have LGN-UNIT in order to specify UNIT.

Verifying That No Unauthorized Use of Your Logonid Has Occurred

Your installation can provide an option to verify that there has been no unauthorized use of your Logonid since your last system entry. Under these options, the following message appears after ACF2 has completed its logon processing:

```
ACF01137 your-logonid LAST SYSTEM  ACCESS hh:mm-mm/dd/yy FROM
         source
```

This message appears if there are no warning messages and if the NOTIFY and LIDRECL(1024) options have been selected in the GSO OPTS record. For further information on this record, see the chapter on GSO records.

TSO FULLSCREEN LOGON PROCEDURE

ACF2 provides optional fullscreen logon support for authorized TSO users. Fullscreen logon features discussed in this section are:

1. Format and display of the logon screen

2. Supplied default logon operands displayed on the screen

3. Ability to bypass fullscreen logon

4. Fullscreen logon support for hardcopy devices

Other areas related to TSO fullscreen logon, including considerations for fullscreen logon under the User Attribute Dataset (UADS), are discussed in the chapter "Special ACF2 Procedures."

Format and Display of the Logon Screen

The logon screen will appear after validation of the Logonid and password if the user has fullscreen attributes.

```
-----------------------------------------------------------------------
|                                                                     |
| -------------- VS2 REL xx.xx TIME SHARING OPTION ------------------ |
|                                                                     |
|  ENTER LOGON PARAMETERS BELOW:                                      |
|                                                                     |
|  USERID    ===> PAYJSD                   MSGCLASS ===>              |
|                                                                     |
|  SOURCE    ===> LV437                    UNIT     ===> SYSDA        |
|                                                                     |
|  PROCEDURE ===> $ABCISPF                 TIME     ===> 0000         |
|                                                                     |
|  SIZE      ===> 1024                                                |
|                                                                     |
|  ACCT NMBR ===> 1234                                                |
|                                                                     |
|  PERFORM   ===> 000                                                 |
|                                                                     |
|                                                                     |
|  ENTER AN 'S' BEFORE EACH OPTION DESIRED BELOW                      |
|                                                                     |
|       -NOMAIL       -NONOTICE        -RECOVER        -RECONNECT     |
|                                                                     |
|  USER KEYS ===>                                                     |
|                                                                     |
-----------------------------------------------------------------------
```

Supplied Default Logon Operands

By default, the following fields may be displayed at your installation. Each of these fields can be saved from session to session except where noted:

USERID
: Specifies the user's Logonid. This value <u>cannot</u> be changed.

SOURCE
: Specifies the physical or logical name of the input device you are at. This value <u>cannot</u> be changed.

PROCEDURE
Specifies the name of the procedure containing the JCL for initiating the TSO session. Can be up to eight characters.

SIZE
Specifies the maximum TSO region size for the user. Can be up to 8M.

ACCT NMBR
Specifies the TSO account number required by the installation. Can be up to 40 characters.

PERFORM
Specifies the TSO performance group to be used during the session. Can be any value from 1 to 255 for MVS/non-SE installations and from 1 to 999 for installations with MVS/SE2 and above.

MSGCLASS
Specifies the user's TSO message class. Must be one character.

UNIT
Specifies the user's TSO unit name. Can be up to eight characters.

TIME
Specifies the maximum CPU time allowed for any job. This value can be specified as mmss (minutes seconds). 1440 means unlimited time.

USER KEYS
(At the bottom of the screen) accommodates special keywords required by the installation for logon.

Also on the logon screen, you can enter 'S' before each of the following operands that you wish to have in effect:

NOMAIL
Suppresses display of system mail at logon time.

NONOTICE
Suppresses display of TSO notices at logon time.

RECOVER
Creates a workfile during your editing session, which you can use for recovering edits made to a dataset in the event of a disconnect or system failure.

RECONNECT
Specifies the option to log on again. This logon must occur within a reconnect time limit after your line has been disconnected. When logging on again, you must specify the same Logonid and password as you used previously for beginning the interrupted session; operand values from the interrupted session remain in effect and cannot be changed. The value of the RECONNECT operand cannot be saved from session to session.

User or installation-wide options for TSO fullscreen logon and these logon operands are discussed in the chapter on special ACF2 procedures. Also, see the explanations of the TSO and TSOKEYS records in the chapter on GSO records.

Bypassing Fullscreen Logon

A TSO user can bypass any normal display of the logon screen. During logon, the user enters the TSO LOGON command, specifying the Logonid and the operand keyword NFSCREEN. For example:

```
logon 'paysdh nfscreen'
     --or--
logon paysdh nfscreen
```

After Logonid and password validation, ACF2 proceeds as if no fullscreen authorization exists.

Fullscreen Logon Support for Hardcopy Devices

The fullscreen display is limited to IBM 3270-type display terminals. If you are at a hardcopy terminal and are authorized for fullscreen logon, you may see a message similar to the following one printed at the terminal at logon time:

```
ACF82022  ACF2, THE FOLLOWING KEYWORDS ARE IN EFFECT:
LOGON PAYJSD/PAYJSD   ACCT(4) PROC($SKKISPF) SIZE(01024) UNIT(SYSDA)
ACF82021  ACF2, ENTER OVERRIDES OR HIT ENTER TO CONTINUE
```

You may do one of the following:

1. Enter any of the listed operands to change the values in effect. For example, you can change the value of the SIZE operand to 512K bytes by entering: size(512). ACF2 repeats the logon message but lists any new values.

2. Enter operands not already listed to put additional values in effect. For example, you can put the NOMAIL operand into effect by entering: NOMAIL. ACF2 repeats the logon message, listing the newly specified operand and any value it may have.

3. Retain the values in effect by pressing the ENTER or the return key. You also press the enter or the return key after changing or adding operands and values as desired. ACF2 will continue validating your logon request. If the validation is successful, normal TSO logon will occur.

Of course, to perform the actions described above in alternative #1 or #2, you must have permission to specify the operand that you want to change at logon time. Authorization for changing logon values is explained in the chapter "Special ACF2 Procedures."

EXTENDED USER AUTHENTICATION DEVICE LOGON

Any user authentication device or routine can be incorporated with the standard ACF2 TSO logon support. The ACF2 user authentication feature expands system access validation so that it is made up of the usual ACF2 logon controls (e.g., password, source, shift) plus some additional user authentication. Up to eight different extended validation routines can be used by the installation.

The additional user authentication can be performed via use of a device or software routine. The use of a device with this interface may require users to enter data into both the device and to the terminal keyboard. Some devices used with this type of processing can input the user's unique information directly to the terminal, such as the Operator Identification Device card readers. Or, the user enters data into the device, the device interprets the information and displays data for the user to then enter into the terminal for validation by a software routine provided by the makers of that device. Software routines can be used in place of the device. SKK has provided an example of how a routine might be utilized for extended authentication. Further information can be found in SKK NOTE #9 in the System Programmer's Guide.

The normal TSO logon process signals ACF2 of the extended authentication requirement via a field in the user's Logonid record. Processing options for another vendor's device or software routines are identified in a Global System Options record on the ACF2 Infostorage Database.

ACF2 coordinates communication between the user and the authentication routine in dialogue fashion. For example, the routine could prompt the user for information, the user would respond, the routine would process the user's response and then possibly would prompt again, and so on.

ACF2 allows the dialogue to continue until the vendor device (or software routine) makes a recommendation as to whether the user should be allowed to access the system. Whether access is allowed or denied is determined entirely by the user authentication routine. ACF2 reacts only on this recommendation. Denied system accesses are reported in the ACF2 Invalid Password/Authority Log report.

During extended authentication processing, the user is required to provide some unique information. The information may change every time a user attempts signon. This information is generally made up of a user challenge algorithm, user-unique keys, and possibly some vendor control data.

This user-provided data can optionally be defined and stored on the ACF2 Infostorage Database. Using this option further centralizes the security and control of a data center. ACF2 can then be used to regulate who and how the information is maintained. And, changes to the data can automatically be included in the standard ACF2 reports and file backups.

For complete information and examples of how to insert, change, or delete individual user authentication records, see the related Chapter on Extended User Authentication in this manual. Information is also contained in the Extended User Authentication Support Chapter of the acf2/MVS System Programmer's Guide.

CICS SIGN-ON PROCEDURE

Under ACF2, CICS sign-on is optional if the ACF2/CICS interface has been installed. If a terminal is running under CICS but a sign-on has not been performed, ACF2 uses the installation-defined default Logonid for validating user access requests.

The sign-on can be under any of the standard formats listed below. Spaces can be substituted for commas:

```
CSSN

CSSN logonid/password/new-password

CSSN LID=logonid,PW=password,NPW=new-password

CSSN PS=password/new-password,NAME=logonid
```

In the standard formats shown above:

* logonid is the user's Logonid as defined under ACF2.

* password is the user's current password under ACF2.

* new-password (optional) is the new password to replace the current password, provided that the installation permits password changes.

Preferably, the user should enter only his Logonid along with the sign-on transaction id so that ACF2 prompts for the password. This procedure allows the user to enter the password separately, in a non-display area.

Under acf2/MVS Release 4.0 and above, an optional sign-on screen is provided. When the user signs on under any of the short, one-word sign-on formats (e.g., CSSN), ACF2 displays:

```
SYSTEM:        CICS        -- CICS/VS RELEASE 1.6.1 SYSTEM --
TERMINAL:      L43E
NODE:          LV43E

DAY:           Thursday

SYSTEM DATE:   June 05, 1985
SYSTEM TIME:   02:03 PM

LOGON ID:      ===>
PASSWORD:      ===>

NEW PASSWORD   ===>   (protected non-display area)
               ===>


CICS/VS - ACF2 (SYSTEM SIGN ON/OFF FACILITY)
```

After the user enters his Logonid and password, ACF2 validates the Logonid and password. Before the user can proceed, ACF2 also validates the user's authority to access CICS from that particular source at that particular time.

To sign off from CICS, the user can enter the standard CSSF or an installation-defined alternative.

The installation can also modify the CICS sign-on facility to accommodate unique installation requirements.

For further information on CICS sign-on, see the acf2/MVS CICS Support Manual.

IMS SIGN-ON PROCEDURE

If the ACF2/IMS interface has been installed, sign-on to IMS is optional for the ACF2 system. A default Logonid will be used for transaction authorization if sign-on has not been issued from a specific terminal. The IMS system itself can specify which terminals are required to sign-on in order for a user to enter transactions.

When sign-on is required, ACF2 takes control (using an IMS exit) when the /SIGN command is entered:

```
/SIGN logonid password
```

or when changing a password:

```
/SIGN logonid old-password/new-password
```

Note: Your installation may use preformatted screens which require the data to be entered in the designated fields.

IDMS SIGN-ON PROCEDURE

Under ACF2, IDMS user sign-on can be required or handled by ACF2 through an installation-specified default Logonid. ACF2 uses the default Logonid when validating data and resource access requests from terminals that are not associated with a particular user.

The following formats of the ACF2-supported sign-on task are acceptable. Note that, in these formats, the slash (/) literally means a slash symbol:

```
SIGNON logonid{/password[/new-password]}
SIGNON NAME=logonid{,PS=password[/new-password]}
SIGNON {PS=password[/new-password]},NAME=logonid
```

In the formats above:

* logonid is the user's ACF2-defined Logonid.

* password is the user's current ACF2 password. The password is required if the Logonid is entered rather than allowed to default. If you do not enter a password, ACF2 prompts you for one.

* new-password is the new password to replace the current one, provided that the installation has allowed password changes. Entry of a new password is optional.

In these formats, you can substitute blank spaces where commas appear.

After you have entered the Logonid and password, if necessary, ACF2 will continue its validation until the sign-on procedure is completed.

BATCH ENVIRONMENT

ACF2 always validates the first //*LOGONID and //*PASSWORD cards in a batch job stream. In instances where multiple entries are found, the first occurrence of each card is validated whereas subsequent entries are treated as comments. Your batch job is validated via the job entry system (JES2/JES3) by the placement of the following two control cards in your input JCL deck:

```
//*LOGONID your-logonid
```

and

```
//*PASSWORD your-password
```

You may change your password easily by adding the new password in this way:

```
//*PASSWORD old-password/new-password
```

You may alternatively specify your Logonid and password via JOB card parameters as follows:

```
USER=logonid,PASSWORD=password
```

Similarly, you may change your password through the JOB card parameters as follows:

```
USER=logonid,PASSWORD=(old-password,new-password)
```

For the USER= jobcard parameter, the Logonid can only be up to 7 characters.

If the user wishes to run a job under the same Logonid being used during the TSO session, then the Logonid and password need not be submitted with the job.

Also, for all batch job submissions, your installation may optionally direct ACF2 to check that the submitting user has the authority to submit batch jobs. For further details, see the explanation of the JOB field in the chapter on Logonid records. Also, refer to the chapter on GSO records for an explanation of the JOBCK field of the OPTS record, and an explanation of the NJE record.

INTRODUCTION TO THE LOGONID RECORD AND ACF2 USER ATTRIBUTES

The Logonid record is one of the most important ACF2 records. It identifies each individual user on a particular system running ACF2. This identification is done through fields that define a user's attributes.

Not only does this record contain the Logonid and password that allow a user to enter the system, it also contains other information that ACF2 uses to validate the user's authority to access data and resources.

This chapter discusses:

* The structure of the Logonid record

* An example Logonid record

* Some important fields in the Logonid record

* The User Identification (UID) String and its purpose

STRUCTURE OF THE LOGONID RECORD

The ACF2 Logonid record contains much information in various fields. These fields are grouped into sections, as follows:

Identification Section
This section contains information such as the user's Logonid, name, phone, and UID string. The UID string is explained later in this chapter.

Cancel/Suspend Section
This section contains information indicating the status of the Logonid (i.e., whether it is cancelled or suspended and the date this action was taken).

Privileges Section
This section tells ACF2 what the user is allowed to do. Can the user access CICS, IMS, or TSO? Can he submit batch jobs? This section can also provide the authority to process Logonid records, access rules, generalized resource rules, entry records, scope records, shift/zone records, and GSO records.

Access Section

This section contains statistics on the number of system accesses that a user makes, as well as the date, time, and source of the user's last access.

Miscellaneous Section

This section specifies values such as the allowable input source(s) from which a user can access the system, data, and other resources. A field of this section also specifies how often the user must change his password. Other fields in this section contain information pertinent to CICS, IMS, and IDMS security.

TSO Section

This section contains information applicable to TSO users, such as the TSO account number, performance group, region size, etc.

Statistics Section

This section has fields indicating the number of password and security violations pertaining to the user. It also contains the dates of the last password change, last password violation, etc.

EXAMPLE LOGONID RECORD

This section illustrates and explains a sample Logonid record for Nadia Tormell, who is an auditor in an accounting department.

This sample Logonid record is shown with extra blank lines to help define the various sections of the record:

```
USER01           ACCTGAUDUSER01   NADIA TORMELL EXT.413

CANCEL/SUSPEND   EXPIRE(02/02/86)

PRIVILEGES       AUDIT JOB TSO

ACCESS           ACC-CNT(133) ACC-DATE(12/15/85) ACC-SRCE(LV248)
                 ACC-TIME(09:21)

MISCELLANEOUS    DEPT(ACCTG) FUNCTION(AUD) PREFIX(USER01)

TSO              DFT-PFX(USER01) DFT-SOUT(A) DFT-SUBM(A)
                 INTERCOM JCL LGN-SIZE LINE(ATTN) MAIL MSGID
                 NOTICES TSOPROC(IKJACCNT) TSORGN(1,024)
                 TSOSIZE(8,172) WTP

STATISTICS       PSWD-DAT(11/10/85) PSWD-TOD(10/28/85-13:23)
                 PSWD-VIO(1) SEC-VIO(1) UPD-TOD(11/11/85-09:21)
```

This sample illustrates some of the information that can be contained in each section of the Logonid record, as follows:

Identification Section

We see that the user's name is Nadia Tormell, her Logonid is USER01 and her phone number is extension 413. The entry ACCTGAUDUSER01 is her expanded User Identification (UID) string. This installation has defined the UID string as the DEPT field, followed by the FUNCTION field, followed by the Logonid. The values ACCTG, AUD, and USER01 are taken from these fields to form the UID string ACCTGAUDUSER01. Note that the DEPT and FUNCTION fields have been defined by the installation and do not appear in the Logonid record supplied with ACF2.

Cancel/Suspend Section

Nadia's Logonid record is temporary, since it will expire on 02/02/85.

Privileges Section

Nadia has the authority: to list but not change ACF2 rule sets, records, and system options (AUDIT); to run batch jobs (JOB); and to use TSO (TSO).

Access Section.

Nadia has made 133 system accesses. The last access was made at 09:21 on 12/15/85 from a terminal identified as LV248.

Miscellaneous Section

The PREFIX field is the only supplied ACF2 field shown here. The others have been defined by the installation. Nadia's prefix is USER01 (the same as her Logonid). This field gives Nadia ownership of all datasets with a high-level index of USER01. For example she would have ownership over the datasets USER01.WORK.TEXT and USER01.STATS.MASTER. Nadia is in the Accounting department (ACCTG) and her job function is Auditor (AUD).

TSO Section

Nadia's default TSO prefix is the same as her Logonid (USER01). Her default sysout and message classes (DFT-SOUT and DFT-SUBM) are A, and she can receive messages from other TSO users (INTERCOM). She can submit jobs from TSO (JCL), and can specify any region size at logon time (LGN-SIZE). For brevity, the other fields are not explained here but are described in the chapter on the LID setting.

Statistics Section

On 11/10/85 (PSWD-DAT) was the last time that Nadia made an invalid password attempt. On 10/28 85 (PSWD-TOD) was the last time the Nadia changed her password. On 11/10/85, Nadia made only one invalid password attempt (PSWD-VIO). To date, she has had a total of only one security violation (SEC-VIO). Her Logonid record was last updated at 09:21 on 11/11/85 (UPD-TOD).

Note that all seven sections will not necessarily be displayed for every user. For example, if there are no Cancel/Suspend fields active for a given user, the display of that user's Logonid record will not contain the Cancel/Suspend section lines.

You can see that a Logonid record contains a great deal of information about a user. Each field of the Logonid record is described in the chapter on the LID setting.

IMPORTANT FIELDS OF THE LOGONID RECORD

Although all fields of the Logonid record are crucial for tailoring ACF2 to the individual system user, some fields are important to understanding major ACF2 concepts. This section presents these fields. They are discussed in detail in the chapter on the LID setting and in other chapters referenced in this section.

Password-Related Fields

In the Logonid record, a user's password is stored in a one-way encrypted format. This password must be memorized because it cannot be displayed. No one, not even a security officer, can display it.

In addition, ACF2 provides other fields for password control, such as the minimum and maximum password lengths, the maximum period of time during which the same password can be used, and the frequency with which a user can change his password.

ACF2 User Privilege Levels

Certain fields of the Logonid record can grant a user certain privilege levels. Privilege levels give the user certain authorities in terms of access to system data and resources, and access to ACF2 rule sets and records.

These privilege levels are:

ACCOUNT

A user with the ACCOUNT privilege level has the ability to insert, delete, and change Logonid records within the limits defined by the user's scope. The scope is discussed in the chapter on scope records.

ACCOUNT managers are usually given the responsibility of establishing, maintaining, and deleting Logonid records at the installation. They can also use the ACF SHOW subcommands, explained in the next chapter. And they can display and change many of the individual Logonid fields.

The ACCOUNT privilege level grants no authority for writing rules or processing other ACF2 records.

SECURITY

The SECURITY privilege level indicates that a user is an installation security officer. Such a user has the authority to insert, change, list, and delete access and generalized resource rules. That user can list and change certain fields of Logonid records. He cannot insert or delete Logonid records unless he also has the ACCOUNT attribute. He can insert, change, list, and delete any Infostorage records within their scope records. In addition, he can access any dataset and execute programs on the restricted programs list. However, any access granted by the SECURITY privilege level can be restricted through a scope, as explained in the chapter on scope records.

AUDIT

A user with the AUDIT privilege level can list Logonid records, access and generalized resource rules, entry records, shift/zone records, scope lists, and GSO records. This user can also issue the ACF SHOW subcommands, which display ACF2 system control options. However, he does not have authority to modify any of these components of the ACF2 system. He cannot update or delete Logonid records nor access any resources other than those authorized to him via rules.

CONSULT

The CONSULT privilege level is usually given to individuals who assist other users on the computer system. In order to answer questions a user may ask about Logonid record information, a consultant can display some fields of Logonid records, but cannot update them. The fields allowable for display are determined by the installation. By default, someone with the CONSULT privilege level can list nearly all fields of the Logonid record. Some exceptions are PASSWORD, ACCOUNT, and TRACE fields, and those fields that the installation has specifically determined to be nondisplayable to a user with the CONSULT privilege level.

LEADER

The LEADER privilege level is similar to CONSULT. However, the LEADER privilege level provides additional authority for updating selected fields of the Logonid records as specified by the installation.

USER

The USER privilege level is the basic attribute that is automatically assigned to every ACF2 system user. It allows the user to display his own Logonid record. An installation can determine whether a user with only the USER privilege level can write access rules for his own datasets by means of the CENTRAL option of the GSO OPTS record.

Combining Privilege Levels. A user can be given the authorities of more than one privilege level. For example, a user can possess both SECURITY and ACCOUNT, which gives that user all authorities associated with the SECURITY privilege level and all authorities associated with the ACCOUNT privilege level.

Ability to List Logonid Records. In unique situations, an installation may be concerned about what privilege level a user must have in order to list the Logonid record of another user with the same or different privilege level. A subroutine called ACSALTCK is used by ACF2 to make a final determination of whether a user can list another Logonid record. For further information, see the acf2/MVS System Programmer's Guide.

Note: The @CFDE macro can be altered at any installation to change the authority required to display or modify any field.

Scopes: Restricting a User's Authority

Privilege fields (as outlined above) grant users certain authorities to access data, ACF2 rule sets, Logonid records, and other ACF2 records. Scope records restrict that authority.

For instance, a user with the ACCOUNT privilege level has the authority to insert, change, and delete Logonid records. With an assigned scope, a user with the ACCOUNT privilege level can be restricted to insert, change, and delete Logonid records for only a certain group of users.

A scope is associated with a user's Logonid via the SCPLIST field of the Logonid record. For a detailed discussion of scopes, see the chapter on scope records.

OTHER IMPORTANT FIELDS OF THE LOGONID RECORD

Several other fields of the Logonid record can grant special privileges that have major effects on ACF2 validation. An installation should use these fields with care:

DUMPAUTH

This field allows a user to generate a storage dump even when his address space is in an execute-only or program path control environment.

MAINT

An installation's "maintenance" job can be allowed any type of access to a dataset or resource without logging. A specific program must be executed from a specific library. Otherwise, the access is subject to normal ACF2 validation.

MUSASS

This field indicates that the Logonid is for a Multi-User Single Address Space System (MUSASS), such as CICS, IMS, or IDMS. A MUSASS has more authority than a normal user, and can access resources on behalf of its users. For further information, refer to the chapter on MUSASS in the acf2/MVS System Programmer's Guide.

NON-CNCL

This field can allow a user any type of access to a dataset or resource despite any security violations that may occur during the access attempt. The user can access the dataset or resource without logging as long as he is executing a designated program from a designated library. Otherwise, the access is subject to normal ACF2 validation. However, any access that would normally be prevented is allowed but logged.

READALL

This field can allow a user to read all data and execute all programs at an installation, regardless of what access rules may specify. Any read or execute accesses (that violate access rules) are allowed but logged. Other types of access, such as write, are validated just as they are when the user does not have this attribute.

STC (Started Task Control).

This field indicates that the Logonid is for use by started tasks only. Validation of started tasks is determined by the STC option of the GSO OPTS record.

OTHER ACF2 COMPONENTS THAT MAY AFFECT A USER'S ACCESS

Two other ACF2 components may have a major impact on a user's authority as defined in the Logonid record:

1. Global System Options (GSO), one of the major components of ACF2, customize ACF2 security to meet installation-specific needs. The Global System Options are discussed in the chapter on GSO records.

2. The ACF2 Field Definition Record (ACFFDR) defines the more permanent ACF2 system options. It also defines the fields of the Logonid record. For further information, refer to the explanation of the ACFFDR in the acf2/MVS System Programmer's Guide.

THE USER IDENTIFICATION (UID) STRING

A User Identification (UID) string can identify an ACF2 user, as does the user's Logonid. However, when compared to the Logonid, the UID string provides further flexibility. Thus, the UID string is used to identify individual users or groups of users in writing rules for access to data and resources. Also, the UID string is not used to log onto the system.

The UID string is made up of selected fields of the Logonid record. Each installation can select which fields will be used. However, the installation must use the same UID string format for all users. These fields can include the fields supplied with ACF2 or fields defined by the installation.

For example, the following UID string consists of four installation-defined fields plus the user's Logonid:

```
                    MM0244MKTPTH
                    ||| | |
                    ||| | |
Munich (site) ------||| | |
                     || | |
Marketing (division)-|| | |
                      | | |
02 (department) ------| | |
                        | |
44 (function code) -----| |
                          |
MKTPTH (Logonid) ---------|
```

The advantage of the UID string is that it can define subsets of users in ways that the Logonid cannot. The Logonid is only effective in grouping users who have the same beginning characters of their Logonids.

The UID string can allow grouping users by any attributes defined in the Logonid record.

Use of the UID string is discussed in the chapters on ACF2 access rules and generalized resource rules.

THE ACF COMMAND

The ACF command provides subcommands for processing ACF2 rule sets and records. This chapter discusses the basic operation of the ACF command and its subcommands.

ISSUING THE ACF COMMAND THROUGH TSO

When the TSO READY message appears at the terminal, you can issue the ACF command:

```
READY
acf
```

After the system responds with the message ACF, you are ready for processing ACF2 rule sets and records:

```
READY
acf

ACF
```

ACF COMMAND SETTINGS

After issuing the ACF command, you must establish the *ACF command setting.* This setting determines the particular type of ACF2 record you can process. The ACF command has the following settings:

Setting	Type of ACF Record Processed
LID	Logonid records
RULE	Dataset access rule sets
ACF	(A combination of LID and RULE settings. In effect by default when you issue the ACF command.)
RESOURCE	Generalized resource rule sets
ENTRY	Entry records
SCOPE	Scope records
SHIFT	Shift/zone records
CONTROL	ACF2 System Control records including Global System Option (GSO)

IDENTITY Extended User Authentication Records

When the ACF command is active, you can establish the ACF command setting by entering a SET subcommand. For example, to process Logonids, you enter SET LID:

```
acf
set lid
```

For some settings, you must be more specific about the type of ACF2 rule set or record you want to process. Therefore, you must specify the setting and a 3-character type code:

```
acf
set entry(src)
```

The ENTRY setting and type code of SRC are explained in a later chapter.

ACF SUBCOMMANDS

After you have entered the ACF command and have established the appropriate ACF command setting, you can issue the various ACF subcommands. These subcommands allow you to do the actual processing of ACF2 rule sets and records.

The ACF subcommands include:

CHANGE
COMPILE
DECOMP
DELETE
END
HELP
INSERT
LIST
SET
SHOW
SN
STORE
SYNCH
TEST
*

For a given command setting, a subcommand will process a certain type of ACF2 record or rule set. In a particular setting, some subcommands may not be valid.

The following table lists each setting and the ACF subcommands that are valid under that setting:

Setting	ACF2 Component	ACF Subcommands under the Setting
LID	Logonid records	INSERT, CHANGE, LIST, DELETE, SYNCH
RULE	Access rules	COMPILE, TEST, STORE, DECOMP, DELETE LIST
ACF	Logonid records and Access rules	INSERT, CHANGE, LIST, DELETE, SYNCH COMPILE, TEST, STORE, DECOMP, DELETE
RESOURCE	Generalized resource rules	COMPILE, TEST, STORE, DECOMP, DELETE
ENTRY	Entry records	INSERT, CHANGE, LIST, DELETE
SCOPE	Scope records	INSERT, CHANGE, LIST, DELETE
SHIFT	Shift/zone records	INSERT, CHANGE, LIST, DELETE
CONTROL	Global System Option (GSO) records	INSERT, CHANGE, LIST, DELETE, (SHOW operates differently than under other settings.)
IDENTITY	Extended User Authentication Support	INSERT, CHANGE, LIST, DELETE (SHOW operates differently than under other settings.)

The *, END, SET, HELP, SHOW, and SN subcommands are common to all settings.

Guidelines for using the ACF subcommands:

1. The COMPILE, DECOMP, TEST, and STORE subcommands apply only to processing of access and generalized resource rules.

2. The INSERT and CHANGE subcommands do not apply to access or generalized resource rules.

3. The DELETE subcommand applies to all ACF2 records.

The following sections of this chapter explain the command subcommands END, HELP, SET, SHOW, and SN. Other chapters explain each setting and the subcommands shown in the previous table.

COMMON ACF SUBCOMMANDS

The END, HELP, SET, SHOW, and SN subcommands are common to all ACF command settings.

This section discusses these subcommands.

New ACF2 Users. We recommend that you read only about the END and HELP subcommands at this time. An understanding of the other common subcommands is necessary only after subsequent chapters have been read.

* Subcommand--all settings

The asterisk (*) in column one indicates a comment line. The comment line is used primarily in the batch or background TMP environments for documentation purposes. The asterisk in column one and any data on the line is ignored by the ACF command processor.

END Subcommand--all settings

The END subcommand, in most instances, ends the ACF command and causes the redisplaying of the TSO READY message.

Syntax. The END subcommand has the following, simple syntax:

```
     END
```

Ending the TEST Subcommand. Under the RULE and RESOURCE settings, the END subcommand terminates the TEST subcommand. When the TEST subcommand is ended, the ACF command still remains active and in its current setting.

HELP Subcommand--all settings

At any time while the ACF command is active you can issue the HELP subcommand. The HELP subcommand provides online explanations of the setting in which you are processing and the ACF subcommands under that setting.

For example, to view an explanation of an ACF subcommand under the setting you are in, enter HELP and then the subcommand name (e.g., HELP CHANGE). For an explanation of the HELP subcommand itself, simply enter HELP HELP while the ACF command is active.

When the ACF command is not active, the HELP ACF command can be issued from TSO READY mode to provide online information on the ACF command.

The ACF HELP subcommand follows TSO conventions.

SET Subcommand--all settings

The SET subcommand allows you to establish the setting of the ACF subcommand. It also has some additional functions, such as modifying the operation of other subcommands.

Syntax. The SET subcommand has the following syntax:

```
    SET {ACF/LID/RULE/RESOURCE(type-code)/
                      ENTRY(SRC/SGP)/
                      SCOPE(SCP)/
                      SHIFT(SFT/ZON)/
                      CONTROL(GSO/TSO)/
                      TERSE/VERBOSE/
                      FORCE/NOFORCE/
                      TRIVIA/NOTRIVIA/
                      NORULES/
                      NOERROR/
                      MEMBER(nnnnnnn)/
                      IDENTITY(AUT)}
```

Parameters. The SET subcommand parameters ACF, LID, RULE, RESOURCE, ENTRY, SCOPE, SHIFT, IDENTITY, and CONTROL are explained earlier in this section. The SET subcommand under the CONTROL(GSO) setting is described in the chapter on Global System Option (GSO) records. The SET IDENTITY command is described in the Chapter on User Authentication Device support.

Other parameters of the SET subcommand are:

TERSE/VERBOSE

TERSE causes a shortened display of ACF2 records and rule sets via the LIST or DECOMP subcommands. For example:

```
    set terse
    set scope(scp)
    list payscope
     ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 07/15/85-11:43
```

With SET TERSE, only the first line of the Logonid, entry, scope, shift/zone, or control (Global System Option) record is displayed by the LIST subcommand. Only the first and trailing lines of access or generalized resource rule sets are displayed by the LIST and DECOMP subcommands. TERSE output is determined by fields specified in the @HEADER macro of the ACFFDR.

With SET VERBOSE, the normal display of ACF2 rules sets and records occurs:

```
    set verbose
    list payscope
     ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 07/15/85-11:43
     DSN(PAY,TEST,PAYSDH-) LID(PAY-) UID(TFINPAY) INF(SSCPPAY-)
```

Once set, TERSE/VERBOSE remains in effect until it is changed or until the ACF command session is ended. Changing the setting of the ACF command will not affect either TERSE or VERBOSE. VERBOSE is the default when the ACF command is first issued.

FORCE/NOFORCE

FORCE stores an access or generalized resource rule set regardless of whether the rule set already exists. This storing can be the result of the STORE subcommand or automatic storing by the COMPILE subcommand.

NOFORCE prevents storing of an access or generalized resource rule set if the rule set already exists.

Once set, FORCE/NOFORCE will remain in effect until it is changed or until the ACF command is ended. Changing the setting of the ACF command will not affect FORCE or NOFORCE. FORCE is the default when the ACF command is first issued. This parameter is relevant only to the ACF, RULE, and RESOURCE settings.

TRIVIA/NOTRIVIA

TRIVIA allows the normal display of the Logonid record (that is, if TERSE has not been specified). NOTRIVIA causes the displaying of only certain fields of the Logonid record when the LIST subcommand is issued. These fields are determined by the FLAGS=LIMIT parameter of the @CFDE macro, explained in the acf2/MVS System Programmer's Guide.

Once set, TRIVIA/NOTRIVIA remains in effect until it is changed or until the ACF command session is ended. Changing the setting of the ACF command will not affect TRIVIA or NOTRIVIA. TRIVIA is the default when the ACF command is first issued. These parameters are relevant only to the ACF and LID settings.

NORULES

The NORULES parameter causes currently held access rules to be cleared from the user's address space. After storing access rules, a user can use this subcommand to make the newly stored access rules effective for subsequent access validation. Another alternative is for the user to log off, and then log on again.

NOERROR

The NOERROR parameter resets the error indication so that the maximum return code of 4 is returned when the ACF command is ended.

MEMBER(nnnnnnn)

The MEMBER parameter is used for determining ACF2-generated member names for partitioned datasets.

Whenever a generalized resource rule set is decompiled into a partitioned dataset (PDS) and no member name is specified, ACF2 uses the key of the decompiled rule set to determine the member name. For example, a generalized resource rule set with the key PAYTRN would be compiled into the member PAYTRN.

However, that key may form an invalid member name, particularly when the key is masked. In such a case, ACF2 generates a member name by taking the rightmost five digits of the value of the MEMBER parameter, incrementing the value of these digits by 1, and then preceding the result by an @ symbol. For example, if the value of this parameter is 00003, the ACF2-generated name to replace any invalid member name will be @00004.

The most recently used ACF2-generated member name is stored. It is incremented by 1 to form the next ACF2-generated member name unless the MEMBER parameter is respecified in the meantime. This parameter must be specified with a number from 0 to 9999999.

SHOW Subcommand--all settings

The SHOW subcommand, under any setting, lists information about ACF2 as it is currently running on your system.

Syntax. The SHOW subcommand has the following syntax:

```
SHOW ACF2/ACTIVE/ALL/DDSN/FIELDS/LINKLST/MODE/PROGRAMS/RESIDENT/
     STATE/SYSTEMS/TSO/ZEROFLDS
```

Parameters. The SHOW subcommand accepts only one parameter at a time. Samples of the SHOW subcommand with various parameters are shown on the following pages:

SHOW ACF2 or SHOW ALL

SHOW ACF2 or SHOW ALL gives you the comprehensive result of issuing separate SHOW subcommands with the parameters ACTIVE, DDSN, LINKLST, PROGRAMs, RESIDENT, STATE, SYSTEM, TSO, ZEROFLDS, and APPLDEF. (These are all possible parameters except for FIELDS and MODE.)

SHOW ACTIVE

SHOW ACTIVE displays the ACF2 intercepts which have received control (denoted by YES). Also displayed are any local exits specified in the EXIT GSO record. This record is described in the chapter on GSO records.

```
acf
show active

 -- ACF2 INTERCEPTS THAT HAVE RECEIVED CONTROL --
 DASD-OPEN(YES)             DASD-EOV(NO)               VSAM-OPEN(YES)
 TAPE-OPEN(YES)             TAPE-EOV(NO)               CATALOG(YES)
 DASD-ALOC(YES)             DASD-RENAME(NO)            DASD-SCRATCH(NO)
 USER CALL(NO)              EXTERNAL CALL(NO)          PROGRAM CALL(YES)
 JOB INIT(YES)              JOB/STEP TERM(YES)         TSO-MVS(YES)
 CAT-CVOL(NO)               READER-VS1(NO)             INTERP-VS1(NO)
 NONVSAM-ERASE(YES)         VSAM-ERASE(YES)

 -- LOCAL EXITS SPECIFIED ON THIS SYSTEM --
 DSN PRE-VALIDATE=ABCVALD (INACTIVE)    DSN POST-VALIDATE=POSTVLD (INACTIVE)
 DSN VIOLATION=NONE                     PSEUDO DSN GENERATE=NONE
 RSRC PRE-VALIDATE=NONE                 RSRC POST-VALIDATE=NONE
 STC VALIDATE=NONE                      SOURCE MODIFICATION=NONE
 LOGON PRE-VALIDATE=NONE                LOGON POST-VALIDATE=NONE
 PASSWORD EXPIRATION=NONE               NEW PASSWORD=NONE
 RULE DB PRE-PROCESS=NONE               RULE DB PST-PROCESS=NONE
 INFO DB PRE-PROCESS=NONE               INFO DB PST-PROCESS=NONE
 SVC INITIALIZATION=NONE                TSO LOGON TERM TYPE=NONE
 TSO LOGON PARM=NONE

 -- ACF2 TRACE FACILITY --

 GSO TRACE OPTION=OFF

 -- ACF2 SAF INTERFACE --

 SAF VALIDATION OPTION=ON
 SAF TRACE OPTION=OFF

 -- AUTHENTICATION EXITS ON THIS SYSTEM:  LIDFLD/PROCESS PROGRAM/INFOSTG --
 OID                 ACFEAOID            INFOSTG
```

SHOW DDSN

SHOW DDSN lists 'the dataset names in use for the Rule, Logonid, and Infostorage databases. Also listed are the backup datasets of these databases if allocated. If a dynamic dataset name (DDSN) list was specified or defaulted at ACF2 startup, any dataset names preallocated but different from the name in the DDSN list are flagged with an asterisk and a note.

```
acf
show ddsn
-- ACF2 DYNAMIC DATASET NAMES SPECIFIED --
DDSN PRIMARY DEFAULTED AT STARTUP.  DSNS IN USE ARE:
         RULES    FDR ACFSYS.ALTRULES
         INFOSTG  FDR ACFSYS.ALTINFO
         LOGONIDS FDR ACFSYS.ALTLIDS
         BACKRULE NOA NOT ALLOCATED
         BACKINFO NOA NOT ALLOCATED
         BACKLID  PRE ACFSYS.BKLIDS

DDSN LISTS DEFINED IN FDR ARE:
PRIMARY  RULES=SYS1.ACF.RULES
         LOGONIDS=SYS1.ACF.LOGONIDS
         INFOSTG=SYS1.ACF.INFOSTG
         BACKRULE=SYS1.ACF.BKRULES
         BACKLID=SYS1.ACF.BKLIDS
         BACKINFO=SYS1.ACF.BKINFO

ALT      RULES=SYS1.ACF.ALTRULES
         LOGONIDS=SYS1.ACF.ALTLIDS
         INFOSTG=SYS1.ACF.ALTINFO
         BACKRULE=SYS1.ACF.ABKRULES
         BACKLID=SYS1.ACF.ABKLIDS
         BACKINFO=SYS1.ACF.ABKINFO
```

The middle column in the section where "DSNS in Use" is specified indicates where the dataset is defined:

FDR = ACFFDR
PRE = pre-allocation by installation
NOA = not allocated or defined before ACF2 startup

SHOW FIELDS

SHOW FIELDS will display all Logonid field names that the user issuing the command can display or modify. Fields that are modifiable by that user are prefixed by an asterisk (*). For example:

```
 acf
 show fields

  -- IDENTIFICATION --
   LID        NAME      *PASSWORD   PHONE      UID
  -- CANCEL/SUSPEND --
   CANCEL     CSDATE     CSWHO      MON-LOG    MONITOR    PSWD-EXP   SUSPEND
   TRACE      TSO-TRC
  -- PRIVILEGES --
   ACCOUNT    ACTIVE     AUDIT      AUTODUMP   CICS       CONSULT    DSNSCOPE
   DUMPAUTH   EXPIRE     IDMS       IMS        JOB        JOBFROM    LEADER
   LIDSCOPE   LOGSHIFT   MAINT      MUSASS     NO-SMC     NO-STORE   NON-CNCL
   PGM        PROGRAM    READALL    REFRESH    RESTRICT   RULEVLD    SCPLIST
   SECURITY   SRF        STC        SUBAUTH    TAPE-BLP   TAPE-LBL   TSO
   UIDSCOPE   USER       VM         VSESRF
  -- ACCESS --
   ACC-CNT    ACC-DATE   ACC-SRCE   ACC-TIME
  -- MISCELLANEOUS --
   AUTHSUP1   AUTHSUP2   AUTHSUP3   AUTHSUP4   AUTHSUP5   AUTHSUP6   AUTHSUP7
   AUTHSUP8   CICSCL     CICSID     CICSKEY    CICSKEYX   CICSPRI    CICSRSL
   IDLE       IDMSPROF   IDMSPRVS   MAXDAYS    MINDAYS    MUSOPT     MUSPGM
   PREFIX     SHIFT      SOURCE     ZONE
  -- TSO --
   ACCTPRIV   ALLCMDS    ATTR2     *CHAR       CMD-LONG   DFT-DEST  *DFT-PFX
  *DFT-SOUT  *DFT-SUBC  *DFT-SUBH  *DFT-SUBM  *INTERCOM   JCL        LGN-ACCT
   LGN-INDX   LGN-MSG    LGN-PERF   LGN-PROC   LGN-RCVR   LGN-SIZE   LGN-TIME
   LGN-UNIT  *LINE      *MAIL      *MODE       MOUNT     *MSGID     *NOTICES
   OIDOLD     OIDOLD-A   OPERATOR  *PAUSE      PMT-ACCT   PMT-PROC  *PROMPT
  *RECOVER    TSOACCT    TSOCMDS    TSOFSCRN   TSOPERF    TSOPROC    TSORBA
   TSORGN     TSOSIZE    TSOTIME    TSOUNIT    UADSINDX   VLD-ACCT   VLD-PROC
  *WTP
  -- STATISTICS --
  *PSWD-DAT   PSWD-TOD  *PSWD-VIO  *SEC-VIO    UPD-TOD
```

Under the CONTROL(GSO) setting, the SHOW subcommand with the FIELDS parameter has the following syntax:

```
SHOW FIELDS(record-name)
```

For an explanation of the SHOW subcommand under the CONTROL(GSO) setting, see the chapter on Global System Option (GSO) records.

SHOW LINKLST

SHOW LINKLST displays the names of the libraries the installation has specified as SYS1.LINKLIB. The names can be found in the GSO LINKLIST record. It is used to provide flexibil'ty to program pathing functions. Libraries are used in conjunction witn program names in access rule validation. For further information, see the explanation of the LINKLST record in the chapter on GSO records.

For example:

```
acf

show linklst
-- DATASETS INCLUDED IN THE LINKLIST --
SYS1.LINKLIB
PAYROLL.WORK.LOAD
```

SHOW MODE

SHOW MODE displays the current setting or "mode" of the ACF command. When issued, the ACF command is initially in the ACF setting by default:

```
acf
show mode
 MODE: ACF
```

SHOW PROGRAMS

SHOW PROGRAMS displays information on the following system options that the installation has established for program name control:

Restricted program names
SHOW PROGRAMS lists the names of those programs that bypass the operating system integrity. Execution of these programs is allowed only to users with the SECURITY privilege level and unlimited scope, or users with the NON-CNCL privilege. See the explanation of the SECURITY and NON-CNCL fields in the chapter "The Logonid."

Maintenance Logonids/programs/libraries
SHOW PROGRAMS lists the Logonids for each user allowed to bypass access rule validation when executing the specified program from a specified library. The MAINT or NON-CNCL privileges are required in the Logonid.

TAPE bypass label programs/libraries
SHOW PROGRAMS list the names of programs that, when executed from a specified library, are valid for tape Bypass Label Processing (BLP).

Logged programs
SHOW PROGRAMS lists the names of programs for which each dataset access will be logged.

```
acf
show programs

  -- RESTRICTED PROGRAM NAMES --
  DRWD**** FDR***   ICKDSF** IEHD****  IEHINIT*

  -- MAINTENANCE LOGONIDS/PROGRAMS/LIBRARIES --
  MAINTLID MAINTPGM SYS1.LINKLIB
  MAINTLID MAINTPG1 SYS1.LINKLIB
  MAINTLID MAINTPG2 SYS1.LINKLIB
  MAINTLID MAINTPG3 SYS1.LINKLIB
  MAINTLID MAINTPG4 SYS1.LINKLIB

  -- NO TAPE BYPASS LABEL PROGRAMS/LIBRARIES --

  -- LOGGED PROGRAMS --
  AMASPZAP
  IMASPZAP
  INCORZAP
```

Note that if no programs exist under a certain category, SHOW PROGRAMS will indicate that no such programs exist.

For further information on these program controls, refer to the explanations of the PPGM, MAINT, BLPPGM, and LOGPGM records in the chapter on GSO records.

SHOW RESIDENT

SHOW RESIDENT displays the names of system resident resource directories and access rules. For example:

```
acf
show resident

 -- RESIDENT DIRECTORIES --
 NONE SPECIFIED FOR THIS SYSTEM

 -- RESIDENT ACCESS RULES --
   PAY          ABC          SYS1
```

SHOW STATE

SHOW STATE displays the ACF2 system options in effect. For example:

```
 acf
 show state

  RUNNING ACF2 REL 4.1.0/MVS;  SP2.1.2;  WITH MODE = RULE, ABORT, ABORT
  USING FDR ASSEMBLY:  15.26  08/08/85

  OPTIONS IN EFFECT:
  TAPE BLP=LOG               CONTROL=DECENTRALIZED      %CHANGE=ALLOWED
  CPUTIME=LOCAL              DATE FORMAT=MM/DD/YY       STC DFLT LID=ACFSTCID
  DEFAULT LID=ABCDFT         JOB CHECK=NO               MAX VIO PER JOB=10
  STC OPTION=ON              TAPE DSN=YES               UADS=BYPASS
  NOSORT=YES                 NONVSAM-ERASE=NO           VSAM-ERASE=NO

  PASSWORD OPTIONS IN EFFECT:
  LOGON RETRY COUNT=2        MIN PSWD LENGTH=5          MAX PSWD ATTEMPTS=3
  PSWD ALTER=YES             PSWD FORCE=YES             PSWD-JES=ON
  PSWD WARN DAYS=3           PSWD ALGORITHM USED=XDES

  UID STRING = COMPANY,SITE,LEVEL,PROJECT,LID,IDNUM

  DECOMP AUTHORITY = SECURITY, AUDIT

  INFO LIST AUTHORITY = SECURITY, AUDIT

  VOLUME PSEUDO DSN = @VOLSER.VOLUME

  -- DSNAME PROTECTED VOLUMES --
  ******
  -- VOLSER PROTECTED VOLUMES --
  ******
  -- NO VOLSER PROTECTED VOLUMES --

  -- NO AUTOMATIC ERASE VOLUMES --
```

For further information on these options, refer to the explanations of the PSWD, OPTS, RESVOLS, and SECVOLS records in the chapter on GSO records. Also, refer to the explanation of the @UID macro of the ACF2 Field Definition Record (ACFFDR) in the acf2/MVS System Programmer's Guide.

SHOW SYSTEMS

SHOW SYSTEMS displays various system parameters, such as the ACF2 SVC numbers, and SMF record numbers. For example:

```
 acf
 show systems

  -- SYSTEM PARAMETERS IN EFFECT --

  SVCS:
  ALTER SVC=222             VALIDATE SVC=221

  SMF RECORD NUMBERS:
  PASSWORD=220              DATASET VIO=221           LID JOURNAL=222
  RULE JOURNAL=223          LID TRACE=224             TSO COMMAND=225
  INFO JOURNAL=226          RESOURCE VIO=227          ACF2 COMMON=230

  BACKUP:
  AUTO BACKUP TIME=03.30    CPU-ID=SKK1
  WORK FILE UNIT=VIO        PRIMARY SPACE=005         SECONDARY SPACE=005
  STRING=S REPROALT

  NJE OPTIONS IN EFFECT:
  VALIDATE OUT =YES         VALIDATE IN =YES          INHERIT =YES

  OTHER:
  CONSOLE MSGS=ROLL         SHR-DASD=SUPPORTED        SMF LOGONID STAMP=NO
  JES2-XBM=NO VALIDATE      LOGONID LENGTH = 1024     LAB NUMBER=       5
  LABEXP=  01:05:00         NOTIFY=YES                CURRENT SYSID=ABC1
  STARTUP SYSID=ABC1        BUILT ACCVT=ABC1
```

Further information on the @CSVC macro, can be found in the acf2/MVS System Programmer's Guide.

SHOW APPLDEF

SHOW APPLDEF displays the Extended User Authentication routines in effect:

```
acf
SHOW appldef

-- USER APPLICATION DEFINITIONS:  CLASS-TYPE/DIVISION/RSB-MODULE/RECID --
/- AUT/OID/ACFOIRSB/********
```

SHOW TSO

SHOW TSO displays TSO default options on the system:

```
 acf
 show tso

  -- TSO RELATED DEFAULTS ACTIVE --
  LOGON ACCOUNT STRING=1
  CMD LIST BYPASS CHAR=#     CHAR DELETE CHAR=NONE      TSO CMD LIST=NONE
  COMMAND SMF RECORDS=NO     LINE DELETE CHAR=NONE      LOGON CHECK=NO
  PERFORMANCE GROUP=NONE     TSO LOGON PROC=IKJACCNT    QUICK LOGON=YES
  TSO REGIONSIZE=1024        SUBMIT CLASS=NONE          SUBMIT HOLD CLASS=NONE
  SUBMIT MSGCLASS=NONE       SESSION TIME=0             SYSOUT CLASS=A
  TSO UNITNAME=SYSDA         LOGON WAIT TIME=60         FSRETAIN=YES
```

SHOW ZEROFLDS

SHOW ZEROFLDS displays those fields of the Logonid record that cannot be copied by the ACF subcommand INSERT USING (under the LID setting). For example:

```
 acf
 show zeroflds

  -- FIELD VALUES WHICH WILL NOT BE COPIED DURING 'INSERT USING' PROCESSING --
  PASSWORD    PSWD-TOD    NAME        PHONE       UPD-TOD     SEC-VIO
  PSWD-VIO    PSWD-DAT    ACC-DATE    ACC-TIME    ACC-CNT     ACCTPRIV
  OPERATOR    NON-CNCL    MOUNT       NO-SMC      MUSASS      JOBFROM
  ACC-SRCE    TSORBA      SECURITY    LEADER      CONSULT     AUDIT
  ACCOUNT     SCPLIST     LOGSHIFT    READALL     RULEVLD     SHIFT
  ZONE        AUTHSUP1    AUTHSUP2    AUTHSUP3    AUTHSUP4    AUTHSUP5
  AUTHSUP6    AUTHSUP7    AUTHSUP8    REFRESH     MAINT       UADSINDX
  OIDOLD      OIDOLD-A    OID         MAINT
```

To alter the fields included on this list, refer to the explanation of the @CFDE macro of the ACF2 Field Definition Record (ACFFDR) in the acf2/MVS System Programmer's Guide.

SN Subcommand--all settings

This subcommand interfaces with the TSO SEND command. The syntax is:

```
SN  'message' {[USER(*/logonid1,logonid2,...,logonidn)] -
    [OPERATOR(2)] [OPERATOR(route-code)] -
    [CN(console-id)]} {NOW/LOGON/SAVE} {NOWAIT/WAIT}
```

You can issue this subcommand under any setting of the ACF command. For an explanation of SN subcommand parameters, refer to the description of the TSO SEND command in your TSO command language reference manual.

THE ACF COMMAND IN BATCH

An MVS installation can execute the ACF command in batch by using a utility called ACFBATCH. In addition to executing the ACF command, the user can execute ACF subcommands, such as INSERT, CHANGE, LIST, and DELETE. For example, an authorized user can establish a new Logonid record with a jobstream such as:

```
//ACFJOB    EXEC   PGM=ACFBATCH
//SYSPRINT  DD     SYSOUT=A
//SYSHELP   DD     DSN=SYS1.HELP,DISP=SHR
//SYSIN     DD     *
SET LID
INSERT USING(PAYSEC) PAYJSD NAME(JANE S. DOE) LEADER -
 PHONE(EXT. 458) TSO
/*
```

The same facility is also available through the execution of the Terminal Monitoring Program (TMP) in background. For example:

```
//ACFJOB    EXEC   PGM=IKJEFT01,DYNAMNBR=25
//SYSTSPRT  DD     SYSOUT=A
//SYSHELP   DD     DSN=SYS1.HELP,DISP=SHR
//SYSTSIN   DD     *
ACF
SET LID
INSERT USING(PAYSEC) PAYJSD NAME(JANE S. DOE) LEADER -
 PHONE(EXT. 458) TSO
/*
```

For further information on the ACFBATCH program, refer to the acf2/MVS Utilities Manual.

THE ACF COMMAND VIA ISPF SCREENS

Most ACF2 processing can be done through IBM's Interactive System Productivity Facility (ISPF). For further information, refer to the general information section of the acf2/MVS Utilities Manual.

THE ACFM CICS TRANSACTION

The ACF2/CICS interface provides a transaction called ACFM, which allows maintenance of ACF2 Logonid records in CICS conversational mode. This transaction provides a function called Command Processor (CP). The CP function allows an authorized user to enter subcommands under CICS similar to the ACF subcommands used for Logonid record processing under TSO.

For further information on Logonid maintenance through the ACFM transaction, refer to the acf2/MVS CICS Support Manual.

THE ACF/IMS TRANSACTION

The ACF2 interface to IMS provides an ACF conversational transaction to allow maintenance of Logonid records directly from an IMS terminal. A user must be signed-on via the /SIGN command to use the ACF/IMS transaction. Enter 'ACF' to enter the IMS conversational transaction. Then the SET, INSERT, CHANGE, DELETE, LIST, or END subcommands may be used.

These subcommands are used in the same way under the ACF/IMS transaction as under the TSO ACF command. Refer to the chapter in this manual entitled "THE LOGONID - Defining Users to ACF2" for detailed descriptions of these subcommands. Additionally, you may refer to the acf2/MVS IMS Support Manual for detailed discussions of the ACF/IMS transaction.

LID SETTING: LOGONID RECORDS

The Logonid record describes the attributes of a system user. These attributes allow ACF2 to validate each user individually upon each request for access to data and resources.

This chapter discusses:

1. Fields of the Logonid record

2. Creating a Logonid record

3. Changing a Logonid record

4. ACF Subcommands Under the LID Setting

FIELDS OF THE LOGONID RECORD

The various fields of the Logonid record are grouped into sections, as mentioned earlier.

The following list organizes the fields of the Logonid record into their respective groups. It also provides a description of each field.

Identification Section--GROUP 0

LID
Contains the Logonid of the user. This Logonid is also used for identifying the Logonid record. (8 characters)

NAME
Contains the name of the user. This name is displayed on ACF2 logging and security violation reports. If the NOUADS field is specified in the OPTS GSO record, the NAME field will also be used as the NAME field of the JOB card created for a TSO logon session. (20 characters)

PASSWORD
Contains the password of the user. This is stored by ACF2 in a one-way encrypted format. For further information on encryption algorithms, see the acf2/MVS System Programmer's Guide. (8 characters)

PHONE
Contains the telephone number of the user. (12 characters)

UID
A pseudo field (not actually stored in the Logonid record) that contains the User Identification (UID) string for a user. This field is a concatenation of selected Logonid fields which may include installation-defined fields such as department and job function. Which fields of the Logonid record are used for a given installation are defined via the ACF2 Field Definition Record @UID Macro.

Cancel/Suspend Section--GROUP 1

CANCEL
Indicates that the Logonid has been cancelled and cannot be used to access the system. ACF2 does not differentiate between CANCEL and SUSPEND (described below), except as to who can alter or display the fields as defined in the Field Definition Record. Each installation may want to establish local procedures for the use of these two fields. (BIT field)

CSDATE
Specifies the date that the CANCEL, SUSPEND, MON-LOG, or MONITOR field was set for this user. (This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE field of the GSO OPTS record. For further information on this field, see the chapter on GSO records.

CSWHO
Indicates which user (Logonid) set the CANCEL, SUSPEND, MON-LOG, or MONITOR field for this user. (8 characters)

MON-LOG
Indicates that an SMF record will be written (for the Invalid Password/Authority Log) each time this user enters the system. No messages are sent. (BIT field)

MONITOR
Indicates that a message is to be sent to the security console and to a designated person each time this user enters the system. The CSWHO field in the Logonid record of the user entering the system indicates who the designated person is. (Bit field)

PSWD-EXP
Indicates that this user's password has been manually (forced) expired. This attribute allows a security officer to force this user to change his password. (BIT field)

SUSPEND
Indicates that the Logonid has been suspended and cannot be used to access the system. See comments on CANCEL above. (BIT field)

TRACE
Indicates that all dataset and resource references made by this user are to be traced via SMF for the Dataset Report Log (ACFRPTDS) and Resource Logging Report (ACFRPTRV) report generators. For details on trace records collected for these two report generators, see the acf2/MVS Utilities Manual. (BIT field)

TSO-TRC
Indicates that all TSO commands issued by this user are to be traced via SMF for the Command Statistics Report (ACFRPTCR). For further information, see the explanation of the ACFRPTCR report generator in the acf2/MVS Utilities Manual. (BIT field)

Privileges Section--GROUP 2

ACCOUNT
Indicates the authority to INSERT, DELETE, and CHANGE Logonid records. This authority can be restricted through a scope, as discussed in the chapter on scope records. A user with ACCOUNT only or SECURITY only privilege cannot list or change a Logonid record of a user who has both ACCOUNT and SECURITY, as the user with both is potentially more powerful than a user with only one of these two authorities. (BIT field)

ACTIVE
Automatically activates the Logonid one minute after midnight on the date contained in this field. The date can be specified as mm/dd/yy, yy/mm/dd, or dd/mm/yy depending upon the installation's date option in the GSO OPTS record. The Logonid cannot be used until the specified date.

AUDIT
Indicates the ability to inspect, but not modify, the parameters of the ACF2 system. (BIT field)

AUTODUMP
Indicates that ACF2 is to take an SVC dump whenever a dataset or resource violation occurs. Should be used for debugging only. (BIT field)

CICS
Indicates whether the user has the authority to sign on to CICS. This is the default field that ACF2 checks for CICS authority. By changing the CICS initialization parameters, an installation can choose to use a different field for one or more CICS regions. For further information, see the chapter on ACF2/CICS parameters in the acf2/MVS CICS Support Manual. (BIT field)

CONSULT

Indicates the authority to display other Logonid records. This authority is generally restricted through a scope, as discussed in the chapter on scope records. (BIT field)

DSNSCOPE

A mask used to limit the scope of a security officer as applies to dataset references (particularly rule compiling or decompiling), or, if an account manager, for setting Logonid PREFIX values when creating new Logonid records. If the SCPLIST field of a user's Logonid record is specified, then this field will not have any effect. All installations should convert to using the SCPLIST field instead of this one. This field will be removed in a future release of acf2/MVS. (8 characters)

DUMPAUTH

Specifies that this user is allowed to generate a dump even when his address space is in an execute only or path control environment. If the Logonid is in that state without this attribute, no dumps will be permitted with the exception of dumps in a non-program pathing environment. (BIT field)

EXPIRE

Indicates Logonid expiration date for "temporary" Logonids. When the specified date is reached, the user will no longer be able to logon or submit jobs. User will receive a 'LOGONID EXPIRED' message. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE field of the OPTS GSO record. For further information on the DATE option, see the chapter on GSO records. To deactivate this expiration date, change the Logonid record and specify the EXPIRE(0) parameter with the CHANGE subcommand.

IDMS

Indicates whether the user has the authority to sign on to IDMS. This is the default field that ACF2 checks for IDMS authority. By changing the IDMS @MOPT macro, an installation can choose to use a different field. See the chapter on ACF2/IDMS option selection in the acf2/MVS IDMS Support Manual. (BIT field)

IMS

Indicates the privilege to use IMS. This is the default field that ACF2 checks for IMS authority. Through the IMS AUTH operand, an installation can choose to use another field. Refer to the explanation of the AUTH operand in the chapter on ACF2/IMS parameter selection in the acf2/MVS IMS Support Manual. (BIT field)

JOB

Indicates the ability to enter jobs (background/batch jobs). Only checked if the JOBCK field of the GSO OPTS record has been specified. For further information on the JOBCK field, see the chapter on GSO records. (BIT field)

JOBFROM

Indicates permission to use //*JOBFROM control cards, primarily for MUSASS support. This control card causes the Logonid and source to be transmitted by the MUSASS for any jobs submitted via that MUSASS. (BIT field)

LEADER

Indicates the authority to display and alter certain fields of Logonid records for other users. This authority is generally restricted through a scope, as discussed in the chapter on scope records. (BIT field)

LIDSCOPE

A mask used to limit which Logonids a user's special authorizations (SECURITY, ACCOUNT, AUDIT, LEADER, or CONSULT) apply to. If the SCPLIST field of a user's Logonid record is specified, then this field will not have any effect. All installations should convert to using the SCPLIST field instead of this one. This field will be removed in a future release of acf2/MVS. (8 characters)

LOGSHIFT

Indicates the privilege to access the system outside of the time period specified in the SHIFT field of the Logonid record. All such system accesses are logged by the ACFRPTPW report generator. (BIT field)

MAINT

Indicates that this user may access resources, without ACF2 rule validation or loggings, by means of a specified program executed from a specified library. This program and library must be defined in the Infostorage database. For further information, see the explanation of the MAINT record in the chapter on GSO records. (BIT field)

MUSASS

Indicates a Multi-User Single Address Space Logonid, such as CICS, IMS, or IDMS. (BIT field)

NO-SMC

Indicates the permission to bypass Step-Must-Complete controls; a job will be considered Non-Cancellable for the duration of the sensitive VSAM update operation. For MUSASS support. (BIT field)

NO-STORE

Indicates that this user may not store rule sets regardless of whether this user owns the datasets related to the rule set, has the SECURITY privilege level, or has been delegated change authority through a CHANGE or RCHANGE control card in the rule set. (BIT field)

NON-CNCL

Indicates that this user will not be cancelled by ACF2 for security violations. The event log will show that the request was allowed because the user was Non-Cancellable. (BIT field)

PROGRAM

Specifies a program name or name mask. The specified APF authorized program(s) must be used to submit jobs for this Logonid. Proper use of this "program-pathing" facility also requires that this Logonid be defined with the RESTRICT and SUBAUTH attributes. This field can also be specified by using the field name PGM instead of the name PROGRAM, except for reports which require the full form of the name (PROGRAM). (8 characters)

READALL

Indicates this Logonid has READ and EXECUTE access to all datasets at the installation. This is similar to the NON-CNCL attribute, but grants READ and EXECUTE access only and enforces any existing rules for other types of access. (BIT field)

REFRESH

Indicates that this user is authorized to issue the F ACF2,REFRESH operator command from the security console. The F ACF2,REFRESH command allows an installation to apply to the system all or selected changes to GSO records. For further information on this command, see the chapter on GSO records. (BIT field)

RESTRICT

Specifies a restricted Logonid for production use. This Logonid does not require a password for user verification. Jobs submitted under this Logonid will be logged and listed on the Restricted Logonid Job Log report. See the explanation of the ACFRPTJL report generator in the acf2/MVS Utilities Manual. This field is not applicable to online user Logonids. (BIT field)

RULEVLD

Indicates that an access rule must authorize any dataset accesses that this user makes. This attribute applies even if the user has ownership of the data or has the SECURITY privilege level. (BIT field)

SCPLIST

Specifies the name of the scope record that restricts accesses for this privileged user (e.g., SECURITY, ACCOUNT). For further information, see the chapter on scope records. (8 characters)

SECURITY

Indicates that this user is a security officer. A security officer has the authority to create, maintain, and delete access rules, generalized resource rules, entry records, scope records, shift/zone records, and Global System Option (GSO) records. This user can also change certain fields in Logonid records, display Logonid records, and access any datasets. ACF2 logs any accesses that this security officer makes that are not allowed through ownership or through access rules. Any authority granted through the SECURITY privilege level can be restricted through a scope, as described in the chapter on scope records. (BIT field)

STC

Indicates that this Logonid is for use by started tasks only. Use of a Logonid without this attribute by a started task will be denied; likewise, use of a Logonid with this attribute by a job or TSO session will also be denied. (BIT field)

SRF

Indicates the user is authorized to use the System Resource Facility (SRF) facility in a VM environment. The field is meaningful to VM installations. It is specified on the Logonid record for installations using Shared Database Support for MVS and VM.

SUBAUTH

Indicates that jobs specifying this Logonid may only be submitted via APF-authorized programs. The RESTRICT attribute must be granted for a Logonid before the SUBAUTH attribute can be used effectively. Normally, the SUBAUTH attribute is also used in conjunction with the PROGRAM attribute. (BIT field)

TAPE-BLP

Indicates that this user may use full Bypass Label Processing (BLP) when accessing tape datasets. When the user has the TAPE-BLP privilege and is accessing tape through BLP, ACF2 allows the access. The checking is normally done by rule validation on the dataset name as coded in the JCL. This privilege should be tightly controlled. (BIT field)

TAPE-LBL

Indicates that this user has Limited Bypass Label Processing authority when using tapes. When a BLP access request is made under this privilege, ACF2 validates the actual volume serial number written on the tape label (if available), checks for a VOLUME rule, and validates any dataset name (DSN) specified in the JCL. This tape dataset validation is dependent upon the TAPEDSN field of the OPTS GSO record and whether the volser is on the SECVOLS GSO record. See the explanation of that record in the chapter on GSO records. (BIT field)

TSO

Indicates that the user is authorized to log on to TSO. The installation has the option of determining whether ACF2 will check the user's TSO logon authorization. For information on this option, refer to the explanation of the LOGONCK field of the TSO record in the chapter on Global System Option (GSO) records. (BIT field)

UIDSCOPE

Specifies a UID mask that determines which Logonid records can be displayed or modified by this user. This mask serves to limit the scope of the SECURITY, ACCOUNT, AUDIT, LEADER, and CONSULT privileges. If the SCPLIST field of a user's Logonid record is specified, then UIDSCOPE will no longer have any effect. All installations should convert to using the SCPLIST field instead of this one. This field will be removed in a future release of acf2/MVS. (24 characters)

USER
All Logonids defined to ACF2 are automatically assumed to have the USER attribute. This field is never displayed and should not be altered by anyone. (BIT field)

VM
Indicates that the user is authorized to log on to VM. This field is meaningful for installations that also have the acf2/VM product and are using Shared Database Support. (Bit field)

Access Section--Group 3

ACC-CNT
The count of the number of system accesses made by this Logonid since it was created. (4 byte binary)

ACC-DATE
The date of the last system access by this user. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE field of the OPTS record. For further information, see the chapter on Global System Option (GSO) records.

ACC-SRCE
The address of the input device from which this user last entered the system. Only applicable if the installation option for the length of Logonid records is 1024 bytes. For further information on the Logonid record length, see the explanation of the LRECL field of the OPTS record in the chapter on Global System Option (GSO) records. (8 characters)

ACC-TIME
The time of the last system access by this user. The display format is hh.mm. (4 byte binary)

Miscellaneous Section--Group 4

*AUTHSUP1 through AUTHSUP8
Logonid record attributes will control the activation of extended authentication validation for each designated system user. This provides the flexibility to require some users to be processed for additional authentication beyond the normal ACF2 Logonid and password validation, and allows other users to signon without further user authentication. An installation may utilize up to eight different types of authentication routines within the installation, but only one may be designated for any one user.

Since there is no way for ACF2 to anticipate what actual authentication routines might be selected by an installation, the new Logonid attributes (e.g., AUTHSUP1) have somewhat generic names. An installation does have the option of changing the @CFDE macro external entry name to a name more closely tied to the device name or application chosen by the installation. Currently these routines are available for TSO logon validation only. (Bit fields)

CICSCL
Indicates CICS Operator Class. For CICS support. (3 characters)

CICSID
Indicates CICS Operator id. For CICS support. (3 characters)

CICSKEY
Contains transaction security key values for CICS Release 1.5 support. Contains the first three bytes of transaction security key values to support CICS Release 1.6 and above (see also CICSKEYx). Must be entered in hexadecimal notation, such as: CICSKEY(00000F). (3 hexadecimal bytes)

CICSKEYX
Contains the last five bytes of transaction security key values to support CICS Release 1.6 and above. (5 hexadecimal bytes)

CICSPRI
Indicates CICS Operator Priority. For CICS support. (1 byte binary)

CICSRSL
Indicates CICS Resource Access Key. For CICS support. (3 hexadecimal bytes)

IDLE
The maximum time allowed (specified in minutes) between terminal transactions for this user. If exceeded, ACF2 will cause the Logonid and password to be revalidated before another transaction is accepted. A value of zero indicates no limit will be enforced. This field is available for IMS and CICS on-line processing. (1 byte binary)

IDMSPROF
The name of the SIGNON Profile CLIST executed when the user signs on to IDMS. (32 characters)

IDMSPRVS
The version of the SIGNON Profile CLIST executed when the user signs on to IDMS. (2 bytes binary)

MAXDAYS
The maximum number of days allowed between password changes before the password will expire. If this field is set to zero, no limit will be enforced. (1 byte binary)

MINDAYS

The minimum number of days that must elapse before the password can be changed by the user. (1 byte binary)

MUSOPT

Specifies the name of the ACF2/IDMS options module that will be used to control the IDMS address space. This field must be specified in the Logonid record assigned to the IDMS address space. See the acf2/MVS IDMS Support Manual for further information. (8 characters)

MUSPGM

Specifies the name of the IDMS startup program. This name must be specified in the Logonid record assigned to the IDMS address space. See the acf2/MVS IDMS Support Manual for further information. (8 characters)

PREFIX

The PREFIX field defines the high-level index of the datasets over which the user has ownership and, thus, access. The PREFIX field also identifies the key(s) of the access rule set(s) that the user can store and decompile. If the PREFIX field is null, then any dataset access that the user makes must be specifically allowed through access rules. The PREFIX field can contain a high-level index mask with asterisks (*), but not with a dash (-). The default PREFIX, normally the user's Logonid, can be specified by the DFT-PFX field of the user's Logonid record. (See the TSO Section of this field listing.) The system wide option, CENTRAL, determines whether users can decompile and store the access rule sets for the datasets that they own. See the explanation of the OPTS record in the chapter on Global System Option (GSO) records. (8 characters)

SHIFT

Specifies the name of any shift record that may define the allowable system access periods for this Logonid record (times of day, dates, or days). The LOGSHIFT field of the Logonid record allows a user to access the system during periods outside those defined by the SHIFT field; however, these accesses are logged. No masking is allowed within this field. For further information, see the chapter on shift/zone records. (8 characters)

SOURCE

The logical or physical input source name or source group name from which this Logonid must access the system. This can be the physical id or input source names defined to ACF2 via entry lists under type code SRC and source group names under type code SGP. No masking is allowed in this field. For further information, see the chapter on source records. (8 characters)

ZONE

Specifies the name of the Infostorage database record defining the time zone from which this Logonid normally accesses the system (i.e., the user's local time zone). No masking is allowed on this field. For further information, see the chapter on shift/zone records. (3 characters)

TSO Section--Group 5

Note that an asterisk (*) before the field name indicates that this field is used only if UADS is bypassed. Fields listed without an asterisk are always used, regardless of UADS mode.

*ACCTPRIV
Indicates user has TSO Accounting Privileges (for UADS updates). (BIT field)

ALLCMDS
Indicates the ability to bypass the ACF2 restricted command limiting lists by entering a special prefix character. (BIT field)

*ATTR2
The PSCBATR2 field used by the IBM Program Control Facility (PCF) for command limiting and dataset protection. The default value is zero (no PCF controls). This default may be changed if (1) the user's Logonid record is created with UADS in effect; (2) the user's Logonid record is copied from a "model" Logonid record; or (3) a new value is inserted. Note that ACF2 also has an option for TSO command limiting. See the discussion on TSO restricted command lists in the acf2/MVS System Programmer's Guide. (2 hexadecimal bytes)

*CHAR
The TSO Character Delete character for this user. May be specified either as a single character or as the special strings "BS" (signifying the backspace key, X'16') or "NO" (signifying no character-delete character desired).

CMD-LONG
When using TSO Command lists, ACF2 will normally accept as a command name the shortest character string that uniquely identifies a command in the list. This attribute bypasses this feature so that only the listed commands and aliases will be valid (i.e., full command names or aliases must be specified). (BIT field)

*DFT-DEST
The default remote destination for TSO spun sysout datasets. (8 characters)

*DFT-PFX
The default TSO prefix that will be set in the user's profile at logon time. This prefix is assumed as the high-level index whenever the user specifies a dataset name that is not fully qualified and within single quotes. (This prefix operates like the UADS Profile Prefix field.) A DFT-DPX value of period (.) indicates that the user must always specify a fully qualified dataset name. Any security administrator who adds or changes this field must not have an associated scope record that restricts access to those datasets with the high-level index to be specified in this field. (8 characters--however, the last character is reserved)

DFT-SOUT
The default TSO Sysout Class; for TSO or TSO/E Command Package Program Product only. (1 character)

DFT-SUBC
The default TSO Submit Class; for TSO or TSO/E Command Package Program product only. (1 character)

DFT-SUBH
The default TSO Submit Hold Class; for TSO or TSO/E Command Package Product only. (1 character)

DFT-SUBM
The default TSO Submit Message Class; for TSO or TSO/E Command Package Product only. (1 character)

*INTERCOM
Indicates this user is willing to accept messages from other users via the TSO SEND command. (BIT field)

*JCL
Indicates the ability to submit batch jobs from TSO (e.g., use TSO SUBMIT). (BIT field)

*LGN-ACCT
Indicates permission to specify account number at logon time. (BIT field)

*LGN-INDX
Indicates permission, during TSO logon under UADS, to specify the INDEX parameter or use the index specified in the UADSINDX field of the Logonid record. This index is used to locate the appropriate procedure in the UADS tree structure. For use of this permission, see the explanation for the UADS field of the OPTS GSO record (in the chapter on GSO records). (BIT field)

*LGN-MSG
Indicates this user has permission to specify message class at logon time. This may be helpful in debugging (in non-UADS mode only) by allowing the debugger to specify the message class for TSO session outputs. (BIT field)

*LGN-PERF
Indicates permission to specify performance group at logon time. (BIT field)

*LGN-PROC
Indicates permission to specify the TSO procedure name at logon time. (BIT field)

*LGN-RCVR
Indicates permission to use the recover option of the TSO or TSO/E Command Package. If not specified, the RECOVER attribute is not

applicable and the PROFILE RECOVER command may not be entered. (BIT field)

*LGN-SIZE
Indicates that this user is authorized to specify any region size at logon time (overriding TSOSIZE). Note: Any user may specify size at logon time without having this attribute, but will be restricted to a maximum allowable size based on his TSOSIZE unless he has the LGN-SIZE attribute. (BIT field)

*LGN-TIME
Indicates permission to specify the TSO session time limit at logon time. (BIT field)

*LGN-UNIT
Indicates permission to specify the TSO unitname at logon time. (BIT field)

*LINE
The TSO Line delete character. May be specified as a single character or as the special strings "ATTN" (signifying ATTENTION key), "CTLX" (signifying the CONTROL-X control character, X'18'), or "NO" (signifying no line-delete character desired). (1 character)

*MAIL
Indicates this user wishes to receive mail messages from TSO at logon time. (BIT field)

*MODE
Indicates this user wishes to receive modal messages from TSO. (BIT field)

*MOUNT
Indicates permission to issue mounts for devices. (BIT field)

*MSGID
Indicates this user wishes TSO messages to have message ids prefixed. (BIT field)

*NOTICES
Indicates this user wishes to receive TSO notices at logon time. (BIT field)

*OPERATOR
Indicates this user has TSO Operator Privileges. (BIT field)

*PAUSE
Indicates this user desires program to pause when a multi-level message is issued by a command executed within a CLIST. This allows the user to enter a question mark to receive the second-level messages. (BIT field)

*PMT-ACCT
Indicates this user will be forced to specify an account number at logon time (will be prompted by ACF2 if none provided). LGN-ACCT field of Logonid record should also be specified. (BIT field)

*PMT-PROC
Indicates this user will be forced to specify a TSO procedure name at logon time (will be prompted by ACF2 if none provided). LGN-PROC field should also be specified. (BIT field)

*PROMPT
Indicates this user wants to be prompted for missing or incorrect parameters. (BIT field)

*RECOVER
Indicates this user wishes to use the recover option of the TSO or TSO/E Command Package. The PROFILE RECOVER option will be set on at logon time. (BIT field)

*TSOACCT
The user's default TSO logon account. (40 characters)

TSOCMDS
The name of a TSO Command List Module that contains the list of the commands that this user is authorized to use. No masking is allowed on this field. Command limiting is effective for all Logonids including privileged ones. It takes place in all modes with the exception of QUIET. (8 characters)

*TSOFSCRN
Indicates this user will have the fullscreen logon display. (BIT field)

*TSOPERF
The user's default TSO performance group (1-255). Zero indicates that no performance group will be specified. (1 byte binary)

*TSOPROC
The user's default TSO procedure name. (8 characters)

*TSORBA
The Mail Index Record Pointer (MIRP) for this user. This pointer is for use only with the ACF2 expanded Logonid record and TSO/E, and is applicable only if the length of the Logonid record is 1024 bytes. For information on determining the Logonid record length, see the explanation of the LIDRECL field of the OPTS record in the chapter on GSO records. ACF2 automatically sets this value. (3 hexadecimal bytes)

*TSORGN
This user's default TSO region size (in K bytes) if no valid size is specified at logon time. (2 bytes binary)

*TSOSIZE
The user's maximum TSO region size (in K bytes), unless LGN-SIZE is specified for that user. (2 bytes binary)

*TSOTIME
The user s default TSO time parameter. (2 bytes binary)

*TSOUNIT
The user's default TSO unit name. (8 characters)

UADSINDX
The index to be used in locating the appropriate procedure in the UADS tree structure at logon time. For use of this field, the bit must be on in the LGN-INDX field of the Logonid record, and the installation must be using the User Attribute Dataset (UADS). See the explanation of the UADS field in the OPTS record in the chapter on GSO records. (8 bytes)

VLD-ACCT
Indicates that the TSO account number is to be validated. An ACF2 resource validation will be done using a generalized resource rule under type code TAC and the account number from logon processing. (BIT field)

VLD-PROC
Indicates that the TSO procedure name is to be validated. An ACF2 resource validation will be done using type TPR generalized resource rules and the procedure name from logon processing. (BIT field)

*WTP
Indicates Write-to-Programmer messages are to be displayed. Note that all ACF2 violation and warning messages are issued as WTPs, so that this attribute should be present on all TSO user Logonid records in order for them to receive ACF2 messages. (BIT field)

Statistics Section--Group 6

PSWD-DAT
Date of the last invalid password attempt. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd,depending upon the DATE field of the OPTS GSO record. For further information on the DATE option, see the chapter on GSO records.

PSWD-TOD
Date and time password was last changed. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE option of the OPTS GSO record. For further information on the DATE option, see the chapter on GSO records.

PSWD-VIO
Number of password violations which occurred on PSWD-DAT. (2 byte binary)

SEC-VIO
Number of security violations for this user (cumulative). (2 byte binary)

UPD-TOD
Date and time that this Logonid record was last updated. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE option of the OPTS GSO record. For further information on the DATE option, see the chapter on GSO records.

Field Summary by Group

The following is an alphabetical summary list of all fields of the Logonid record described previously, with the appropriate group number indicated:

ACCOUNT(2)
ACC-CNT(3)
ACC-DATE(3)
ACC-SRCE(3)
ACC-TIME(3)
ACCTPRIV(5)
ACTIVE(2)
ALLCMDS(5)
ATTR2(5)
AUDIT(2)
AUTHSUP1(4)
AUTHSUP2(4)
AUTHSUP3(4)
AUTHSUP4(4)
AUTHSUP5(4)
AUTHSUP6(4)
AUTHSUP7(4)
AUTHSUP8(4)
AUTODUMP(2)
CANCEL(1)
CHAR(5)
CICS(2)
CICSCL(4)
CICSID(4)
CICSKEY(4)
CICSKEYX(4)
CICSPRI(4)
CICSRSL(4)
CMD-LONG(5)
CONSULT(2)
CSDATE(1)
CSWHO(1)

DFT-DEST(5)
DFT-PFX(5)
DFT-SOUT(5)
DFT-SUBC(5)
DFT-SUBH(5)
DFT-SUBM(5)
DSNSCOPE(2)
DUMPAUTH(2)
EXPIRE(2)
IDLE(4)
IDMS(2)
IMS(2)
INTERCOM(5)
JCL(5)
JOB(2)
JOBFROM(2)
LEADER(2)
LGN-ACCT(5)
LGN-INDX(5)
LGN-MSG(5)
LGN-PERF(5)
LGN-PROC(5)
LGN-RCVR(5)
LGN-SIZE(5)
LGN-TIME(5)
LGN-UNIT(5)
LID(0)
LIDSCOPE(2)
LINE(5)
LOGSHIFT(2)
MAIL(5)
MAINT(2)

MAXDAYS(4)
MINDAYS(4)
MODE(5)
MON-LOG(1)
MONITOR(1)
MOUNT(5)
MSGID(5)
MUSASS(2)
MUSOPT(4)
MUSPGM(4)
NAME(0)
NO-SMC(2)
NO-STORE(2)
NON-CNCL(2)
NOTICES(5)
OPERATOR(5)
PASSWORD(0)
PAUSE(5)
PGM(2)
PHONE(0)
PMT-ACCT(5)
PMT-PROC(5)
PREFIX(4)
PROGRAM(2)
PROMPT(5)
PSWD-DAT(6)
PSWD-EXP(1)
PSWD-TOD(6)
PSWD-VIO(6)
READALL(2)
RECOVER(5)
REFRESH(2)

RESTRICT(2)
RULEVLD(2)
SCPLIST(2)
SECURITY(2)
SEC-VIO(6)
SHIFT(4)
SOURCE(4)
SRF(2)
STC(2)
SUBAUTH(2)
SUSPEND(1)
TAPE-BLP(2)
TAPE-LBL(2)
TRACE(1)
TSO(2)
TSO-TRC(1)
TSOACCT(5)
TSOCMDS(5)
TSOFSCRN(5)
TSOPERF(5)
TSOPROC(5)
TSORBA(5)
TSORGN(5)
TSOSIZE(5)
TSOTIME(5)
TSOUNIT(5)
UADSINDX(5)
UID(0)
UIDSCOPE(2)
UPD-TOD(6)
USER(2)
VSESRF(2)

VLD-ACCT(5)
VLD-PROC(5)
VM(2)
WTP(5)
ZONE(4)

Field Name Syntax and Types

The external names of the Logonid record fields as shown on the previous pages are one to eight characters in length and are used to display the contents of a Logonid record for modification. Each field name has various parameters associated with it that tell ACF2 how to process the field. For example, the permissible field types are:

BIT fields

These fields are associated with bits that are turned on through specification of the field name. For example, TSO grants the TSO attribute. Specification of the field name preceded by the keyword NO turns the bit off. For example, NOTSO takes away the TSO attribute. On output, the field will only be listed as the field name or the field name preceded by the word NO. Optionally, the listing of the field may be suppressed altogether if the bit is off.

CHARACTER fields

These fields are specified with the field name followed by the value of the field in parentheses. To specify a blank field, the null string of () may be used. Optionally, the listing of the field can be suppressed if the field is blank or zero.

PACKED DECIMAL DATE fields

These fields are specified by the field name followed by the Gregorian date in parentheses. This date can be in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE field of the OPTS GSO record. For further information on the DATE option, see the chapter on GSO records. Optionally, the listing of the field can be suppressed if the value is zero.

BINARY fields

These fields may be defined as being 1, 2, 3, or 4 bytes long. This byte length determines the maximum value that may be assigned to the field. Only binary 2- and 4-byte fields may contain negative values. The permissible formats are:

ddddd

Up to five binary digits with no sign will assign the specified value to the field.

+ddddd

the string preceded by a plus sign will increment the field value by the specified value, and

-ddddd

the minus sign will decrement the field value by the specified value.

Optionally, the listing of the field can be suppressed if the value is zero.

HEXADECIMAL fields
These fields may be defined as a maximum of 8 bytes (16 hexadecimal digits). The permissible operands are a string of valid hexadecimal digits (0-9,A-F) of even length, enclosed in parentheses. The assignment to the field is left-justified. Trailing zeros are truncated. Optionally, the listing of the field can be suppressed if the value is zero.

CREATING A LOGONID RECORD

The INSERT subcommand allows a user with the ACCOUNT privilege level to create a new Logonid record. For example:

```
insert pay7777 name(nadia tormell) leader
```

In the this example, a record is inserted for the Logonid PAY7777. The name associated with the Logonid is Nadia Tormell, who has the LEADER privilege level.

The above INSERT subcommmand inserts only two fields, NAME and LEADER, into the Logonid record for PAY7777. Normally, a number of fields are added to the record. Thus, the USING parameter is specified:

```
insert using(paymod) pay7777 name(nadia tormell) leader
```

The USING parameter allows a new record to be copied from an existing, model Logonid record. Fields in the new record are added, changed, or deleted as necessary. The example INSERT subcommand above uses the Logonid record named PAYMOD. To create the new record, fields are copied from the Logonid record PAYMOD. The user's name is changed to Nadia Tormell, and the LEADER privilege level is added.

When listed, this new record might look like:

```
PAY7777         PAY7777 NADIA TORMELL EXT. 458
PRIVILEGES      TSO LEADER SCPLIST(FINANCE)
ACCESS          ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS   PREFIX(PAY7777)
TSO             TSOACCT(ACCT01) MAIL NOTICES
STATISTICS      PSWD-TOD(01/04/86-12:01) UPD-TOD(01/04/86-12:01)
```

Synchronizing New Logonid Records. After inserting new Logonid records for the first time, you should synchronize those records with the BRODCAST dataset for all TSO users. This is not necessary once ACF2 is running since the INSERT USING command will add an entry to the SYS1.BRODCAST dataset. With the INSERT USING command, Logonids are added to SYS1.BRODCAST dataset based upon the LOGONCHK field in the TSO

GSO record. If LOGONCHK is opted, TSO Logonids will be entered on the SYS1.BRODCAST dataset. Otherwise, all Logonids will be added to the dataset. It should be noted that the ACF SYNCH processing is restricted to a Logonid of no more than seven characters in length. Any Logonid of eight characters in length will br entered on the SYS1.BRODCAST dataset under only its first seven characters. The last character is truncated. For further information on this synchronization, see the explanation of the ACF SYNCH subcommand later in this chapter.

ISPF Screens for Logonid Record Processing. ISPF screens for Logonid record processing are provided with ACF2. Please refer to the acf2/MVS Utilities Manual for information on the use of this feature.

CHANGING A LOGONID RECORD

Users with the ACCOUNT, LEADER, or SECURITY privilege levels can change selected Logonid records. Access to these records can be restricted by scopes, as discussed in the chapter on scope records.

A Logonid record is changed through the CHANGE subcommand. To understand the CHANGE command, first view the Logonid record previously created for Nadia Tormell:

```
list pay7777

PAY7777            PAY7777 NADIA TORMELL EXT.458
PRIVILEGES         LEADER SCPLIST(FINANCE) TSO
ACCESS             ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS      PREFIX(PAY7777)
TSO                MAIL NOTICES TSOACCT(ACCT01)
STATISTICS         PSWD-TOD(01/04/86-12:01) UPD-TOD(01/04/86-12:01)
```

Suppose she needs to use CICS but no longer needs to use TSO. The following CHANGE subcommand adds the CICS attribute and removes the TSO and other TSO-related attributes:

```
change pay7777 cics notso tsoacct() nomail nonotices
```

In the above subcommand, the NO prefix is used to reverse the effects of a bit field. For example, NOTSO removes that privilege from the Logonid. Since the TSOACCT field has an associated value in parentheses, that field must be removed by specifying closed parentheses with no value.

When listed, her Logonid record now looks like:

```
PAY7777            PAY7777 NADIA TORMELL EXT.458
PRIVILEGES         CICS LEADER SCPLIST(FINANCE)
ACCESS             ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS      PREFIX(PAY7777)
STATISTICS         PSWD-TOD(01/04/86-12:01) UPD-TOD(01/04/86-12:10)
```

The CICS field now appears in the Privileges Section of this Logonid record and the TSO attribute and its related fields have all been removed. The time and date of the update is recorded in the UPD-TOD field.

Changing More than One Logonid Record Simultaneously. When specified with the CHANGE subcommand, the LIKE parameter allows you to change more than one Logonid record at a time. The Logonid records to be changed are identified by a Logonid mask:

```
change like(pay***) cics notso tsoacct() nomail nonotices
```

The Logonid mask PAY*** changes all Logonid records for the payroll department (all records for Logonids consisting of the letters PAY followed by any 0 to 3 characters, including blanks). Masking of Logonids is discussed in the next section.

No synchronization of the SYS1.BRODCAST dataset will take place for masked entries in the LIKE field of the CHANGE command. However, the actual Logonids with the TSO attribute that DO NOT contain masked | characters (but are within the range of the masked entry) will be added to the SYS1.BRODCAST dataset.

MASKING OF LOGONIDS

A mask allows you to specify multiple Logonids at one time.

For example, the Logonid mask PAY*** represents all Logonids that begin with the letters PAY and end with any 0 to 3 characters.

Masking is useful for processing multiple Logonid records with a single ACF subcommand. Logonid masks have other uses in processing other ACF2 records and in creating ACF2 rule sets.

Two different symbols can signify masking: asterisks and dashes.

Asterisks

A Logonid mask containing asterisks represents all valid Logonids beginning with the specified characters and ending with any number of characters not exceeding the number of asterisks. These characters can include blanks. For example:

```
PAY*** will match  PAY
                   PAYDS
                   PAYDSJ

         but not   PA
                   PBYKL
                   PAYKLST
```

Dashes

A Logonid mask containing a dash represents all valid Logonids that begin with the specified characters. The total length of a valid Logonid must be eight or fewer characters. For example:

```
PAY-  will match  PAY
                  PAYDS
                  PAYDSJ
                  PAYDS12J

          but not PA
                  PBYKL
                  PAYKLSTSS
```

AUTHORITY OF PRIVILEGED LOGONIDS

A privileged (e.g., SECURITY, ACCOUNT) Logonid without a scope list is granted various levels of access to the ACF2 databases. Careful attention should be taken when granting these privileges to users.

Privilege Levels and Record Processing. A user with the SECURITY privilege level can list and change fields of Logonid records within the restrictions imposed by his scope. However, he cannot create or delete Logonid records unless he also has the ACCOUNT privilege level. Since the SECURITY privilege enables the user to write access rules as well as update the Infostorage database, those Logonids are considered the most powerful.

A user with the ACCOUNT privilege has the authority to insert, list, delete, and change Logonid records within the restrictions imposed by his scope. Because of this authority, a user with the ACCOUNT privilege level is normally assigned the task of maintaining Logonid records or user accounts. However, a user with the ACCOUNT privilege alone cannot change or list the Logonid record of a user with both the ACCOUNT and SECURITY privilege levels. A user with both these privilege levels is more powerful than a user with only one of these.

A user with the LEADER privilege level can list and change certain fields of Logonid records. A user with the LEADER privilege level is usually restricted by a scope and can therefore only list or change fields in a selected group of Logonid records.

The following chart illustrates the types of access to the Infostorage, Logonid and/or Rules databases that are granted by these special privileges. The access described is that which is over and above the basic privileges allotted to all users. The asterisk (*) is used to indicate the unique privileges.

ACF2 DATABASE ACCESS

		LID	RULE	INFOSTORAGE
PRIVILEGES	SECURITY	Display, update only	*Compile, store and display (Note 1)	*Create, update and display (Note 2)
	ACCOUNT	*Create, update and display *Use SYNCH command	None	None
	AUDIT	Display	Decompile (Note 3)	Display (Note 2, 3)
	LEADER	Display and update designated fields	None	None
	CONSULT	Display and update designated fields	None	None
	USER	Display and update limited fields in own record only (Note 4)	Compile, store, and display rules that match owner PREFIX in a decentralized environment (Note 5)	None

Notes:

1. Since an unrestricted Logonid with SECURITY authority can write access rules for any dataset, this Logonid is also granted access (but logged) to all datasets outside of the ACF2 databases. There is no requirement for an access rule unless the RULEVLD bit is set in the user's Logonid. In this situation, a rule must be written on the ACF2 Rule database before that Logonid can access the dataset.

2. The various ACF2 SHOW commands can be used by SECURITY, and AUDIT Logonids.

3. The AUDIT privileges to decompile rule and display Infostorage records are based upon the DECOMP and INFOLST fields of the OPTS GSO record.

4. All users have the ability to display their own Logonid and update a limited number of fields.

5. In a decentralized environment, all users can write access rules for their own PREFIX dataset.

FIELD-LEVEL AUTHORITY

Each installation has the flexibility to define which fields of the Logonid record can be listed and which fields can be altered. Furthermore, the installation can determine which privilege levels are required for each type of access.

For example, only a user with either the ACCOUNT or SECURITY privilege levels may have the authority to change the NAME field of a particular Logonid record. Yet, all users may have the authority to see the NAME when they list the Logonid record. Of course, a user must also have record-level authority to access the Logonid record.

Field-level authority to access Logonid records is defined by the LIST= and ALTER= operands in the @CFDE macros of the ACF2 Field Definition Record (ACFFDR). See the System Programmer's Guide for a list of the default LIST= and ALTER= operand values for each field.

DELEGATING AUTHORITY FOR ACF2 DATABASES

Installations have several options to delegate a limited number of the tasks granted a privileged Logonid without allowing all of the authority of an unrestricted privileged Logonid. How much authority is delegated to that Logonid depends upon the nature of the special privileges themselves.

The most common method used to delegate a limited amount of authority, but not all the authority granted to privileged Logonids, is via scope records. A scope record can be used to limit a user's access to the Logonid, Rule, and Infostorage ACF2 databases. The authority granted depends upon the privilege that user has. For example, adding a scope list to an ACCOUNT Logonid can limit that user's access to those Logonid records related to his office, rather than to his entire company. To establish these limits, a SCOPE record must be created on the Infostorage database. Entries must be made in the LID and UID fields of the SCOPE record related to those Logonids which the user will create. Next, the name of the SCOPE record must be entered in the SCPLIST field of that ACCOUNT Logonid before any restrictions will apply.

Logonids with the privilege named in the INFOLST field of the GSO OPTS record will be permitted to read the Infostorage database. Similarly, Logonids with the privilege named in the DECOMP field of the GSO OPTS record will be permitted to decompile generalized resource or access rules. These privileges will override the read restrictions imposed by scope lists on associated Logonids. However, write restrictions will still be enforced by the scopes.

For example, assume the default privileges on both the DECOMP and INFOLST fields in the OPTS GSO record are SECURITY and AUDIT. Next, a scope record is created to limit a user's access to rule sets beginning with PAY and the creation of Infostorage records for entry and source groups. When listed, this scope might look like:

```
list limit3

ACF60062 SCOPE LIMIT3 STORED BY PAYDSH ON 07/15/85 - 11:43
DSN(PAY-) INF(ESGP-,ESRC-)
```

A SECURITY Logonid with this scope will be limited to creating rules on the Rule database and source records on the Infostorage database. However, the Logonid will still be able to decompile any rule and display any Infostorage record (since it has the same privilege named in the DECOMP and INFOLST fields of the OPTS GSO record).

The authority to update generalized resource or access rules can be delegated using the %CHANGE or %RCHANGE parameters in a rule set. This is discussed in the ACCESS RULE chapter in this manual.

The authority to maintain entry lists can also be delegated via the PSEUDODSN field on the ENTRY record. The security officer enters any name in this field. Then an access rule set is created by entering the PSEUDODSN name in the high-level index field of the $KEY parameter. An access rule for the Logonid which will maintain the entry records is then entered in the rule set.

DSNSCOPE, LIDSCOPE and UIDSCOPE Fields to Be Phased Out. The DSNSCOPE, LIDSCOPE or UIDSCOPE fields of a user's Logonid record can also restrict that user's privileges and act as scopes. However, these fields will be phased out in a future release of acf2/MVS. All installations should put the name of the SCOPE record (in Infostorage) in the SCPLIST field on the user's Logonid.

Any installations currently using the DSNSCOPE, LIDSCOPE, or UIDSCOPE fields should convert to the use of the SCPLIST field. It should be noted that any scopes specified through the SCPLIST field on the Logonid will override any scopes specified via the DSNSCOPE, LIDSCOPE or UIDSCOPE fields.

ACF SUBCOMMANDS UNDER THE LID SETTING

You can process Logonid records after establishing the LID setting of the ACF command:

```
set lid
```

After establishing the LID setting, you can issue any of the following ACF subcommands:

INSERT
CHANGE
LIST
DELETE
END
HELP
SET
SHOW
SN
SYNCH

The common subcommands END, HELP, SET, SHOW, and SN operate under the LID setting as previously explained. The following text describes the function, syntax, and parameters of the other subcommands under LID mode.

INSERT Subcommand--LID Setting

The INSERT subcommand, under the LID setting, allows a user with the ACCOUNT privilege level to add a new Logonid record to the Logonid database.

Syntax. The syntax of the INSERT subcommand is:

```
INSERT [USING(old-logonid)] new-logonid {field1,field2,...,fieldn}
```

Example. The following example INSERT subcommand creates a Logonid record for the Logonid PAY7777:

```
insert pay7777 name(nadia tormell) leader lidscope(pay-) -
 phone(ext.458) password(xxxx) tso tsoacct(acct01) mail notices
```

When listed, the new Logonid record looks like:

```
list pay7777

PAY7777           PAY7777 NADIA TORMELL EXT. 458
PRIVILEGES        TSO
ACCESS            ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS     PREFIX(PAY7777)
TSO               TSOACCT(ACCT01) MAIL NOTICES
STATISTICS        PSWD-TOD(01/04/86-12:01) UPD-TOD(01/04/86-12:01)
```

Synchronizing New Logonid Records. After inserting new Logonid records for the first time, you should synchronize those records with the BRODCAST dataset. For further information on this synchronization, see the explanation of the ACF SYNCH subcommand later in this chapter. Once the SYNCH utility has been run, additional TSO Logonids will be added to the BRODCAST dataset using the INSERT command if LOGONCK is set in the TSO GSO record. In all circumstances, eight character Logonids will not be added to the SYS1.BRODCAST dataset. Otherwise, all Logonids (for ROSCOE, VM, etc.) will be added to the BRODCAST dataset with the INSERT command.

Parameters

The INSERT command can be followed by an asterisk (*) meaning INSERT the last Logonid I referenced. This can be especially useful if you just deleted one Logonid and wish to recreate a similar one without your installation's privileges, zerofields, or statistics.

USING(old-logonid)

The USING parameter allows you to use an existing Logonid record as a model, thus reducing the number of fields you must enter. You enter this parameter along with the 1- to 8-character Logonid for the record that you wish to use as the model.

An installation can specify fields that cannot be copied from a model Logonid record to a new Logonid record. These fields are specified through the @CFDE macro of the ACF2 Field Definition Record (ACFFDR). Use the SHOW ZEROFLDS command of the ACF2 subsystem to display the list of specified fields for your installation. See the acf2/MVS System Programmer's Guide for further information on this macro.

new-logonid

The new-logonid is the 1- to 8-character Logonid for the new record that you are adding. This parameter must be specified.

field1,field2,...,fieldn

Any fields to be in the new Logonid record are specified by the field name and, if necessary, the corresponding value for the field. The allowable fields are described toward the beginning of this chapter.

Two different types of fields can exist:

1. Fields without values are specified by the field name (for example, TSO). These fields have been identified in the previous field descriptions by the word BIT. These field can be removed from the Logonid record by specifying the field name preceded by the letters NO (for example, NOTSO).

2. Fields with values are specified by the field name immediately followed by the value in parentheses. The format for these fields are described in the previous field descriptions.

You can specify these fields in any order but must separate them by commas or blank spaces.

If you have specified the USING parameter, then any specified fields are added to the new Logonid record. If a particular field is already in the existing, model Logonid record, then that field is either changed or deleted accordingly.

ISPF Screens. ISPF screens are provided with ACF2 for inserting Logonid records. Please refer to the acf2/MVS Utilities Manual for details on the use of this feature.

CHANGE Subcommand--LID Setting

The CHANGE subcommand, under the LID setting, allows you to add, change, or delete fields of selected Logonid records.

Syntax. The syntax of the CHANGE subcommand is:

```
CHANGE {*/logonid/LIKE(logonid-mask)/
UID(uid-mask)/IF(field1,NOfield2)
```

Example. For example, take the following Logonid record for the Logonid PAY7777:

```
list pay7777

PAY7777          PAY7777 NADIA TORMELL EXT.458
PRIVILEGES       TSO
ACCESS           ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS    PREFIX(PAY7777)
TSO              MAIL NOTICES TSOACCT(ACCT01)
STATISTICS       PSWD-TOD(01/04/84-12:01) UPD-TOD(01/04/84-12:01)
```

To add the CICS attribute and remove the TSO and other TSO-related attributes, the following CHANGE subcommand is used:

```
change pay7777 cics notso tsoacct() nomail nonotices
```

In the above subcommand, the field name CICS is used to add the CICS attribute. The letters NO precede the field name to remove the TSO, MAIL and NOTICES attributes. Since the TSOACCT field has a value associated with it, then the field is removed by specifying the field name followed by no value in parentheses. (A value could alternatively be specified in order to change the current value of the field.)

After this change, the Logonid record looks like:

```
PAY7777          PAY7777 NADIA TORMELL EXT.458
PRIVILEGES       CICS
ACCESS           ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
MISCELLANEOUS    PREFIX(PAY7777)
STATISTICS       PSWD-TOD(01/04/82-12:01) UPD-TOD(01/04/82-12:01)
```

Parameters. The CHANGE subcommand takes the following parameters:

*
The asterisk specifies changing of the last Logonid record processed since the LID setting was established.

logonid
A Logonid identifies a single Logonid record to be changed.

LIKE(logonid-mask)
The LIKE parameter specifies a 1- to 8-character mask that identifies the group of Logonid records to be changed simultaneously. For example, the mask PAY*** identifies records for all Logonids beginning with the letters PAY and ending with any 0 to 3 characters excluding imbedded blanks. PAY- would match all Logonids that begin with PAY regardless of length. If this parameter is used to add or delete the TSO attribute to a series of Logonids, be aware that the synchronization of the SYS1.BRODCAST dataset will not take place. To re-create the SYS1.BRODCAST dataset, use the ACF2 SYNCH or BSYNCH utilties.

IF(field1,NOfield2....fieldn)
The IF parameter allows you to change Logonids with similar attributes. Attributes can be used in combination for this type of entry. For instance, CHANGE IF(ACCOUNT,NOSECURITY) would change all the Logonid records with ACCOUNT but without SECURITY (both criteria must be met). Multiple fields specified in one IF command process are treated as "AND" fields for processing. In other words, records are only treated as matching the request if all fields specified are a match for that record. Prefix the field name with NO to indicate Logonids without that option. Any bit fields in the Logonid record can be used for this command.

UID(uid-mask)
The UID parameter allows you to specify a UID string/mask identifying Logonids that are to be included in the change.

LIST Subcommand--LID Setting

The LIST subcommand, under the ACF or LID setting, allows you to list Logonid records. Any users, even without privilege levels or special privileges, can list their own Logonid records.

The ability to display other Logonid records can be granted through privilege levels and restricted through scope records.

If you have issued a SET TERSE subcommand, only the first line of each Logonid record is displayed. This display can be controlled through the @HEADER macro of the ACF2 Field Definition Record (ACFFDR). See the acf2/MVS System Programmer's Guide for further information.

Syntax. The LIST subcommand can be entered in any combination of the following format:

```
      LIST {*/logonid/LIKE(logonid-mask)/IF(field-list)/UID(uid-mask)
```

Parameters. The LIST subcommand takes the following parameters:

*
An asterisk allows you to list the last Logonid record that you referenced in your current TSO session. Your own Logonid record is displayed if you have not specified an individual Logonid in any subcommands since you issued the ACF command.

logonid
An individual Logonid allows you to list one, specific Logonid record.

LIKE(logonid-mask)
The LIKE parameter allows you to specify a mask for the Logonids whose records you want to display. Since masking is supported, a dash can be entered to display all Logonids within the user's scope.

IF(field1,NOfield2,....)
The IF parameter allows you to list the Logonid records for those users with the specified attribute. For example:

```
      LIST IF(ACCOUNT,NOSECURITY)
```

This subcommand lists Logonid records for those users with the ACCOUNT privilege but without SECURITY. You can use the IF parameter to list records for Logonids with particular combinations of attributes. For example:

```
      LIST IF(ACCOUNT,SECURITY)
```

This subcommand lists those Logonid records for those users with both the ACCOUNT and SECURITY privileges.

With the IF parameter, you can only specify the BIT fields of the Logonid record as defined by ACF2 or the installation. The IF parameter can be specified by itself or with either the LIKE or UID parameter. To list Logonids without a specific bit, prefix the field name with NO.

UID(uid-mask)
The UID parameter allows you to specify a mask for the UID strings identifying the users whose Logonid records will be displayed.

DELETE Subcommand--LID Setting

The DELETE subcommand, under the LID or ACF setting, allows for deletion of Logonid records. By default, this subcommand also deletes any access rule set whose key matches that of any deleted Logonid record. Additionally, any entry to the Logonid on the SYS1.BRODCAST dataset will also be deleted.

Only users with the ACCOUNT or SECURITY privilege levels can delete Logonid records. This authority can be restricted through the use of scopes.

Syntax. The syntax of the DELETE subcommand can be used in any combination:

```
     DELETE {*/logonid/LIKE(logonid-mask)/UID(uid-string-mask)/
             IF(field1,NOfield2)} [NORULE]
```

Example. The following subcommand deletes the Logonid record and corrsponding access rule set for the Logonid PAY7777:

```
     delete PAY7777
```

After the DELETE command is issued, the message DELETED is returned.

Parameters. The DELETE subcommand takes the following parameters:

*
An asterisk allows you to delete the last Logonid record that you referenced in your current TSO session. You cannot delete your own Logonid.

logonid
An individual Logonid allows you to delete one, specific Logonid record.

LIKE(logonid-mask)
The LIKE parameter allows you to specify a mask for the Logonids whose records you want to delete.

UID(uid-mask)
The UID parameter allows you to specify a mask for the UID strings identifying the users whose Logonid records will be deleted.

IF(field1,NOfield2,....)
The IF parameter allows you to delete Logonids with similar attributes. These attributes can be used in combination for this type of entry. For instance, DELETE IF(ACCOUNT,NOSECURITY) causes any Logonids with the ACCOUNT but not the SECURITY privilege to be deleted. To change Logonids without a specific privilege, prefix the fieldname with NO. Any bit Logonid fields can be used.

NORULE
The NORULE parameter indicates that although the Logonid record(s) specified will be deleted, the associated access rules with the same identifier should not be deleted. The default is for deletion of any access rule set whose ruleid ($KEY) matches the Logonid of any deleted record.

SYNCH Subcommand--LID Setting

The SYNCH subcommand, under the ACF or LID setting, allows you to synchronize the ACF2 database with the TSO SYS1.BRODCAST dataset. This subcommand may have to be issued when Logonids records, particularly for TSO users, have been inserted for the first time. The synchronization can be done for all or selected Logonid records. Each time the synchronization is performed, the SYS1.BRODCAST dataset is completely rebuilt. Before doing the synchronization, please examine your installation's synchronization standards to ensure that the appropriate users are included in the BRODCAST dataset.

Only "unrestricted" users with the ACCOUNT privilege level can issue this subcommand. (See the chapter on scope records for an explanation of unrestricted users.)

Synchronization Utility. Alternatively, the ACFBSYNC utility can perform the Logonid record synchronization in batch. See the acf2/MVS Utilities Manual for information on ACFBSYNC.

Syntax. The SYNCH subcommand has the following syntax:

```
SYNCH {LIKE(logonid-mask)/UID(uid-mask)/IF(field-list)}
```

Parameters. The SYNCH subcommand takes the following parameters:

LIKE(logonid-mask)
The LIKE parameter allows you to specify a mask for the Logonids whose records will be synchronized.

UID(uid-mask)
The UID parameter allows you to specify a mask for the UID strings identifying the users whose Logonid records will be synchronized.

IF(field1,NOfield2,....)
The IF parameter allows you to synchronize Logonid records for those users with the specified attribute. For example:

```
SYNCH IF(TSO)
```

This subcommand will synchronize Logonid records for those users with the TSO privilege. You can use the IF parameter to synchronize records for Logonids with particular combinations of attributes. For example:

```
SYNCH IF(TSO,NON-CNCL)
```

This subcommand synchronizes the Logonid records for those users that have the TSO and NON-CNCL privileges. Only those Logonids which match these attributes will be on BRODCAST. ACF2 rebuilds the dataset each time the SYNCH command is used. Care must be taken when combining the attributes of the IF subcommand.

With the IF parameter, you can specify any bit field of the Logonid record. To specify Logonids without a specific privilege, prefix the field with NO.

RULE SETTING: ACCESS RULES

An ACF2 access rule specifies which system users and under what conditions those users can access an individual or a group of datasets. ACF2 protects all system data by default. Datasets can only be accessed by:

1. A user who has been granted specific access to the dataset by means of an ACF2 access rule.

2. The owner of the dataset as defined by the PREFIX field of the Logonid record when NOCENTRAL is specified on the GSO OPTS record.

3. A user with special override Logonid privileges:

   a) NON-CNCL

   b) SECURITY (unless RULEVLD is also on)

   c) Scoped Security (DSNSCOPE or SCPLIST specified and RULEVLD is also on)

   d) READALL (if access is for READ or EXECUTE only)

All data on a system running ACF2 is protected, even when new datasets are created.

This chapter discusses:

1. An example of an ACF2 access rule set

2. The elements of the ACF2 access rule set

3. Creating an access rule set

4. Masking in access rule sets

5. Use of the NEXTKEY feature

6. ACF2 features that simply the writing of access rules

7. ACF subcommands under the RULE setting

EXAMPLE OF AN ACF2 ACCESS RULE SET

When listed, a simple access rule set might look like:

```
$KEY(PAYROLL)
MASTER.DATA UID(TFINPAYNLT) READ(A) WRITE(A) EXEC(A)
```

This access rule set pertains to all datasets with a high-level index of PAYROLL. Only one rule entry exists in this rule set, and is interpreted as follows:

* MASTER.DATA specifies that this rule entry pertains only to the dataset PAYROLL.MASTER.DATA.

* UID(TFINPAYNLT) specifies the User Identification (UID) string of the user to which access to the dataset will be granted.

* READ(A) WRITE(A) EXEC(A) specifies that these users will have read, write, and execute authority to the dataset. Since ALLOC(A) is not specified, then allocate authority is assumed to be prevented--i.e., ALLOC(P).

THE ACCESS RULE SET

An access rule set consists of those access rules that pertain to datasets of a particular high-level index.

Each access rule set is made up of:

1. Control cards

2. Access rule entries

3. Comment cards

In the previously shown access rule set, the first line contained a control card. The second line was an access rule entry.

An access rule set can also contain comment cards, which are explained after the explanation of the control cards. Control cards must all begin in column 1. The two basic types of control cards are the $ control cards and % control cards. A rule set can contain only one of each type of $ control card but as many of each type of % control card as desired. The $KEY control card is the only required control card.

The types of control cards are:

$KEY(high-level-index)

The $KEY control card supplies the high-level index of the dataset name for which this rule is being written or the VSAM key of the rule set. For example, when compiling a rule set to allow access to the dataset SYS1.PARMLIB, the $KEY control card contains $KEY(SYS1), since SYS1 is the high-level (or first) index of the dataset name. During access validation, the $KEY value will be used as the prefix unless the $PREFIX control card is specified.

When writing rules for your own datasets, the $KEY control card will usually contain your Logonid, since the Logonid is often specified as the PREFIX for your own datasets. This field is non-maskable.

Note: The security officer can completely delegate the authority of writing rule entries by setting up skeleton rule sets that contain only a $KEY, a %CHANGE or %RCHANGE, and any other control cards.

$MODE(QUIET/LOG/WARN/ABORT)

This control card is only effective if the system-wide option of RULE mode has been selected (see the explanation of the MODE field of the OPTS record in the chapter on GSO records). ACF2 access rule validation will be based upon the MODE contained in this control card. If the MODE is LOG, access violations will be logged but allowed; if the MODE is ABORT, the access violations will be logged and not allowed. Valid modes affect the access rule disposition as follows:

$MODE(QUIET) Implies that all accesses to any dataset(s) covered by this ruleset that were not specifically allowed by a rule are to be allowed. No ACF2 dataset/program SMF loggings are to occur.

$MODE(LOG) Implies that all accesses to any dataset(s) covered by this ruleset which were not specifically allowed by a rule are to be allowed, but access is to be logged. ACF2 dataset/program SMF loggings are to occur.

$MODE(WARN) Implies that all accesses to any dataset(s) covered by this ruleset that were not specifically allowed by a rule are to be allowed, but the installation warning message (GSO WARN record) is to be issued along with the ACF2 violation message (ACF99913). In addition, ACF2 dataset/program SMF loggings are to occur.

$MODE(ABORT) Implies that all accesses to any dataset(s) covered by this ruleset that were not specifically allowed by a rule are to be denied. ACF2 console messages are to be issued and ACF2 dataset/program SMF loggings are to occur.

The $MODE control card takes effect when access to the specified datasets would have been denied for that user or if the specific dataset name accessed is not contained in the rule set. Observe the following:

```
$KEY(ACCTPAY)
$MODE(LOG)
 MONTHLY.DATA UID(ACCTUID) R(A) W(A)
```

Based on this rule, only the user whose UID is 'ACCTUID' would be allowed access to ACCTPAY.MONTHLY.DATA. This user would only be allowed read or write privileges to this dataset. If this user attempted an access to this dataset other than read or write, then this rule would not apply. This rule would also not apply to a user accessing this dataset whose UID is not 'ACCTUID'. If no rule applies, ACF2 would check the current GSO OPTS record MODE field. In this example, if ACF2 were in RULE mode, then access would be allowed based on the $MODE(LOG) control card in the access rule.

The $MODE control card is an aid in the transition to full security by allowing the phasing-in of protection at the rule set level. Use of this control card also eliminates the need for a violation or post-validation exit for this type of checking during the transition period.

$NOSORT

This control card prevents standard ACF2 sorting of access rules when a rule set is stored. The rules remain in the order in which they were first entered into the compiler via the terminal or a partitioned dataset. Without this control card, ACF2 sorts and then stores a set of access rules from most specific to most general in terms of access environment. (For example, PAYROLL.PROD.- is more specific than PAY-.-) A warning message is issued after compilation whenever a $NOSORT card is used. This card should be used with caution, since a general access rule may prevent a more specific rule from being evaluated. The $NOSORT card is effective only when the installation has specified $NOSORT in the OPTS GSO record. The $NOSORT option is discussed in the chapter on GSO records.

$OWNER(owner-id)

Up to 24 characters may be specified via the $OWNER control card. This is an informational field only--no ACF2 processing is based on this information. The Logonid or name of the "owner" of this rule set may be entered here for local tracking purposes.

The $OWNER data will be stored with the rule set and will be displayed when the rule set is decompiled (similar to the $USERDATA information). The $OWNER data will also be contained in SMF records which can be used to produce reports. A site may use its own conventions for the information placed in $OWNER to facilitate reporting methods. Note that use of the $OWNER control card does not grant the user specified any special privileges regarding the rule set.

$PREFIX(prefix)

This control card indicates the value that will override the rule set key as a prefix to all dataset names in this rule set. A maximum of 24 characters may be entered. Multiple dataset name levels may be specified within the $PREFIX control card.

Note that when the NEXTKEY parameter is used in the rule set to indicate that an "alternative" rule set should be checked, the $PREFIX control card must be specified in that alternative rule set to indicate the true high-level index of the dataset name.

The ACF2 Access Rules database key for this rule set is still determined by the $KEY operand. The rule set key to be used to validate an access request may be set by use of the NEXTKEY rule parameter or an installation exit. When a local exit is used, it is the responsibility of the installation-supplied dataset pre-validation exit code to recognize that the search key must be set to something other than the dataset name high-level index and modify the rule key to be used in the search.

If $PREFIX() is specified, the prefix is set equal to the $KEY entry and the $PREFIX control card will not be generated when this rule is decompiled. ACF2 will issue a warning message indicating that the $PREFIX specified is null and will be ignored.

$USERDATA(text)

$USERDATA may contain any text string up to 64 characters. Information placed in the $USERDATA field is stored with the rule set. This information may be accessed by installation post-validation or violation exits. Comment cards represented by an asterisk(*) and a space in columns 1 and 2 can also represent user information and will be removed after compilation. You can also use the DATA parameter (discussed later in this chapter) to store user information in a rule entry within an access rule set.

%CHANGE uidmask1,uidmask2,...,uidmaskn

This control card indicates who, besides the high-level index owner or security officer, may replace or delete a set of rules. Specified on this contol card are the User Identification (UID) strings or UID-string masks that identify the users who have this "change authority." Multiple UID strings or masks must be separated by commas or blank spaces.

Note that use of %CHANGE allows the user designated to change the rule set to further delegate %CHANGE authority to other users. The designated user may change or delete any part of the rule set. For activation or deactivation of the installation's ability to use this control card, refer to the explanation of the CHANGE field of the OPTS record in the chapter on GSO records.

For the purpose of delegating rule writing authority, a security officer may compile and store a rule set that contains only the $KEY and %CHANGE control cards (i.e., no rule entries). This allows a "skeleton" rule set to be stored, awaiting refinement by the designated 'changer', without the security officer having to write any rule entries.

%RCHANGE uidmask1,uidmask2,...,uidmaskn

This control card specifies the User Identification (UID) strings or UID-string masks that identify the users who have "restricted change authority" over the rule set. A designated user can change individual rule entries, but not control cards. He cannot further delegate any change authority, nor can he delete the rule set.

If the same user matches entries in both %CHANGE and %RCHANGE, the %CHANGE authority will be in effect for that user. The designated user may change or delete any part of the rule set.

For activation or deactivation of the installation's ability to use the %RCHANGE control card, refer to the explanation of the CHANGE option of the OPTS record in the chapter on GSO records.

Placement of Control Cards. Any number of '$' or '%' control cards may be in the compiler input, but if the same $ control card type is entered more than once, only the last entered value will be used. All $ control cards must come before any other cards. Comment cards may appear anywhere in the input. Multiple $ Control Cards can be coded on one line if the $ appears in Column 1:

```
    $KEY(SYS1) MODE(ABORT) OWNER(JOESMITH)
```

Continuation of Control Cards. All input to the compiler may be continued on multiple cards by the use of a dash (-) as the last non-blank character on the line.

Note that a continuation is recognized unconditionally. For instance, if a comment card is continued, the next line is treated as a continuation of the comment even if that line has the format of a control card.

Format Requirements for Control Cards. Input may be either variable format, with the sequence field as the first eight characters of the record (as in TSO CLIST or PLI), or as fixed-format 80-byte records with the sequence field as the last eight characters in the record (as in DATA- or CNTL-type datasets). Note that multiple "$" control cards may be specified on the same line with a single "$" in column 1. For example, either of the following formats may be used:

```
1.) $KEY(SYS9)
    $PREFIX(SYS*)
    $USERDATA(user-comments)

2.) $KEY(SYS9) PREFIX(SYS*) USERDATA(user-comments)
```

Comment Cards

Comment cards, denoted by an asterisk (*) in column 1, allow you to place any text inside the uncompiled rule set. The text on a comment card will be stripped off the compile/decompile sequence.

Access Rule Entries

Individual access rule entries follow the control cards in a rule set.

A rule entry may extend up to 72 positions and may be continued from one card to the next by use of a dash (-). If a line ends with a dash, the next line is interpreted as a continuation of that previous line. If a comment ends with a dash, the next line will begin a continuation of that comment.

You should generally get into the practice of starting rule entries in column two, so that when you have an entry beginning with an asterisk it will not be treated as a comment line.

The full syntax of an individual access rule entry is as follows:

```
dsn VOL(volser) UID(uid-string) SOURCE(source-id) SHIFT(shift) -
LIB(lib) PGM/PROG(pgm) DDN(ddname) UNTIL(date)/FOR(days) -
READ(r) WRITE(w) ALLOC(a) EXEC(e) DATA(text) -
NEXTKEY(next-key)
```

The syntax of an individual rule entry is described below.

Parts of an Access Rule Entry. Each ACF2 access rule contains three parts:

- The environment specifies what is to be accessed, who can access it, and under what conditions the access can take place. The parameters that specify the environment are the dataset-name mask, VOL, UID, SOURCE, SHIFT, LIB, PGM (or PROG), DDN, UNITL, FOR, and DATA.

- The access permission specifies whether the access will be allowed logged or prevented, as explained later.

- The pointer is an optional part that allows an access rule set to work in conjunction with other rule sets. The pointer involves the NEXTKEY parameter.

Parameters in an Access Rule Entry. The parameters that make up the parts of an access rule entry are:

dsn

(Required parameter) This parameter specifies the name of the dataset(s) to which the rule pertains; however, the high-level index is omitted. (The high-level index is specified on the $KEY or the $PREFIX control card.

For example, WORK.MASTER would represent the name of the dataset PAYROLL.WORK.MASTER. (The $KEY or the $PREFIX control card would specify the high-level index PAYROLL.)

A dataset name can have from 1 to 22 levels of qualifiers. Each level must begin with an alphabetic character or the character @, $, or #, with 1 to 8 characters in all. The entire dataset name, including periods, can contain up to 44 characters,

This parameter can contain a dataset-name mask, as explained later in this chapter in the section on masking.

VOL(volser)

(Optional) A mask for the specific volume or set of volumes on which the dataset must reside in order for this rule to apply. If omitted, any volume will be considered.

UID(uid-string) -

(Optional) A mask identifying the set of users to which this rule should apply. If omitted, the entry applies to all users of the system. Refer to the section on masking.

SOURCE(source-id)

(Optional) This parameter specifies a logical or physical input source or source group name for which this rule should apply. If omitted, any input source will be valid. Contact your security officer for a list of valid source group names. This field is non-maskable.

SHIFT(record-name)

(Optional) This parameter specifies the name of the shift record that applies to this rule. This record name defines the allowable days, dates, and times for access under this rule. This field is non-maskable.

LIB(lib)

(Optional) This parameter specifies a name or mask identifying a single library or set of libraries from which a program must execute in order for the access rule to apply. If you do not enclose the name or mask within single quotes, then, when evaluating access, ACF2 will prefix that name or mask with the high-level index specified on the $KEY of this $PREFIX control card. For instance, if you specify LIB(PROGLIB) in the rule set with control card $KEY(PAYROLL), the compiler will assume PAYROLL.PROGLIB as the library name. If the LIB parameter is omitted, any library is included in the environment covered by the rule. Also, note that specifying 'SYS1.LINKLIB' covers all libraries in the link list and Link Pack Area (LPA).

| PGM/PROG(pgm-mask)
(Optional) This parameter specifies a mask defining the set of programs (within the set of libraries specified by the LIB keyword) for which this rule will apply. If omitted, any program will be considered matched. May be specified as PROG or PGM.

| DDN(ddname-mask)
(Optional) This parameter specifies a mask identifying the specific ddnames that must be used for this rule to apply. If omitted, any ddname will be allowed.

UNTIL(date)
(Optional) This parameter specifies a Gregorian date specified as mm/dd/yy, yy/mm/dd, or dd/mm/yy depending on an installation option, which will be the last date on which this rule will be considered valid.

FOR(days)
(Optional) This parameter specifies the number of days, starting from the day the access rule set was compiled, for which this rule will be considered valid. The minimum number that may be specified is zero (meaning today) and the maximum number is 365.

DATA(text)
(Optional) This parameter specifies any character string up to 64 characters. This string will be retained with the rule set and formatted when the rule set is decompiled. Your installation may have standards concerning the format of this string. Values in it are not used by standard ACF2 but may be meaningful in your local implementation of ACF2 (via special program exit checking, etc.).

READ(A/L/P)
(Optional) This parameter specifies the letter A, L, or P to indicate the access permission for a read. Before this access permission applies, the actual access attempt must match the environment defined by other parameters of the access rule entry.

The letter codes are defined as:

A	Allow the access
L	Allow the access but log the event
P	Prevent the access

If not specified in the rule entry, this access permission defaults to READ(P).

WRITE(A/L/P)
(Optional) This parameter specifies the letter A, L, or P to indicate the access permission for a write. This parameter works similarly to how the READ parameter does. If not specified in the rule entry, this access permission defaults to WRITE(P).

ALLOC(A/L/P)
(Optional) This parameter specifies the letter A, L, or P to indicate the access permission for an allocate. This parameter works similarly to how the READ parameter does. If not specified in the rule entry, this access permission defaults to ALLOC(P).

EXEC(A/L/P)
(Optional) This parameter specifies the letter A, L, or P to indicate the access permission for an execute. This parameter works similarly to how the READ parameter does. However, its access permission is either the specified value or the value of the READ parameter--whichever designates the most permissive access. (For example, if READ(P) and EXEC(L) are specified, then EXEC(L) will apply.)

NEXTKEY(next-key)
(Optional) This parameter specifies the key of the alternate rule set that should be checked if access to this dataset is denied based on this rule entry. See the section on the use of the NEXTKEY parameter.

Sample Access Rule Sets

We can look at some simplified examples using Nadia Tormell's payroll department. Assume that the payroll information, including salary rates and other confidential information, is contained on the dataset PAYROLL.MASTER.DATA. Nadia wishes to allow only herself and her lead payroll clerk (PAY7777) access to this dataset. Additionally, the clerk will only be allowed access to read the data, not update it, and only for thirty days.

Nadia's rule set might look like this:

```
$KEY(PAYROLL)
 MASTER.DATA UID(FINPAY1234) READ(A) WRITE(A) EXEC(A)
 MASTER.DATA UID(FINPAY7777) READ(A) FOR(30)
```

The $KEY indicates that the high-level index of the dataset is PAYROLL. The first field in each rule entry is the remainder of the dataset name (MASTER.DATA). The UID fields specify the users being given access authority and their allowable types of access, thus the "A" after each READ, WRITE, and ALLOC for Nadia, and only READ for her clerk. The FOR parameter indicates the clerk may have access only for 30 days.

Perhaps Nadia keeps a current project list online for her department in the dataset PAY1234.CURRENT.PROJECTS. She wishes to allow everyone in the Financial department to read this dataset. And to make her job easier, she has authorized her lead payroll clerk (PAY7777) to update the rule set that governs her own datasets (which she indexes using her Logonid PAY1234). Here is a sample of how Nadia's rule set would appear:

```
     $KEY(PAY1234)
     %CHANGE FINCLKPAY7777
      CURRENT.PROJECTS UID(FIN) READ(A)
```

Note that in the above examples, access not allowed (such as WRITE) may have been entered as WRITE(P) to PREVENT write access. However, "P" is the default, so it is not necessary to enter it in the rule.

CREATING ACCESS RULE SETS

You can create access rule sets directly from the terminal or by first building the rule set text in a file.

The general procedure is as follows:

1. If necessary, build a file with the access rule set text. This step can be accomplished through editing.

2. From TSO READY mode, issue the ACF command.

3. With the ACF command active, establish the RULE setting.

4. Compile the access rule set. For direct compilation from the terminal, issue the COMPILE subcommand without a dataset name. The COMPILE subcommand will allow you to enter the control cards and rule entries at the terminal. For compilation from a partitioned dataset (PDS), issue the COMPILE subcommand with the name of the dataset that contains the rule set text.

5. To test the rule set, issue the TEST subcommand. The TEST subcommand can give you an idea of whether the rule set will do the proper validation of dataset accesses.

6. To save the rule set on the Rules database, issue the STORE subcommand. Otherwise, the rule set will eventually be lost.

7. If the rule set is to be resident, then the F ACF2,RELOAD(ruleid) console operator command must be issued in order to dynamically reload the new rule set. Otherwise, the rule is not made resident until the next system IPL. For further information on resident rules, refer to the explanation of the RESRULE record in the chapter on GSO records. Also, refer to the chapter on console operator commands for an explanation of the RELOAD command.

MASKING OF DATASET NAMES AND UID STRINGS

In access rules, masks are allowed to represent multiple dataset names or User Identification (UID) strings.

Dataset Name Masks

In access rules, the masking of dataset names (other than the name used in the $KEY high-level index field) allows an access rule to apply to more than one dataset. For example:

```
$KEY(PAYROLL)
 WORK.- UID(TFINPAY) R(A)
```

In the access rule set PAYROLL, the dataset name mask WORK.- allows the listed rule entry to apply to all datasets with a high-level index of PAYROLL and a second-level index of WORK. Such datasets may include:

```
PAYROLL.WORK.TEST
PAYROLL.WORK.MASTER
PAYROLL.WORK.BACKUP.VER1
```

A dataset name mask can be created through the use of dashes (-) and asterisks (*), as follows. The high-level index ($KEY), however, cannot be masked.

The Dash. A dataset name mask containing a dash must fit one of the following cases:

1. If the dash falls at the end of an incomplete dataset name index, then the dash represents any number of characters that validly complete the index. (An index can be from 1 to 8 characters.) For example:

```
WORK.BA-       can represent:     WORK.BA
                                  WORK.BACKUP
                                  WORK.BAK
```

2. If the dash appears as a separate index in the mask, then the dash can represent any zero or more indexes. For example:

```
WORK.-         can represent:     WORK.TEST
                                  WORK.TEST.VER1

-.TEST         can represent:     WORK.TEST
                                  WORK.VER1.TEST
```

3. If a dash falls between or before any characters within an index, then the dash is literally a dash. For example, W-RK cannot represent WORK.

The Asterisk. A dataset name mask containing asterisks must fit one of the following cases:

1. If the asterisks fall at the end of a partial dataset name index, then the asterisks represent any number of characters from zero to the number of asterisks. For example:

```
WORK.BACK**     can represent:     WORK.BACK
                                   WORK.BACKUP

                     but not:      WORK.BAC
                                   WORK.BACLUP
                                   WORK.BACKUPP
                                   WORK.BACK.L
```

2. If the asterisks form a separate index, then asterisks represent any index (of at least one character) whose length is no greater than the number of asterisks. For example:

```
WORK.****       can represent:     WORK.M
                                   WORK.TST
                                   WORK.BACK

                     but not:      WORK
                                   WORK.BACKUP
```

3. If the asterisks fall between or before any characters of an operand, then each asterisk represents exactly one character. For example:

```
WORK.**ST       can represent:     WORK.TEST
                                   WORK.LIST

                     but not:      WORK.ST
                                   WORK.MASTER
                                   WORK.TEST.M
```

An Asterisk Followed by a Dash. A dataset-name mask containing an asterisk followed by a dash must fit one of the following cases:

1. If the asterisk and dash fall at the end of a dataset name index, then they represent any characters that validly complete the index (as does a dash alone).

2. If the asterisk and dash form a separate index, then they represent exactly one index of at least one character. For example:

```
WORK.*-         can represent:     WORK.M
                                   WORK.TEST

                     but not:      WORK
                                   WORK.BCK.VER1
```

If the dash precedes any asterisks, then the dash is treated literally as a dash while the asterisks are treated as the asterisks of a mask.

UID String Masks

A UID string mask represents more than one UID string, and thus allows an access rule to apply to multiple users. For example, take the following UID string:

```
UID(TFINPAY)
```

This UID string can represent any UID string that begins with the letters TFINPAY and ends with up to any 17 characters. (A valid UID string can contain up to 24 total characters.)

A UID string mask can be defined by omitted ending characters, asterisks (*), or a dash (-), as follows:

Omitted Ending Characters. Any UID string is automatically treated as a mask. For instance, the UID string TFINPAYNLT not only matches itself, but also matches any string that begins with the characters TFINPAYNLT and contains no more than 24 characters.

By omitting characters, a more general UID string mask can be formed. For example, characters can be omitted from the UID string TFINPAYNLT to form a mask that represents all users in the payroll department:

```
UID(TFINPAY)
```

The mask matches any UID string beginning with the characters TFINPAY and containing up to 24 total characters.

The Dash. A UID string mask containing a dash must fit one of the following cases:

1. If the dash falls at the end of a UID string mask, it has the same effect as no dash. For example, the following two UID string masks are equivalent:

   ```
   UID(TFINPAY-)
   UID(TFINPAY)
   ```

2. If the dash is alone, then the UID string represents all valid UID strings:

   ```
   UID(-)
   ```

If the dash falls within the UID string mask, it is treated literally as a dash and cannot represent any other character.

The Asterisk. A UID string mask containing asterisks must fit one of the following cases:

1. Asterisks that fall at the end of the UID string mask have the same effect as a dash or as no asterisks. For example, the following three UID strings are equivalent:

   ```
   UID(TFINPAY-)
   UID(TFINPAY****)
   UID(TFINPAY)
   ```

2. If the asterisks are separate, then the asterisks represent all valid UID strings:

   ```
   UID(****)
   ```

3. If the asterisks fall between or before any characters of a UID string mask, then each asterisk represents exactly one character. For example:

   ```
   UID(TFIN***NLT)
   ```

   The mask TFIN***NLT can match any UID string beginning with the letters TFIN, followed by any 3 characters except nulls, followed by the letters NLT, and then followed by any other characters to form a UID of up to 24 characters.

The Asterisk and Dash Combined. If both asterisk and dashes are contained in a UID string mask, they must fall at the end of the mask. They are equivalent to no asterisks or dashes. For example, the following two UID string masks are equivalent:

```
UID(TFINPAY-*)
UID(TFINPAY)
```

Imbedded Blank Characters. A UID string mask can contain an imbedded blank character. However, as with the dash, the blank character is treated literally as a blank character.

USE OF NEXTKEY

The NEXTKEY parameter of the rule entry enables you to split a very large rule set into several sets or, conversely, to merge several rule sets together. NEXTKEY may also be used to delegate rule maintenance authority via the %CHANGE and %RCHANGE control cards.

The NEXTKEY parameter allows an alternate access rule set to be evaluated when a particular environment applies to the access but the access is prevented. Validation of the access continues with the evaluation of the alternate access rule set. The index of the alternate rule set is specified in the NEXTKEY operand.

Merging Rule Sets

Similar datasets, such as production files, may require similar ACF2 validation. The following rule sets provide an example of the use of NEXTKEY to merge multiple rule sets:

```
$KEY(ACCT01)
DATA.FILE UID(TFINPAYNLT) NEXTKEY(ACCTXX)

$KEY(ACCT02)
DATA.FILE UID(TFINPAYNLT) NEXTKEY(ACCTXX)

$KEY(ACCT03)
DATA.FILE UID(TFINPAYNLT) NEXTKEY(ACCTXX)

$KEY(ACCT25)
DATA.FILE NEXTKEY(ACCTXX)
```

Users matching the UID string mask TFINPAYNLT can be given READ and WRITE access to all of the above Accounting datasets, even though each dataset has a different high-level index. Use the NEXTKEY operand to direct ACF2 evaluation to one main rule set, ACCTXX. That rule set might be written as follows:

```
$KEY(ACCTXX)
$PREFIX(ACCT**)
 DATA.FILE UID(TFINPAYNLT) R(A) W(A)
```

Note that the $PREFIX control card must be contained in the NEXTKEY rule set to ensure that all high-level qualifiers of the rule sets directed to it (e.g., ACCT01, ACCT02, etc.) will match the dataset name patterns specified.

In this accounting file example, special access permission (such as ALLOCATE) to one particular dataset can be specified using NEXTKEY in two ways.

1. The dataset name, such as ACCT04.FILE, can be specified in a rule entry in the alternate ACCTXX rule set by enclosing the dataset name in single quotes as follows:

```
$KEY(ACCTXX)
$PREFIX(ACCT**)
 'ACCT04.DATA.FILE' UID(TFINPAYNLT) R(A) W(A) A(A)
  DATA.FILE UID(TFINPAY) R(A)
```

In the above rule set, only user TFINPAYNLT has ALLOCATE access to ACCT04.DATA.FILE. All other payroll department users, including TFINPAYNLT, have READ access to all accounting datasets.

2. Another method for specifying special permission to a particular dataset is to place a rule entry in the original ACCT04 ıule set to specify READ, WRITE, and ALLOCATE access for the user TFINPAYNLT, while retaining the NEXTKEY rule entry to govern all other access attempts:

```
$KEY(ACCT04)
 DATA.FILE UID(TFINPAYNLT) R(A) W(A) A(A)
 DATA.FILE NEXTKEY(ACCTXX)
```

Dividing Rule Sets

The NEXTKEY feature may also be used to divide a particular high-level index rule set. This dividing may be needed if a rule set is very large (and will exceed the 4K limit), or to delegate rule maintenance (%CHANGE or %RCHANGE) authority.

For example, an installation may have numerous datasets all having the high-level index of PAYROLL:

```
PAYROLL.MASTER.SHOP1   PAYROLL.BACKUP.SHOP1
PAYROLL.MASTER.SHOP2   PAYROLL.BACKUP.SHOP2
PAYROLL.MASTER.SHOP3   PAYROLL.BACKUP.SHOP3
PAYROLL.MASTER.SHOP4   PAYROLL.BACKUP.SHOP4
PAYROLL.MASTER.SHOP5   PAYROLL.BACKUP.SHOP5
```

A single rule set for all of these datasets would be very large. The NEXTKEY feature can be used to divide the rule set for the high-level index PAYROLL into smaller rule sets. The rule set is divided according to the second-level indexes:

```
$KEY(PAYROLL)
 MASTER.SHOP- NEXTKEY(MASTER)
 BACKUP.SHOP- NEXTKEY(BACKUP)
```

In the above rule set, the NEXTKEY parameter specifies the alternative rule sets to be used in validating access to the MASTER and BACKUP files. Two smaller rule sets can then be written as follows:

```
$KEY(MASTER)                      $KEY(BACKUP)
$PREFIX(PAYROLL.MASTER)           $PREFIX(PAYROLL.BACKUP)
%CHANGE TFINPAYDIR                %RCHANGE TFINOPSDIR
 SHOP1 UID(TFINPAYC1) R(A) W(A)   SHOP- UID(TFINOPR) R(A)
 SHOP2 UID(TFINPAYC2) R(A) W(A)
```

```
     SHOP3 UID(TFINPAYC3) R(A) W(A)
     SHOP- UID(TFINPAYMGR) R(A)
```

The rule sets above are smaller than the single rule set for the high-level index PAYROLL. Also, in each of the above rule sets, the $PREFIX control carα is specified so that the true high-level index will be appended to each dataset name.

In the first rule set, the payroll clerks (TFINPAYC1, TFINPAYC2, etc.) in each of the installation's shops need READ and WRITE access to only their particular shop's MASTER files in order to update the files. The Payroll Manager (MGR), however, is given READ access to all shop MASTER files.

In the second rule set, the installation's computer operators (TFINOPR) must have READ access to only the BACKUP files for all shops.

It should be emphasized that ACF2 validation is directed to the rule set specified in the NEXTKEY option only when access based on the current rule set is prevented. A maximum chain of 25 NEXTKEY options may occur. If more than 25 are specified, ACF2 will deny access. An SMF logging record will be written to log the event.

When using the NEXTKEY parameter, you must ensure that "looping" is avoided. In other words, a rule set containing a NEXTKEY parameter cannot be interpreted more than once during a single access validation. ACF2 will issue an error message if a loop condition occurs. In addition, ACF2 will deny the access request and log the event.

Delegation of Change Authority. The NEXTKEY feature also permits %CHANGE and %RCHANGE authority to change a particular rule set.

As in the two previously shown rule sets, the Director of Payroll (TFINPAYDIR) is given authority to change the access rule set for the PAYROLL.MASTER files. This authority is delegated through the %CHANGE control card. The Director of Operations (TFINOPSDIR) is given restricted authority to change only the rule entries for the PAYROLL.BACKUP files. This restricted authority is delegated through the %RCHANGE control card.

ACF2 FEATURES THAT SIMPLIFY ACCESS RULE WRITING

The writing of access rules for all datasets residing on a system may seem to be a substantial undertaking at the time of installation. But keep in mind these three important features of ACF2:

Modes
  Modes allow ACF2 to be integrated into your computer operating system in stages. These stages are designed to provide time for the development and testing of rules and various local ACF2 features.

For information on the modes of ACF2 (QUIET, LOG, WARN, ABORT, and RULE), refer to the explanation of the MODE field of the OPTS record in the chapter on GSO records.

Centralization/decentralization option

Your installation has the ability to centralize or decentralize the different aspects of ACF2 administration. In a centralized environment, the security officer generally has sole responsibility for the writing of access rules. Decentralization grants each user the authority to store an access rule set for datasets that the user owns. This centralization/decentralization feature is controlled through the OPTS record, described in the chapter on GSO records.

Masking

Masking allows an access rule environment to apply to a group of datasets and a group of users. This feature was described previously in this chapter.

ACF SUBCOMMANDS UNDER THE RULE SETTING

You can process access rules after establishing the RULE setting of the ACF command:

```
set rule
```

After establishing the RULE setting, you can issue any of the following ACF subcommands:

```
*
COMPILE      LIST       SET
TEST         DELETE     SHOW
STORE        END        SN
DECOMP       HELP
```

The common subcommands *, END, HELP, SET, SHOW, and SN operate under the RULE and ACF settings as previously explained. The following text describes the function, syntax, and parameters of the other subcommands under the RULE setting. Most of these subcommands work in the same manner under the ACF setting. See the chapter on the ACF setting for a summary of these subcommands under that setting.

COMPILE Subcommand--RULE or ACF Setting

The COMPILE subcommand allows you to create a set of access rules.

Syntax. The syntax of this subcommand is as follows:

```
COMPILE [*/dsname] [LIST/NOLIST] [STORE/NOSTORE] [FORCE/NOFORCE]
        [MAXRULE(nnn)] [ALL]
```

ACF2 provides two ways of compiling access rule sets:

1. Directly at the terminal.

2. From a partitioned dataset (PDS). STORE is not the default in this instance, STORE is the default when compiling all members of a PDS.

Compiling Directly at the Terminal. You can enter an access rule set directly from the terminal by first entering the COMPILE subcommands without parameters:

```
acf
set rule
compile
```

ACF2 will respond with:

```
ACF70010 ACF COMPILER ENTERED
```

Begin by entering the $KEY control card on the first line. Enter the other control cards and then the rule entries. Begin each rule entry in column 2, on a separate line. Press the ENTER or return key after each line.

```
acf
set rule
compile
ACF70010 ACF COMPILER ENTERED

$key(payroll)
  work.master uid(tfinpaynlt) r(a) w(a) e(a)
  work.backup uid(tfinpayiso) r(a) w(l) e(a)
```

To end the rule set, enter a blank line or type END in column 1 and press enter. After compiling the access rule set, you can use the TEST subcommand to get an idea of whether the access rule set allows for proper access validation. To retain the access rule set, you must use the STORE subcommand.

Compiling from a Partitioned Dataset (PDS). You can also create access rule sets by first entering the control cards and rule entries into a PDS member. Each control card or rule entry must be on a separate line. The last line does not have to be a blank line. For example:

```
$KEY(PAYROLL)
 WORK.MASTER UID(TFINPAYNLT) R(A) W(A) E(A)
 WORK.BACKUP UID(TFINPAYISO) R(A) W(L) E(A)
```

After entering the control cards and rule entries into the PDS member, you can invoke ACF2 and compile the access rule set. Issue the COMPILE subcommand with the name of the PDS and member. For example:

```
acf
set rule
compile work.text(rule)
```

In the COMPILE subcommand, the PDS name is specified according to TSO conventions. (Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYNLT.WORK.TEXT(RULE)' or simply WORK.TEXT(RULE). To compile all members of a PDS, specify WORK.TEXT ALL. This causes the members to be automatically stored.

Using "Ditto" for Entering Access Rules. A ditto mark (") provides a convenient method of using any of the parameters and values in the previous rule entry. For example:

```
acf
compile
ACF70010 ACF COMPILER ENTERED

$key(payroll)
 work.master uid(tfinpay***) r(a)
 work.backup uid(") r(")
 work.test uid(") r(")
end
```

In this example, rule set named PAYROLL, the first rule entry applies to the dataset PAYROLL.WORK.MASTER. Read access to that dataset is allowed for any user whose UID string fits the mask TFINPAY***. The second rule entry contains ditto marks for the UID and READ parameters. Thus, read access to PAYROLL.WORK.BACKUP is allowed for any user whose UID string fits the mask TFINPAY***. The third rule entry also has ditto marks for the UID and READ parameters. Thus, read access to PAYROLL.WORK.TEST is allowed for any user whose UID string fits the mask TFINPAY***. The UID and READ parameters and any values are "dittoed" from the second rule entry.

The ditto feature eliminates the need for you to reenter parameter values if they are the same as in the previous rule entry. You can use this feature to repeat the entire dataset name as well as other parameter values. However, you cannot ditto the parameter name itself (such as UID, READ, LIB, etc.). You can only ditto the parameter value. Up to two dataset name levels can be dittoed.

Parameters. Under the RULE or ACF setting, the COMPILE subcommand takes the following parameters:

*

An asterisk indicates that the text that follows will be input to the compiler. In an online environment, the system will prompt you to enter the access rule text directly from the terminal. In batch, the cards following the COMPILE card will be assumed as input.

(no parameters)

Use of the COMPILE subcommand without parameters is equivalent to specifying an asterisk.

dsname

A partitioned dataset (PDS) and member name specifies the PDS and member that contains the access rule text to be compiled. The PDS name follows TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYNLT.WORK.TEXT(RULE)'.

If you do not specify a member name, ACF2 will prompt you for one. To compile input from all PDS members, specify the ALL parameter. An access rule set cannot be compiled from a sequential dataset. When ALL is specified, ACF2 will do an automatic STORE.

LIST/NOLIST

The LIST parameter causes the input to the compiler to be displayed on your screen or printed on your listing during compilation of a rule set. NOLIST causes no such display or printed list. LIST is the default.

STORE/NOSTORE

The STORE parameter causes the rule set to be automatically stored at compilation time. NOSTORE causes no automatic storing of the rule set; you must issue the STORE subcommand to store the rule set. STORE is the default if you are using the ALL parameter to compile all members of a PDS. Otherwise NOSTORE is the default.

FORCE/NOFORCE

The FORCE parameter allows the access rule set to be stored regardless of whether it currently exists. NOFORCE allows the access rule set to be stored only if it does not already exist. FORCE is the default.

Note that when used as parameters of the COMPILE subcommand, STORE/NOSTORE and FORCE/NOFORCE apply only to the current compilation of this particular access rule set. When used as parameters of the SET subcommand, STORE/NOSTORE and FORCE/NOFORCE are in effect until they are changed or until the ACF command is ended.

MAXRULE

The MAXRULE parameter specifies a number from 0 through 999 that limits the size of the rule set that you can input to the compiler. This number is a scaling factor and has no direct correlation with the number of rules in a rule set. (Different rules require different amounts of space.) The default is 250. If a rule set compiles correctly but encounters space problems when you try to store it, try specifying MAXRULE with a value less then 250. Alternatively, if a rule set contains a lot of duplication (i.e., input source names, etc.), you can specify a higher value for MAXRULE to allow the compiler to accept a larger rule set. However, using the NEXTKEY parameter in your access rules is the preferable way to handle large rule sets.

ALL

The ALL parameter causes compilation and storing of access rule sets from all members of a specified partitioned dataset (PDS). For example:

```
acf set rule compile work.text all
```

If any members of a PDS do not contain an access rule set, then do not specify this parameter.

TEST Subcommand--RULE or ACF Setting

The TEST Subcommand allows you to interactively test a compiled access rule set. This testing can give you an idea of whether the access rule set provides for the proper validation of dataset and volume accesses.

The Extent of Access Rule Testing. While the TEST subcommand is active, only access rule interpretation is done. This testing does not take into account any installation-specific system options or attributes of the Logonids being tested. Nor does this testing take into account any exits. The test subcommand is simplified because of the impossibility of testing all possible exit combinations.

Syntax. The syntax of the TEST subcommand is:

```
TEST [*/ruleid]
```

Example. Suppose an access rule set with the key PAYROLL were just compiled:

```
ACF70010 ACF COMPILER ENTERED

$key(payroll)
 work.master uid(tfinpaynlt) r(a) w(a) e(a)
 work.backup uid(tfinpayiso) r(a) w(l) e(a)

ACF70051 TOTAL RECORD LENGTH= 162 BYTES, 3 PERCENT UTILIZED
```

The TEST subcommand could then be issued:

```
test *
.
```

When the period(.) is displayed, the TEST subcommand will be active. You can enter any of the TEST subcommand keywords to specify the particular environment that you want to test. An asterisk entered in place of a parameter such as DSN(*) causes the previous parameter to be repeated. For example, the following keywords test whether the access rule set PAYROLL will allow the user TFINPAYNLT to access the dataset PAYROLL.WORK.MASTER:

```
test *
.

dsn(work.master) uid(tfinpaynlt)
```

After the TEST subcommand keywords are entered, the system responds by displaying all of the current values that describe the environment being tested. At the bottom of the display, is an indication of whether the access will be allowed, logged, or prevented:

```
test *
.

dsn(work.master) uid(tfinpaynlt)
ACF71014 THE FOLLOWING PARAMETERS ARE IN EFFECT:
 DDN=******** UID=TFINPAYNLT DATE=11/10/84 SOURCE=********
 VOL=******** DSN=PAYROLL.WORK.MASTER
 PGM=******** LIB=***.***
 TIME=*******

 THE FOLLOWING WOULD APPLY: READ(A) WRITE(A) ALLOC(P) EXEC(A)
```

In this example, the result is that this user will be allowed read, write, and execute authority, but not allocate authority.

After a result is displayed, you can make another entry of keywords and values to specify another environment for testing. The END subcommand will terminate the TEST subcommand.

How the Values Describing the Environment Work. After you enter values describing the test environment, these values remain in effect until they are specifically changed.

Any values that have NOT been specified are assumed to be completely masked--the default. For instance, if you specify no UID keyword, then the subcommand will test whether all UIDs are allowed access. However, the DSN( ) parameter must be specified.

Parameters. The TEST subcommand, when issued under the RULE or ACF setting, takes the following parameters:

*

An asterisk indicates that the previously compiled access rule set should be tested.

(no parameter)

When specified without a parameter, the TEST subcommand operates the same as when an asterisk is specified.

ruleid
The ruleid identifies the key of the access rule set to be tested. To specify an access rule set by its ruleid, either authorization to update the access rule set or the AUDIT privilege level is required.

TEST Subcommand Keywords. After you have issued the TEST subcommand along with any of the parameters described above, the TEST subcommand will become active. You can specify a test access environment by entering any of the following keywords with the appropriate values:

DSN(dsname-mask)
The DSN keyword is specified with the name of the dataset for which access will be tested. This name can be masked.

VOL(volser-mask)
The VOL keyword is specified with the serial number (volser) of the volume to which access will be tested. The volser can be masked.

LID(logonid-mask)
| Specifies a Logonid identifying a single user whose access is to be tested. To specify this keyword, the user making the request needs the appropriate access to the Logonid record of the user whose access is being tested.

UID(uid-mask)
The UID keyword is specified with the UID string identifying the user whose access will be tested. This UID string can be masked. To specify this keyword, the user making the test does not need access to the Logonid record of the user whose access is being tested. If both LID and UID are specified, then the last LID or UID value specified will used. For example, if LID(PAYJJD) UID(TFINPAYNLT) is specified, then UID(TFINPAYNLT) will only be used.

LIB(libname-mask)
The LIB keyword is specified with the name of a library. Any program through which the access is being made must reside in this library. This library name can be masked.

PGM(pgm-mask)
The PGM keyword is specified with the name of the program through which the access is being made. The keyword PROG can be used instead of PGM.

DDN(ddname)
The DDN keyword is specified with the ddname associated with the dataset being accessed.

DATE(date)
The DATE keyword is specified with the date for which access will be tested. This date can be in the format mm/dd/yy, yy/mm/dd, or

dd/mm/yy. The appropriate format is specified in the DATE field of the OPTS record. See the chapter on GSO records. The TEST subcommand uses the current date as the default.

SOURCE(source-id)
The SOURCE keyword is specified with the physical or the logical name of the input source or source group for which access will be tested The identification of input sources is discussed in the chapter on entry records.

TIME(hhmm)
The TIME keyword is specified with the time (in hours and minutes) for which access will be tested.

These keywords must be separated by blanks and can be specified on one or more input lines.

TEST Subcommand Results. The results of the TEST subcommand show the read, write, allocate, and execute authority as granted by the test access rule set. The codes associated with the results are:

A	Access would be allowed
L	Access would be allowed but logged
P	Access would be specifically prevented

If no rule entry specifically applies to the test access environment, the following message is displayed:

```
NO RULES APPLY, ACCESS WOULD BE DENIED.
```

If a test access attempt is prevented but matches a NEXTKEY environment defined in the rule set, then ACF2 displays the ruleid for the next rule set to be interpreted.

At this point, validation of the test access attempt ends. ACF2 prompts for TEST subcommand keywords and values for the next test access attempt. The END subcommand will terminate the TEST subcommand.

To test the NEXTKEY rules, you must test the rule set pointed to by the NEXTKEY entry.

STORE Subcommand--RULE or ACF Setting

The STORE subcommand, under the RULE or ACF setting, allows you to store the previously compiled set of access rules.

The syntax of the STORE subcommand is simply:

```
STORE
```

This subcommand accepts no parameters.

If SET NOFORCE has been previously issued, then the access rule set is stored only if it does not already exist. You will receive a message if an access rule is not stored. The NOFORCE parameter of the SET subcommand is explained in the chapter on the ACF command.

The store operation may be rejected because of insufficient authority for storing access rule sets.

DECOMP Subcommand--RULE or ACF Setting

The DECOMP subcommand, under the RULE or ACF setting, decompiles an access rule set that has been previously compiled and stored. This subcommand is useful for examining, updating, or changing access rule sets. An access rule set can be decompiled at the terminal or into a member of a partitioned dataset (PDS). The LIST subcommand (under the RULE setting) can also be used to accomplish the same function as DECOMP when used under the RULE setting.

Syntax. The syntax of the DECOMP subcommand is:

```
DECOMP {*/ruleid/LIKE(ruleid-mask)} [INTO(dsname)]
```

Parameters. The DECOMP subcommand takes the following parameters:

*
An asterisk allows you to decompile the last individual access rule set that you processed since establishing the RULE or ACF setting.

ruleid
The ruleid specifies the key of an individual access rule set to be decompiled or listed.

LIKE(ruleid-mask)
The LIKE parameter allows you to specify a mask of ruleids for decompiling or listing a group of access rule sets.

INTO(dsname)
The INTO parameter allows you to specify the data set into which the rule set will be decompiled. This data set must be a PDS. No other types will qualify. If specifying a fully-qualified dataset name (i.e., including the high-level index), you must enclose that name within single quotes. For example:

```
DECOMP PAYKLN INTO('PAYJSD.WORK.RULE')
```

If you specify no member name, the key of the access rule set is used. However, if the member name is invalid, then a member name is automatically generated. For further information on the automatic generation of this member name, see the explanation of the SET MEMBER subcommand in the chapter on the ACF command.

DELETE Subcommand--RULE Setting only

The DELETE subcommand, under the RULE setting, allows for deletion of access rule sets. By default, this subcommand also deletes any Logonid record corresponding to the key of the deleted access rule set.

Only users with the SECURITY privilege level can delete access rule sets. This authority can be restricted through the use of scopes.

Example. For example, the following subcommand deletes the Logonid record and corresponding access rule set for the Logonid PAY7777:

```
     delete PAY7777
```

After the DELETE command is issued, the message DELETED is returned.

Syntax. The DELETE subcommand can follow one of two syntax formats:

```
     DELETE RULE(ruleid)

     or

     DELETE {*/ruleid/LIKE(ruleid-mask)/UID(uid-mask)} [RULE(ruleid)]
```

Parameters. The DELETE subcommand, under the RULE setting, takes the following parameters:

*
An asterisk allows you to delete the last access rule set that you referenced in your current TSO session. The corresponding Logonid record is also deleted.

ruleid
An individual ruleid allows you to delete one, specific access rule set and its corresponding Logonid record.

UID(uid-mask)
The UID parameter allows you to specify a mask for the UID strings identifying the users whose Logonid records and corresponding access rule sets that will be deleted.

RULE(ruleid)
The RULE parameter identifies an individual access rule set to be deleted. The corresponding Logonid record will not be deleted.

ACF2 UTILITIES FOR ACCESS RULE PROCESSING

In addition to the ACF subcommands and ISPF screens, ACF2 also provides the following utilities for access rule processing. These utilities are described in detail in the acf2/MVS Utilities Manual:

ACFCOMP TSO command
The ACFCOMP command can be issued from TSO READY mode. It is similar to the ACF COMPILE subcommand, and allows you to compile access rule sets directly at the terminal or from a PDS member.

ACFBCOMP batch utility
The ACFBCOMP utility allows you to compile an access rule set in the batch environment. The control cards and rule entries can be input as card data.

ACFNRULE command/batch utility
ACFNRULE can be used as a batch program or a TSO command. It allows you to add or delete individual access rules. Multiple access rules containing the same character string can be located, optionally verified, and then deleted.

ACFBATCH batch utility
The ACFBATCH utility allows you to execute the ACF subcommands in the batch environment.

RULE SELECTION ALGORITHM

The rule compiler converts the input into a form which can be scanned by the rule interpreter and then used for verification checking. In addition to doing this, the rule compiler orders the rules according to the following criteria. However, when $NOSORT is specified, the rules will remain in the exact order entered.

* DSN patterns from most specific to most general

* VOL patterns from most specific to most general

* UID patterns from most specific to most general

* SOURCE operands in alpha-order, with 'not specified' last

* SHIFT operands in alpha-order, with 'not specified' last

* LIB patterns from most specific to most general

- * PGM/PROG patterns from most specific to most general

- * DDN patterns from most specific to most general

- * UNTIL dates from earliest to latest

The first rule entry that matches the actual dataset, volume, user identification string, source, shift, library, program, and date being used (i.e., the defined "environment") will be the rule entry used to determine the access privileges.

The compiler recognizes duplicate patterns in adjacent rules so that there is not a great deal of storage or processing overhead in having many rules that differ only in UID, LIB, or PGM patterns.

TSO CLIST CONSIDERATIONS

ACF2 protection of TSO CLISTs (Command Lists) is handled through dataset access rules and the control statement in the CLISTs. In an access rule entry, the CLIST library name to be accessed is specified as the DSN parameter. This rule entry grants specific access to CLISTs residing on that library. For example, to provide execute-only access to CLISTs residing in library TEST.CMD.CLIST, a sample access rule would be:

```
$KEY(TEST)
 CMD.CLIST UID(ABC) EXEC(A)
```

Note that if a user specifies a CONTROL LIST instruction in the CLIST, the contents of the CLIST will be listed during execution even if a user has execute-only access. To prevent this situation from occurring, installations should insert the NOLIST and NOCONLIST operands in sensitive CLISTS.

GENERATION DATA GROUPS

Access to generation data groups (GDGs) may be provided by use of masking in the dataset name within the rule. For example, to allow access to all generations of a payroll dataset called PAYROLL.HOURS, the rule set might be:

```
$KEY(PAYROLL)
 HOURS.- UID(PAY) READ(A)
```

If a rule for only one specific generation is to be defined, specify the full dataset name, such as:

```
$KEY(PAYROLL)
 HOURS.G0015V00 UID(PAY) READ(A)
```

SECURED VOLUME ACCESS RULES

An installation can protect datasets at the volume level through a secured volume list. On that list, the installation specifies the volume serial numbers (VOLSERs) for the datasets to be protected.

Once a volume has been specified on the secured volume list, the installation can establish either:

* An access rule set for each secured volume

* One access rule set for all secured volumes. An example of each method of establishing these access rules is given below.

Protection at the volume level does not exist for a given volume until that volume is specified on the secured volume list.

Validation of Secured Volume Accesses

To validate a dataset access request to a secured volume, ACF2 generates a "pseudo dataset" name. This name will have a format of either @volser.VOLUME or VOLUME.@volser. In this format, "volser" represents the volume serial number, and "VOLUME" is literally the word VOLUME.

Sample Access Rule Set for a Single Secured Volume

The installation may choose to establish an access rule set for each secured volume. To validate an access request to a volume with the serial number TEST01, ACF2 will generate a pseudo dataset name of @TEST01.VOLUME to evaluate a rule set such as:

```
$KEY(@TEST01)
 VOLUME UID(TFINPAYJSD) R(A) W(A)
 VOLUME UID(TFINACTKLW) R(A)
```

Note that the access rule set key is the volume serial number, and each access rule entry begins with the word VOLUME.

Sample Access Rule Set for All Secured Volumes

Alternatively, the installation may choose to establish one access rule set for all secured volumes. For a volume with the serial number TEST01, ACF2 will generate a pseudo dataset name of VOLUME.@TEST01 to evaluate a rule set such as:

```
$KEY(VOLUME)
 @TEST01 UID(TFINPAYJSD) R(A) W(A)
 @TEST01 UID(TFINACTKLW) R(A)
 @TEST02 UID(TFINACT) R(A)
 @TEST03 UID(TFINFINCRH) R(A) W(A)
```

Note that, in this case, VOLUME is the key of the access rule set, and a volume serial number (or volume serial number mask) begins each access rule.

Further Information on Control of Secured Volume Accesses

For further information on the control of secured volume accesses, refer to the chapter on GSO records. To specify the list of secured volumes, refer to the explanation of the SECVOLS GSO record. To specify whether the installation will be using one or more than one access rule set for secured volumes, refer to the explanation of the VOLRULE field of the OPTS GSO record.

VSAM ALLOCATION CONSIDERATIONS

VSAM-generated dataset names and related VSAM dataspaces may require some special considerations for allocation processing. For this reason, it is recommended that both the data and index portions of the VSAM cluster be named with additional dataset name qualifiers of DATA and INDEX, respectively. For example:

```
cluster-name:  A.B.C
   data-name:  A.B.C.DATA
  index-name:  A.B.C.INDEX
```

For read or write access of a VSAM cluster, authorization is required only for the cluster name. Normal dataset name access rules are therefore sufficient for normal user processing. For allocate access, authorization is also required for the data and index names. In the following example, the user X can read and write to the cluster, and user Y can define, delete, and alter the cluster.

```
$KEY(A)
 B.C UID(X) READ(A) WRITE(A)
 B.C.- UID(Y) ALLOC(A)
```

Protection of VSAM Dataspaces

Names are not given to VSAM dataspaces. Therefore, when a VSAM dataspace or pagespace is created or deleted, ACF2 will validate access at the volume level.

For dataspaces, VSAM generates an internal name such as Z9999999x. ACF2 recognizes the VSAM naming convention and will generate a pseudo dataset name in the format @volser.VOLUME or VOLUME.@volser. Then, ACF2 searches for a volume rule that will allow access.

For example, either of the following access rule sets will allow the user PAYSDH to define or delete VSAM dataspaces on the volume VSAM01:

```
$KEY(@VSAM01)
 VOLUME UID(TFINPAYSDH) ALLOC(A)
```

or

```
$KEY(VOLUME)
 @VSAM01 UID(TFINPAYSDH) ALLOC(A)
```

For further information, see the previous section on secured volume access rules.

VTOC RULES

If the VTOC (Volume Table of Contents) is used as a dataset (referenced as a dataset and opened for processing), a pseudo dataset name of SYSVTOC.volser is generated by ACF2 for validating these VTOC requests. The Access Rule database is then searched for a SYSVTOC rule. A sample VTOC rule set is as follows:

```
$KEY(SYSVTOC)
 WORK01 UID(ABCD) R(A) W(A)
```

ACF SETTING

When first issued, the ACF command defaults to the ACF setting. (The system may respond by displaying either the message ACF or a question mark (?). This is controlled by the MODE attribute in the user's Logonid record. If MODE is on, ACF is displayed, otherwise ? is displayed.

```
acf

ACF
```

Often referred to as a combination of the LID and RULE settings, the ACF setting allows for:

1. Logonid record processing

2. Access rule set processing

3. Use of the common ACF subcommands

The following sections explain the similarities and differences between the operation of the ACF subcommand under the ACF setting as opposed to the LID or RULE setting.

LOGONID RECORD PROCESSING UNDER THE ACF SETTING

The following subcommands allow for Logonid record processing under the ACF setting:

```
INSERT      LIST        DELETE
CHANGE      SYNCH
```

When compared to ACF command operation under the LID setting, ACF command operation under the ACF setting is the same for all of the above subcommands except DELETE. Operation of the DELETE subcommand is summarized below. For further information on the other subcommands, see the chapter on Logonid records (LID setting).

ACCESS RULE PROCESSING UNDER THE ACF SETTING

The following subcommands allow for access rule processing under the ACF setting:

COMPILE
DECOMP
TEST
END
STORE
DELETE

When compared to ACF command operation under the RULE setting, ACF command operation under the ACF setting is the same for all of the above subcommands except DELETE. See the explanation of the DELETE subcommand below. For further information on the other subcommands, see the chapter on access rules (RULE setting).

Note that, under the RULE setting, the END subcommand can be used to end the TEST subcommand as well as end the ACF command.

THE COMMON SUBCOMMANDS UNDER THE ACF SETTING

The following subcommands can be issued under the ACF setting:

END
HELP
SET
SHOW
SN

These common subcommands operate as described in the chapter on the ACF command. Note, however, that the END subcommand can also be used to terminate the TEST subcommand.

OTHER SUBCOMMANDS UNDER THE ACF SETTING

Under the ACF setting, most subcommands operate the same as under the LID or RULE setting, as mentioned previously in this chapter. However, the DELETE subcommand operates differently.

DELETE Subcommand--ACF Setting

Under the ACF setting, DELETE subcommand allows you to delete either Logonid records or access rules, or both.

Syntax. The DELETE subcommand has the following syntax:

```
DELETE {*/logonid/LIKE(logonid-mask)/UID(uid-mask)/
       [RULE(ruleid)]} [NORULE]
```

Note that the DELETE subcommand under the LID setting contains no RULE parameter. The DELETE subcommand under the RULE setting contains no UID or NORULE parameter.

Example. The following subcommand deletes the Logonid record, but not the corresponding access rule set for the Logonid PAY7777:

```
delete pay7777 norule
```

The following subcommand deletes the access rule set that has the key PAY7777. This subcommand does not delete the corresponding Logonid record:

```
delete rule(pay7777)
```

Parameters. The DELETE subcommand takes the following parameters:

*
An asterisk allows you to delete the last Logonid record and/or access rule set that you referenced in your current TSO session.

logonid
An individual Logonid allows you to delete one, specific Logonid record and/or access rule set.

LIKE(logonid-mask)
The LIKE parameter allows you to specify a mask for the Logonids whose records and/or corresponding access rules you want to delete.

UID(uid-mask)
The UID parameter allows you to specify a mask for the UID strings identifying the users whose Logonid records and/or corresponding access rule sets will be deleted.

NORULE
The NORULE parameter indicates that although the Logonid record(s) specified will be deleted, the corresponding access rules will not be deleted. The default is for deletion of any access rule set whose key matches the Logonid of any deleted record.

RULE(ruleid)
The RULE parameter allows for deletion of only the access rule set whose ruleid matches the specified Logonid. The Logonid record itself is not deleted. With the use of the LIKE or UID parameter, multiple access rule sets can be deleted.

RESOURCE SETTING: GENERALIZED RESOURCE RULES

In addition to controlling access to the computer system and data, ACF2 provides protection for other system resources. This protection is through generalized resource rules, which have similarities to access rules. This chapter discusses:

- An example of a generalized resource rule
- Types and names of generalized resource rule sets
- The elements of the generalized resource rule set
- Masking in generalized resource rule sets
- ACF subcommands, under the RESOURCE setting, for processing generalized resource rules

EXAMPLE OF A GENERALIZED RESOURCE RULE SET

When listed, a simple generalized resource rule set might look like:

```
$KEY(ACFM) TYPE(CKC)
 UID(TFINTEC) ALLOW
```

This rule set can be interpreted as follows:

1. $KEY(ACFM) indicates that the rule set pertains to the ACF2/CICS transaction named ACFM. (This transaction allows for ACF2 security administration through CICS.)
2. TYPE(CKC) defines the type of rule set. This particular rule set pertains to a CICS transaction.
3. UID(TFINTEC) identifies all users whose UID strings begin with TFINTEC.
4. ALLOW specifies that the previously identified users be allowed access to the ACFM transaction.

Generalized resource rule sets are generally larger than this sample rule set. For example:

```
$KEY(ACFM)
 UID(TFINPAYISO) ALLOW
 UID(TFINFINISO) LOG
 UID(TFINADMISO) UNTIL(11/24/85) ALLOW
 UID(TFINPAYNLT) SHIFT(REGULAR) ALLOW
```

The various parameters that you can specify in generalized resource rule sets are explained later in this chapter.

TYPES AND NAMES OF GENERALIZED RESOURCE RULE SETS

Generalized resource rule sets are identified by resource type, and then by name, as follows.

Types of Generalized Resources

A generalized resource type is identified by a type code made up of three alphanumeric characters (for example, CKC for CICS transactions). The ACF2 system pre-defines several resource type codes that cannot be changed locally:

SAF	The type code used for System Authorization Facility (SAF) resource validation other than datasets, disk, or tape volumes.
TAC	Time Sharing Option (TSO) account resource rule sets
TPR	Time Sharing Option (TSO) procedure resource rule sets

Additional resource type codes are predefined by ACF2, but can be renamed locally including:

CFC	CICS file control rule sets
CKC	CICS transaction control rule sets
CPB	CICS DL/I request rule sets
CPC	CICS program control rule sets
CTD	CICS transient data rule sets
CTS	CICS temporary storage rule sets
DAT	IDMS area control rule sets
IAG	IMS application group name rule sets
ITR	IMS transaction control rule sets
PGM	IDMS program control rule sets
PGN	IDMS non-protected program control rule sets
SSC	IDMS subschema rule sets
TSK	IDMS task control rule sets

Generalized Resource Names

An individual generalized resource is assigned a name that is unique from the names of all other resources of the same type. This name can be from one to forty characters long, and is used to identify the resource being protected. In the previous example, the name ACFM identified a CICS transaction.

Resource directories for each specific resource type may be built. A directory contains an entry for each generalized resource type. With the use of this directory, more than one individual resource can be conveniently named in a generalized resource rule using masking techniques. This will be explained later in this section and in the RESDIR Section for GSO records.

THE GENERALIZED RESOURCE RULE SET

A generalized resource rule set looks similar to an access rule set, except it is associated with a type code and has some structural differences. A generalized resource rule set is made up of:

1. Control cards, each beginning with a $ or % symbol

2. Rule entries, which follow all control cards

The syntax of the generalized resource rule set is:

```
$KEY(resource-name)
$TYPE(type-code)
$NOSORT
$USERDATA(text)
%CHANGE uid-mask,uid-mask,...
 [UID(uid-mask)] [SOURCE(source-id)] -
 [SHIFT(record-name)] [UNTIL(date)] [FOR(days)]-
 [SERVICE(READ,UPDATE,ADD,DELETE)] -
 {ALLOW/LOG/PREVENT} [DATA(text)] [VERIFY]
```

The first 5 lines above show the syntax for each of the various types of control cards. The $KEY is the only required control card. The indented lines show the syntax for a single rule entry. Usually, a rule set will contain several entries.

Comment Cards. In addition, a generalized resource rule set can contain comment cards, as described after the explanations of control cards and rule entries.

Continuation of Control Cards, Rule Entries, or Comments. Any card or entry may be continued by use of a dash (-) as the last non-blank character on the card. Any continuation is recognized unconditionally. For example, if a %CHANGE control card ends with a dash, the next card will be treated as a continuation of the %CHANGE card, even if it is another control card or a comment.

Format of Rule Set Text. Rule set text may be either variable format with the sequence field as the first eight characters of each record (as in TSO CLIST or . PLI), or fixed-format eighty-byte records with the sequence field as the last eight characters in the record (as in TSO DATA or CNTL datasets).

Control Cards for Generalized Resource Rule Sets

Control cards identify the rule set and determine some rule set characteristics. Identified by the $ or % symbol in column 1, all control cards must precede the rule entries.

The $KEY control card is the only required control card. Following that control card, only one of each type of $ control card can be specified. However, more than one of each type of % control card can be specified.

The types of control cards are:

$KEY(resource-name) TYPE(type-code)

Indicates the resource name being compiled. This name can be up to forty characters. It can be masked if the directory for the specified type of resource is made resident. (See the section on masking.) However, this key should never be specified as $KEY(-). In addition, a dash(-) cannot be used to partially mask a high-level index. Only asterisks(*) can be used to mask the $KEY field and must correspond to the exact number of positions in that field. See the section further on in this manual, "Masking in Generalized Rule Sets".

The $KEY control card also specifies the three-character type code that identifies the type of generalized resource being protected. Type codes were discussed earlier in this chapter. The type code can also be entered as a $ Control card.

$NOSORT

This control card specifies that no sorting will be done with the rules in the generalized resource rule set being stored. Without this control card, ACF2 sorts the rules from specific to most general in terms of access environment. See the last section of this chapter. A warning message is issued whenever a rule set containing a $NOSORT card is compiled. This control card should be used with caution, since ACF2 will always use the first rule that matches the access environment. Use of the $NOSORT card requires setting of the NOSORT field in the OPTS record (type GSO) of the Infostorage database.

Alternatively, you may specify a NOSORT parameter on the $KEY control card to achieve the effect of the $NOSORT card (in generalized resource rule sets only). For example:

```
$KEY(ACFM) TYPE(CKC) NOSORT
```

$TYPE(type-code)

This control card provides any alternative way of specifying the resource type. The resource type is generally specified on the $KEY control card, as discussed above.

$USERDATA(text)

This control card may contain any string of text of up to 64 characters. This string is passed to the installation exits and the calling application and can indicate further checking or control information.

%CHANGE uid-mask,uid-mask,...

This control card specifies a mask (or masks) for the UID strings of those users who can store and delete this rule set. Note that the user or users given "change authority" may further delegate this authority to other users.

Rule Entries for Generalized Resource Rule Sets

Individual generalized resource rule entries follow the control cards in a rule set. Each rule entry specifies an environment and access permission under which access to a particular resource can take place.

When an access attempt is being made, a generalized resource rule will apply only if the environment described by the rule matches the actual access attempt. When this match occurs, then access is determined by the access permissions.

The syntax of an individual resource rule entry is as follows:

```
[UID(uid-mask)] [SOURCE(source-id)] -
[SHIFT(record-name)] [UNTIL(date)] [FOR(days)]-
[SERVICE(READ,UPDATE,ADD,DELETE] -
{ALLOW/LOG/PREVENT} [DATA(text)] [VERIFY]
```

Parameters Describing the Environment. The following parameters describe the environment that an access attempt must match in order for a rule to apply:

UID(uid-mask)

Specifies a UID string or UID-string-mask identifying the user(s) making the access attempt. If omitted, this parameter will include all users. Masking of the UID string is explained in the chapter introducing the Logonid record.

SOURCE(source-id)
Specifies the name of an input source or group of input sources through which the access (either logical or physical) attempt is being made. If omitted, this parameter will include all input sources.

SHIFT(record-name)
Specifies the 1- to 8-character name of the shift record defining the time, day, or date of the access attempt. If omitted, this parameter includes any time or day.

Do not enter more than one SHIFT parameter in a rule entry. To create new day/time combinations, insert a new shift record. This procedure is explained in the chapter on shift/zone records (SHIFT setting).

UNTIL(date)
Specifies that last date on which an access can take place under this rule. This Gregorian date can be in the format mm/dd/yy, yy/mm/dd, or dd/mm/yy (The DATE field of the OPTS record determines this date format, as discussed in the chapter on GSO records.)

FOR(days)
Specifies the number of days, from the compilation date of the generalized resource rule set, during which an access attempt can be made under this rule. The minimum number that can be specified is zero (meaning today only), and the maximum is 365.

SERVICE(READ,ADD,UPDATE,DELETE)
Specifies the type of file access associated with the access attempt. The SERVICE parameter is valid for PANEXEC/PANVALET, CICS files, user interfaces, and IDMS area resource rules. One or more file access keywords may be specified, and separated by blank characters or commas. If SERVICE is not specified, all types are the default. The possible keywords are:

* READ indicates read-only access.

* ADD indicates access for the purpose of adding new records to an existing CICS file. Not applicable to IDMS area resource rules.

* UPDATE indicates access for the purpose of modifying existing records.

* DELETE indicates access for the purpose of deleting records. Not applicable to IDMS area resource rules.

For further information on CICS, IDMS, or IMS, refer to the acf2/MVS CICS Support Manual, acf2/MVS IDMS Support Manual, or the acf2/MVS IMS Support Manual.

DATA(text)
Specifies up to 64 characters of data for the installation's own use. This data will be passed to the installation exits and back to the calling application subsystem along with the global USERDATA field. This data could indicate further checking or contain some control information.

VERIFY
Requests password validation for any access attempt made under this rule. The application subsystem requesting access to a resource will be informed of the request for password validation.

Parameters Specifying the Access Permission. A rule entry can contain one of the following parameters specifying the access permission. This access permission will apply only if the environment described by the rule matches the actual access attempt.

ALLOW
Specifies that access to the resource will be allowed.

LOG
Specifies that access to the resource will be allowed but logged. A System Management Facility (SMF) record will be written to log the event.

PREVENT
Specifies that access to the resource will be prevented. An SMF record will be written to log the event.

If none of these access permissions is specified, then PREVENT will be assumed.

Comment Cards

Comment cards are denoted by an asterisk (*) in column one and a blank in column two. They will not be lost on a compile/decompile sequence.

MASKING IN GENERALIZED RESOURGE RULE SETS

In generalized resource rule sets, the resource name ($KEY) and the UID parameter can be masked.

Masking the Resource Name. By masking the name for a generalized resource rule set, an installation can write rules that apply to a group of generalized resources rather than just one individual resource. The following sample $KEY control cards show some allowable masks for generalized resource rule set keys.

```
$KEY(G***)   $KEY(G**D)   $KEY(**DD)
```

A mask for a generalized resource ruleid follows the conventions of Logonid masking, except a resource-name mask cannot contain a dash (-). It can only contain asterisks.

A resource-name mask can contain a dash imbedded between other characters however, such a dash is treated literally as a character, not a mask. For more details on masking, see the explanation on Logonid masks in the chapter that introduces the Logonid record.

Masking is especially useful when an installation wants to create its own generalized resource types. The names of resources of a certain type can be standardized such that certain names will fit a particular mask, allowing access to those resources to be controlled by one rule set.

With masking, an installation can also write a generalized resource rule set for a group of resources, and still write a unique rule set for a single resource within that group. For example, a generalized resource rule set may apply to all resources of a given type code whose names are from 1 to 8 characters in length and begin with G:

```
$KEY(G*******)
```

Yet, another generalized resource rule set can apply especially for the transaction GETT:

```
$KEY(GETT)
```

When validating access to the resource, ACF2 will first evaluate the generalized resource rule set for the resource named GETT, since that name forms a more specific ruleid.

Masking the UID Parameter. A generalized resource rule can apply to multiple users if a UID string mask is specified in the UID parameter of the rule entry. Conventions for UID masking are explained in the chapter introducing the Logonid record.

CREATING GENERALIZED RESOURCE RULE SETS

You can create generalized resource rule sets directly from the terminal or by first building the rule set text in a file.

The general procedure is:

1. If necessary, build the generalized resource rule set text in a partitioned dataset(PDS). This step can be accomplished through editing.

2. From TSO READY mode, issue the ACF command.

3. Establish the RESOURCE setting with the appropriate type code (for example, SET RESOURCE(CKC)).

4. For direct compilation from the terminal, issue the COMPILE subcommand without a dataset name. The COMPILE subcommand will allow you to enter the control cards and rule entries at the terminal. For compilation from a PDS, issue the COMPILE subcommand with the name of the dataset that contains the rule set text.

5. To test the rule set, issue the TEST subcommand. The TEST subcommand can give you an idea of whether the rule set will do the proper validation of accesses to the resource.

6. To save the rule set on the database, issue the STORE subcommand. Otherwise, the rule set will eventually be lost.

7. If the rule set has been designated as being resident, issue the F ACF2,REBUILD(type-code) console operator command to rebuild the directory of resident generalized resource rules. For further information on resident rules, refer to the explanation of the RESDIR record in the chapter on GSO records. Also, refer to the chapter on console operator commands for an explanation of the REBUILD command.

ACF SUBCOMMANDS UNDER THE RESOURCE SETTING

You can process generalized resource rules after establishing the RESOURCE setting of the ACF command along with the appropriate type code. For example:

```
acf
set resource(ckc)
```

After establishing the RESOURCE setting and appropriate type code, you can issue any of the following ACF subcommands:

*COMPILE
*TEST
*STORE
*DECOMP
*LIST
*DELETE
END
HELP
SET
SHOW
SN

The common subcommands END, HELP, SET, SHOW, and SN operate under the RESOURCE setting as explained in the chapter on the ACF command. The following text describes the function, syntax, and parameters of the other subcommands, listed above with asterisks (*).

COMPILE Subcommand--RESOURCE Setting

The COMPILE subcommand allows you to create a set of generalized resource rules. The syntax of this subcommand is:

```
COMPILE [*/dsname] [LIST/NOLIST] [STORE/NOSTORE] [ALL]
```

ACF2 provides two ways of compiling generalized resource rule sets:

1. Directly from the terminal

2. From a partitioned dataset (PDS)

Compiling Directly from the Terminal. You can enter the text for a generalized resource rule set directly from the terminal by first entering the COMPILE subcommands without parameters:

```
acf
set resource(ckc)
compile
```

ACF2 will respond with:

```
ACF70010 ACF COMPILER ENTERED
```

Begin by entering the $KEY control card on the first line. Enter the other control cards and then the rule entries, each on a separate line. Press the ENTER or return key after each line.

```
acf
set resource(ckc)
compile
ACF70010 ACF COMPILER ENTERED

$key(acfm) type(CKC)
 uid(tfinpayiso) allow
 uid(tfinadmiso) until(11/24/85) allow
```

To end the rule set, enter a blank line. After compiling the generalized resource rule set, you can use the TEST subcommand to get an idea of whether the generalized resource rule set allows for proper access validation. To retain the generalized resource rule set, you must use the STORE subcommand.

Compiling from a Partitioned Dataset (PDS). You can create generalized resource rule sets by first entering the control cards and rule entries into a PDS member. Each control card or rule entry must be on a separate line. The last line does not have to be a blank line. For example:

```
$KEY(ACFM) TYPE(CKC)
 UID(TFINPAYISO) ALLOW
 UID(TFINADMISO) UNTIL(11/24/85) ALLOW
```

After entering the rule set text into the PDS member, you can invoke ACF2 and compile the generalized resource rule set. Issue the COMPILE subcommand with the PDS name and member name. For example:

```
acf
set resource(ckc)
compile work.text(rule)
```

In the COMPILE subcommand, the PDS name is specified according to TSO conventions. (Your own high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYNLT.WORK.TEXT(RULE)' or simply WORK.TEXT(RULE).

Parameters. Under the RESOURCE setting, the COMPILE subcommand takes the following parameters:

*

An asterisk indicates that the subsequent text entered at the terminal will be input to the compiler. In a TSO environment, the system will prompt you to enter the generalized resource rule text directly from the terminal. In batch, the cards following the card that contains the COMPILE subcommand will be assumed as input.

(no parameters)

Use of the COMPILE subcommand without parameters is equivalent to specifying an asterisk.

dsname

A partitioned dataset (PDS) and member name specifies the PDS and member that contains the generalized resource rule text to be compiled. The PDS name follows TSO conventions. Your own high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYNLT.WORK.TEXT(RULE)'.

If you do not specify a member name, ACF2 will prompt you for one. To compile input from all PDS members, specify the ALL parameter. A generalized resource rule set cannot be compiled from a sequential dataset.

LIST/NOLIST

The LIST parameter causes the input to the compiler to be displayed on your screen or printed on your listing during compilation of a rule set. NOLIST causes no such display or printed list. LIST is the default.

STORE/NOSTORE

The STORE parameter causes the rule set to be automatically stored at compilation time. NOSTORE causes no automatic storing of the rule set; you must issue the STORE subcommand to store the rule set. STORE is the default if you are using the ALL parameter to compile all members of a PDS. Otherwise NOSTORE is the default.

ALL

The ALL parameter causes compilation and storing of generalized resource rule sets from all members of a specified partitioned dataset (PDS). For example:

```
acf
set resource(ckc)
compile work.text all
```

If any members of a PDS do not contain generalized resource rule set text, then do not specify the ALL parameter.

TEST Subcommand--RESOURCE Setting

The TEST Subcommand allows you to interactively test a compiled generalized resource rule set. This testing can give you an idea of whether the generalized resource rule set provides for the proper validation of access to a resource.

The Extent of Generalized Resource Rule Testing. While the TEST subcommand is active, only generalized resource rule interpretation is done. This testing does not take into account any installation-specific system options or attributes of the Logonids being tested. Nor does this testing take into account any installation-developed exits. The test subcommand is simplified because of the impossibility of testing all possible exit combinations.

Syntax. The syntax of the TEST subcommand is:

```
TEST [*/ruleid]
```

Example. Suppose a generalized resource rule set with the key ACFM were just compiled:

```
ACF70010 ACF COMPILER ENTERED

$key(acfm) type(ckc)
 uid(tfinpayiso) allow
 uid(tfinadmiso) until(11/24/85) allow

ACF70051 TOTAL RECORD LENGTH= 162 BYTES, 3 PERCENT UTILIZED
```

The TEST subcommand could then be issued:

```
test *
```

When the period (.) is displayed, the TEST subcommand will be active. You can enter any of the TEST subcommand keywords to specify the particular environment that you want to test. For example, the following keywords test whether the generalized resource rule set shown above will allow the user TFINPAYISO to have read, add, and update access to the ACF2/CICS ACFM transaction:

```
test
.

uid(tfinpayiso)
```

After the TEST subcommand keywords are entered, the system responds by displaying all of the current values that describe the environment being tested. At the bottom of the display, is an indication of whether the access will be allowed, logged, or prevented:

```
test
.

uid(tfinpayiso)
ACF71114 THE FOLLOWING PARAMETERS ARE IN EFFECT:
 DATE=11/27/84 TIME=1501 SOURCE=******** UID=TFINPAYISO

 ACCESS WOULD BE ALLOWED
.
```

In the above example, the result is that this user would be allowed to read, update, or add records under the ACFM transaction.

After a result is displayed, you can make another entry of keywords and values to specify another environment for testing. The END subcommand will terminate the TEST subcommand.

How the Values Describing the Environment Work. After you enter values describing the test environment, these values remain in effect until they are specifically changed.

Any values that have not been specified may appear as asterisks (********). The asterisks can match any value. For instance, if you specify no UID keyword, then the subcommand will test whether all UIDs are allowed access.

Parameters. The TEST subcommand can be issued with any of the following parameters:

*
   An asterisk indicates that the previously compiled generalized resource rule set should be tested.

(no parameter)
   When specified without a parameter, the TEST subcommand operates the same as when an asterisk is specified.

ruleid
The ruleid identifies a specific generalized resource rule set to be tested.

TEST Subcommand Keywords. After you have issued the TEST subcommand along with any of the parameters described above, the TEST subcommand will become active. You can specify a test access environment by entering any of the following keywords with the appropriate values:

LID(Logonid)
Specifies a Logonid identifying a single user whose access is to be tested. To specify this keyword, the user making the request needs the appropriate access to the Logonid record of the user whose access is being tested.

UID(uid-mask)
Specifies a mask of the UID strings identifying the users whose access will be tested. To specify this keyword, the user making the test does not need access to the Logonid records of the users whose access is being tested.

DATE(date)
Specifies the date for which access will be tested. This date can be in the format mm/dd/yy, yy/mm/dd, or dd/mm/yy. The appropriate format is specified in the DATE field of the OPTS record. See the chapter on GSO records. Initially, the TEST subcommand uses the current date as the default.

SOURCE(source-id)
Specifies the physical or the logical name of the input source or source group for which access will be tested. Initially, the TEST subcommand uses all input sources as the default. Input sources are discussed further in the chapter on entry records.

TIME(hhmm)
Specifies the time (in hours and minutes) for which access will be tested. Initially, the TEST subcommand uses the current time as the default.

SERVICE(READ,UPDATE,ADD,DELETE)
Specifies the type of file access that is being tested. This access type can be READ, UPDATE, ADD, or DELETE. Multiple access types must be separated by blank characters or commas. This keyword applies only to generalized resource rules for CICS file and IDMS data area access. Omission of SERVICE defaults to ALL services.

TEST Subcommand Results. The results of the TEST subcommand show that access to the resource under the specified service would be one of the following:

ALLOWED	Access would be allowed
LOGGED	Access would be allowed but logged
PREVENTED	Access would be specifically prevented

If no rule entry specifically applies to the test access environment, the following message is displayed:

```
NO RULES APPLY, ACCESS WOULD BE DENIED.
```

STORE Subcommand--RESOURCE Setting

The STORE subcommand, under the RESOURCE setting, allows you to store the previously compiled set of generalized resource rules.

Syntax. The STORE subcommand has the following syntax:

```
STORE
```

Parameters. This subcommand accepts no parameters.

If SET NOFORCE has been previously issued, then the generalized resource rule set is stored only if it does not already exist. You will receive a message if an generalized resource rule is not stored. The NOFORCE parameter of the SET subcommand is explained in the chapter on the ACF command.

The store operation may be rejected because a user may have insufficient authority for storing generalized resource rule sets.

DECOMP Subcommand--RESOURCE Setting

The DECOMP subcommand, under the RESOURCE setting, decompiles a generalized resource rule set that has been previously compiled and stored. This subcommand is useful for examining generalized resource rule sets. A generalized resource rule set can be decompiled at the terminal or into a member of a partitioned dataset (PDS). The LIST subcommand, under the RESOURCE setting, follows the same syntax and accomplishes the same function as the DECOMP subcommand.

Syntax. The DECOMP subcommand has the following syntax:

```
DECOMP */ruleid/LIKE(ruleid-mask) [INTO(dsname)]
```

Parameters. The DECOMP subcommand takes the following parameters:

*

An asterisk allows you to decompile the last individual generalized resource rule set that you processed since establishing the current RESOURCE setting and type code.

ruleid
The ruleid specifies an individual generalized resource rule set to be decompiled or listed.

LIKE(ruleid-mask)
The LIKE parameter specifies a mask of ruleids for decompiling or listing a group of generalized resource rule sets.

INTO(dsname)
The INTO parameter specifies the PDS into which the rule set will be decompiled. If specifying more than one rule set, you cannot specify a member name. If specifying a fully-qualified dataset name (i.e., including the high-level index), you must enclose that name within single quotes. For example:

```
DECOMP ACFM INTO('PAYISO.WORK.RULE')
```

If you specify no member name, the ruleid is used. However, if the member name is invalid, then a member name is automatically generated. For further information on the automatic generation of this member name, see the explanation of the SET MEMBER subcommand format in the chapter on the ACF command.

DELETE Subcommand--RESOURCE Setting

The DELETE subcommand, under the RESOURCE setting, allows for deletion of generalized resource rule sets.

Only users with the SECURITY privilege level can delete generalized resource rule sets. This authority can be restricted through the use of scopes.

Example. For example, the following subcommand deletes the generalized resource rule set for the ACF2/CICS ACFM transaction:

```
delete acfm
```

After the DELETE command is issued, the message DELETED is returned.

Syntax. The DELETE subcommand has the following syntax:

```
DELETE {*/ruleid/LIKE(ruleid-mask)}
```

Parameters. The DELETE subcommand, under the RESOURCE setting, takes the following parameters:

*
An asterisk specifies the deletion of the last generalized resource rule set that you referenced in your current TSO session.

ruleid

Identifies one specific generalized resource rule set for deletion.

LIKE(ruleid-mask)
Specifies a mask of resource names for which rule sets will be deleted.

RULE SELECTION ALGORITHM

The rule compiler converts the rule set input into a form usable by the rule interpreter during verification checking. The compiler also orders the rules according to the following criteria:

* UID patterns, from most specific to most general.

* SOURCE parameters in alpha-order with "not specified" last.

* SHIFT parameters in alpha-order with "not specified" last.

* UNTIL dates, from earliest to latest.

The first active rule whose environment matches the actual access attempt will be used to determine whether access will be granted.

ENTRY SETTING: ENTRY RECORDS

Entry records can be used in several ways to define input sources:

- Identify individual input sources, such as terminals or card readers, for the purpose of ACF2 validation. This type of record is used to translate physical input sources to logical input sources. ACF2 recognizes both.
- Identify groups of input sources for the purpose of ACF2 validation. These groups can be made up of logical or physical input source names.

This chapter discusses:

1. Examples of entry records
2. Types and names of entry records
3. Fields of each type of entry record
4. Creating each type of entry record
5. ACF subcommands under the ENTRY setting

EXAMPLES OF ENTRY RECORDS

This section gives brief examples of the various types of entry records.

Example of an Input Source Record

When listed, an input source record might look like:

```
TYPE: SRC   ENTRY: LV133    1 DATA ITEMS
 ROOM110
```

In this example, the type code SRC identifies this record as an input source record. LV133 is the name of the entry record, and is also the physical name that identifies a terminal. The data item, ROOM110, defines a logical name for the terminal. The installation has chosen this logical name because it is more meaningful than the name LV133.

Example of a Source Group Record

When listed, a source group record might look like:

```
TYPE: SGP   ENTRY: PAYROLL    3 DATA ITEMS
 ROOM110
 ROOM120
 LV156
```

In this example, the type code SGP identifies this record as a source group record. PAYROLL is the name of the entry record, and is a logical name that identifies all terminals in the payroll department. This record contains three data items, which identify the individual terminals or groups making up the source group.

In a source group record, individual terminals can be identified by either their logical names or physical names. A source group record can contain a maximum of 255 entries. Also, an individual input source can simultaneously be defined in more than one source group.

TYPES AND NAMES OF ENTRY RECORDS

As shown by the examples, entry records are identified by type, and then by name. Entry records can be one of several types, with each type identified by a type code of 3 alphanumeric characters:

SRC	For input source records
SGP	For source group records

Each record of a given type is further identified by a name. For input source and source group records, this name can be from 1 to 8 characters. For OID records, this name must match the length of Logonids at the installation.

FIELDS IN EACH TYPE OF ENTRY RECORD

The fields of an entry record are referred to as data items. For each type of entry record, the following table summarizes the type code and what information is allowable for the record name and data items:

TYPE CODE	RECORD NAME	DATA ITEM
SRC	Physical input source name	Logical input source name to represent the physical input source name (one data item only)
SGP	Source group name	Physical or logical names of input sources in the group. Or names of source groups contained in the group.

CREATING ENTRY RECORDS

This section gives examples of how to create each type of entry record. Only users with the SECURITY privilege level can create entry records.

Creating Input Source or Source Group Records

To create an input source or source group record, first issue the ACF command, and then establish the ENTRY(SRC) or ENTRY(SGP) setting. Issue the INSERT subcommand along with the record name and a data item:

```
acf
set entry(src)
insert lv433 newdata(room100)
```

In this example, LV433 (the record name) is the physical name of the input source. ROOM100 (the data item) is the logical name that the installation has chosen to identify the input source.

To make a newly inserted source group record active, an F ACF2,NEWXREF command can be entered at the system operator console. See the chapter "Console Operator Commands" for information on this command.

ACF SUBCOMMANDS UNDER ENTRY MODE

You can process entry records after you have issued the ACF command and have established the appropriate setting, as follows:

SET ENTRY(SRC) For input source records

SET ENTRY(SGP) For source group records

After establishing one of the ENTRY settings, you can issue any of the following ACF subcommands:

*INSERT
*CHANGE
*LIST
*DELETE
END
HELP
SET
SHOW
SN

The common subcommands END, HELP, SET, SHOW, and SN operate under the ENTRY settings as previously explained in the chapter on the ACF command. The following sections describe in detail each of the other ACF2 subcommands marked with asterisks (*).

INSERT Subcommand--ENTRY(SRC) and ENTRY(SGP) Settings

The INSERT subcommand, under the ENTRY(SRC) and ENTRY(SGP) settings, allows for insertion of an entry record onto the database.

Syntax. The INSERT subcommand has the following syntax:

```
INSERT [USING(model-record-name) [TYPE(model-type)]] -
       record-name [NEWDATA(data-item)] [DSN(dsn)] [CLEAR]
```

Example. In its simplest form, the INSERT subcommand allows for insertion of an entry record with a record-name and one data item:

```
insert lv433 newdata(room100)
```

In this example, the physical input source name LV433, a terminal, is the record name. The logical input source name ROOM100 is the data item to be contained in the record.

Parameters. The INSERT subcommand has the following parameters:

USING(record-name)
Allows you to insert an entry record by copying the data items and any pseudo dataset name from an existing, model entry record. (The pseudo dataset name is described below under the explanation of the DSN parameter.) The following example illustrates the USING parameter:

```
insert using(payroll) type(sgp) finance newdata(room200)
```

Using the source group record PAYROLL as a model, this example subcommand inserts a record for the source group FINANCE. Any data items and any pseudo dataset name in PAYROLL is copied in FINANCE. The logical input source name ROOM200 is also added to the source group FINANCE.

TYPE(xxx)
(Optional) Specifies the type code of the model entry record. This type code is either SRC or SGP.

CLEAR
(Specified only with the USING parameter) Allows you to copy the model entry record, clear the pseudo dataset name, and then add new data items, all with one subcommand.

DSN(pseudo-dsn)
Allows control over which users can access the entry record for the purpose of listing, changing, or deleting it. This parameter allows you to specify a "pseudo dataset name," which can be the name of an actual or nonexistent dataset. This dataset name must be fully qualified and entered without single quotes. Any users with access to the pseudo dataset, via access rules, also have access to the entry record. Without the use of the DSN parameter, a user's access

to the entry record is limited by other user attributes, such as special privileges and scopes.

Making the New Record Active. After inserting or changing a source group (SGP) record, the new values are not active until the next system startup (IPL) or until an F ACF2,NEWXREF operator command is issued. The NEWXREF command will dynamically update the system's active input source cross-reference table. See the chapter on console operator commands for information on this command.

CHANGE Subcommand--ENTRY(SRC) and ENTRY(SGP) Settings

The CHANGE subcommand, under the ENTRY(SRC) or ENTRY(SGP) setting, provides several methods of adding, replacing, and removing data items in entry records. Since input sources (SRC) use only the first data item, the CHANGE should not be used for them.

Syntax. The syntax of this subcommand is:

```
CHANGE {*/record-name/LIKE(record-name-mask)} -
       [OLDDATA(data-item)] [NEWDATA(data-item)] -
       [CLEAR] [VERDATA(data-item] [DSN(dsn)]
```

Parameters. The CHANGE subcommand takes the following parameters:

*
Specifies the last record of this type processed since establishing the ENTRY setting with a specific type code (e.g.; SET ENTRY(SGP)).

record-name
Specifies the name of one particular entry record.

LIKE(record-name-mask)
Specifies a record-name mask, which allows a group of entry records with similar record-names to be changed by one CHANGE subcommand.

NEWDATA(data-item), OLDDATA(data-item), and VERDATA(data-item)
Determine the effect of the CHANGE subcommand, as follows. (Note that at least one of the parameters NEWDATA, OLDDATA, or CLEAR must be specified.)

1. To add a new data item to an entry record, use the NEWDATA parameter:

   ```
   CHANGE record-name NEWDATA(data-item)

   Example:  change pay03 newdata(room100)
   ```

   The above subcommand adds another input source name ROOM100 to the source group PAY03.

2. To remove a data item from an entry record, use the OLDDATA parameter:

   ```
   CHANGE record-name OLDDATA(data-item)

   Example:  change pay03 olddata(room100)
   ```

   The above subcommand removes the input source name ROOM100 from the source group PAY03.

3. To replace a data item with a new one, use the OLDDATA and NEWDATA parameters:

```
CHANGE record-name OLDDATA(data-item) NEWDATA(data-item)

Example:  change pay03 olddata(room100) newdata(room110)
```

For the source group PAY03, the above subcommand replaces the input source name ROOM100 with the input source name ROOM110.

4. To remove a data item from an entry record if that record contains the data item specified by the VERDATA parameter, use the syntax:

```
CHANGE record-name VERDATA(data-item) OLDDATA(data-item)

Example:  change pay03 verdata(room100) olddata(room110)
```

The above subcommand removes the input source name ROOM110 if the input source name ROOM100 currently exists in the source group PAY03. The names specified in the VERDATA and OLDDATA parameters can be the same.

5. To add a data item to an entry record if that record contains the data item specified by the VERDATA parameter, use the syntax:

```
CHANGE record-name VERDATA(data-item) NEWDATA(data-item)

Example:  change pay03 verdata(room100) newdata(room110)
```

The above subcommand adds the input source name ROOM110 if the input source name ROOM100 currently exists in the source group PAY03. The names specified in the NEWDATA and VERDATA parameters can be the same.

6. To replace a particular data item in an entry record if that record contains the data item specified by the VERDATA parameter, use the syntax:

```
CHANGE record-name VERDATA(data-item) NEWDATA(data-item) -
       OLDDATA(data-item)

Example:  change pay03 verdata(room100) newdata(room112) -
            olddata(room110)
```

The above subcommand replaces the input source name ROOM110 with the input source name ROOM112 if the source group PAY03 currently contains the input source name ROOM100. The names specified in the VERDATA parameter can be the same as the one specified in the OLDDATA or NEWDATA parameter.

LIKE(record-name-mask)

(Usually combined with the VERDATA parameter) Allows you to add a data item to more than one entry record at a time:

```
CHANGE LIKE(record-name-mask) NEWDATA(data-item) -
    VERDATA(data-item)
```

The new data item is added to each entry record that has a record-name matching the specified mask, and that contains the data item specified by the VERDATA parameter.

The OLDDATA and VERDATA parameters can also be used in the same manner for removal of a data item from more than one record at a time:

```
CHANGE LIKE(record-name-mask) OLDDATA(data-item) -
    VERDATA(data-item)
```

An existing (old) data item can be removed from each entry record that has a record-name matching the specified mask and that contains the data item specified by the VERDATA parameter.

DSN(pseudo-dsn)

Allows you to change the pseudo dataset name that controls which users have access to the entry record. If this parameter specifies a pseudo dataset name where none existed previously, the name is added to the entry record. If this parameter specifies a name where one currently exists, the new name replaces the old name. If this parameter is specified as DSN(), the existing name is removed. This dataset name must be fully qualified (the full name included) and specified without single quotes.

CLEAR

Clears an existing entry record of all data items and any pseudo dataset name.

Making a Changed Record Active. After you have changed a source group record, an F ACF2,NEWXREF console operator command should be issued. This command will update the input source cross-reference table. See the chapter "Console Operator Commands" for information on this command.

LIST Subcommand--ENTRY Setting

The LIST Subcommand under the ENTRY setting lists the type, record-name and number of data items for the specified entry record(s).

Syntax. The LIST subcommand has the following syntax:

```
    LIST {*/record-name/LIKE(record-name-mask)}
```

Parameters. The LIST subcommand takes the following parameters:

*
Specifies, by record-name, listing of the last entry record processed since establishing the current ENTRY setting.

record-name
Specifies the record-name of one particular entry record to be listed.

LIKE(record-name-mask)
Specifies a record-name mask, which allows a group of entry records with similar record-names to be listed by one LIST subcommand.

DELETE Subcommand--ENTRY Setting

The DELETE Subcommand deletes an entry record from the database (including record-name, data items, and any pseudo dataset name).

Syntax. The DELETE subcommand has the following syntax:

```
DELETE {*/record-name/LIKE(record-name-mask)}
```

Parameters. The DELETE subcommand takes the following parameters:

*
Specifies, by record-name, processing of the last entry record processed since estalishing the current ENTRY setting.

record-name
Specifies the record-name of one particular entry record.

LIKE(record-name-mask)
Specifies a record-name mask, which allows a group of entry records with similar record-names to be changed by one CHANGE subcommand.

Making Deleted Records Inactive. Note that after you have deleted a source group record, the system operator should issue an F ACF2,NEWXREF command. This command will update the input source cross-reference table. See the chapter "Console Operator Commands" for information on this command.

SCOPE SETTING: SCOPE RECORDS

A scope record specifies a list of dataset high-level indices, Logonids, UID strings and/or Infostorage keys to be the authorized control limits or scope of a privileged user. Use of the scope record limits the special privileges of a user as they apply to access to the ACF2 databases. Scopes themselves grant no special privileges to a Logonid. They merely provide one method where the various functions of security administration can be delegated to other Logonids while limiting the full power of those Logonids.

The following topics will be discussed in more detail below:

- * Examples
- * Names
- * Record fields
- * Creating scope records
- * Scope-related changes to the Logonid record for future acf2/MVS releases
- * ACF subcommands for scope record processing under the SCOPE setting

SCOPE RECORD EXAMPLE

A scope record may look like:

```
ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYWORK,PAYTEST) LID(PAY-) UID(FIN-) INF(RCKCPAYT,RIAGPAYUPD)
```

The scope list (PAYSCOPE) when associated with a SECURITY Logonid grants that user authority to:

1. Write access rules governing the two groups of Payroll work files (the high-level indices PAYWORK and PAYTEST).

2. Write generalized resource rules for CICS and IMS Payroll transactions (PAYT on CICS and PAYUPD on IMS).

3. Update those Logonid records which match both the Logonid mask PAY- (e.g., all Payroll employees) and the UID mask beginning with FIN. The LID and UID entries must both be specified. A Logonid record being processed must match both the LID and UID entries specified to authorize access. If either entry is not specified, the default is no records can be updated (since none will match and the default is no access).

This scope record would not apply to a given user until its record name (PAYSCOPE) is specified in the SCPLIST field of that user's Logonid record.

NAMES OF SCOPE RECORDS

Only one type of scope record exists on the Infostorage Database. Scope records are identified by the type code SCP and a specific name from 1 to 8 characters in length. The scope list record is kept on the Information Storage database rather than the user's Logonid record. The SCPLIST field of the Logonid record contains only the name of the applicable scope record for that user. It acts as a pointer to the Infostorage Database record which contains the list of control limits for that privileged user.

FIELDS OF A SCOPE RECORD

Scope records restrict a user's access to the ACF2 database which a special privilege (e.g., SECURITY, ACCOUNT) would normally allow. Therefore, scopes contain four fields which act as the parameters that define a privileged user's access to the ACF2 database involved. If one of the fields contains no entries, the Logonid associated with that scope will be completely restricted from using that privilege relative to that database. Each of the fields listed can contain a single value or a list of values.

* The DSN parameter can contain multiple dataset high level index names or masks. Each entry contains one to eight characters representing the high-level index ($KEY) of the dataset(s) you wish to include within the scope of a user. These entries grant rule writing privileges to scoped users and can be masked. If none of the fields contain an entry, the Logonid associated with that scope will be completely restricted from rule access. Logonids with a privilege included in the DECOMP field of the GSO OPTS record can decompile generalized resource or access rules but not create or store the rules. This privilege overrides the restrictions or limits implied by a scope list.

* The INF field can contain one or more one to forty-four character entries indicating which matching Infostorage database records are within the scope of the user to INSERT, ALTER or DELETE. The authority to list these fields is granted to all users with similar privileges as specified on the INFOLST field of the GSO OPTS record regardless of scope records. Masks are allowed in this entry.

```
Format:  ctttk1,...,ctttkn

c      = storage class (e.g., R for Resource Rule)
ttt    = type code (e.g., CKC for CICS transaction rules)
k1-k40 = name (e.g.,  PAYT for  the name  of a  specific CICS
         transaction)
```

- The LID field contains Logonids or Logonid masks to be placed within the scope of the user. A UID entry must also be specified to allow access to records on the Logonid Database to occur.

- The UID field contains the UID string or UID string mask (extending from one to twenty-four characters) to be placed within the scope of the user. If used, the LID field is also required.

The true effect of a scope relates directly to the authority privileges of the Logonid. Multiple entries or fields can exist on a scope record.

The following chart describes each field on the scope record and its effect on the specific privilege type on a Logonid. Notice there is a distinction made between the ability to update or create records. This authority is granted by the special privilege, not the scope. See the section in this manual which describes the privileges in the Logonid.

Authority Level	LID	UID	DSN	INF
SECURITY*	Update matching Logonids	Update matching Logonids	Compile/store test/delete matching Rule sets	Create/update matching Infostorage Records
ACCOUNT**	Create matching Logonids	Create matching Logonids	Create/update delete Logonids with a matching PREFIX field	None
AUDIT	Display matching Logonids	Display matching Logonids	Decompile matching Rule sets	Display matching Infostorage records
LEADER	Update selected fields in matching Logonids	Update selected fields in matching Logonids	None	None
CONSULT	Update selected fields in matching Logonids	Update selected fields in matching Logonids	None	None
USER	None	None	None	None

* Certain fields of the Logonid record no longer can be changed once a user with the SECURITY privilege level becomes restricted. These fields are indicated by the RESTRICT flag in the @CFDE macro. For example, a scoped SECURITY Logonid can no longer update the SCPLIST fields of Logonid records. However, he can create scope records if that is within his INF scope. (See the explanation of the ACF2 Field Definition Record (ACFFDR) in the acf2/MVS System Programmer's Guide.)

** A scoped ACCOUNT Logonid cannot use the ACF2 SYNCH command.

Notes:
The LID and UID fields interact together. If you scope an ACCOUNT Logonid, both the LID and UID fields require an entry. This way the ACCOUNT Logonid can create other Logonids with the UID named in the scope. These two fields are required when scoping any privileged Logonid to allow access to other Logonid records.

If the ACCOUNT Logonid is going to assign different PREFIX values to Logonids, the values must be within the list or masks specified in the DSN field.

The absense of an entry in any one of these fields defaults to no access. For example, a SECURITY Logonid may have a SCPLST with INF entries for entry records:

```
     SCOPE LIMIT1 INF(ESRC-,ESGP-)
```

Since no entries were made in the LID, UID, or DSN fields, the SECURITY Logonid will not have the authority to update Logonids or access rule sets. (Default equals no access.) The authority of his SECURITY privilege as it relates to the ACF2 databases will be to create and update the entry records in Infostorage.

The INFOLIST field of the GSO OPTS record allows users with the specified privilege level(s) to read records on the ACF2 Infostorage database. This authority applies to entry records, scope records, shift/zone records, and control records. It does not apply to generalized resource rules. The DECOMP field on the GSO OPTS record allows users with the specified privilege level to decompile access and generalized resource rules. Neither field allows the related Logonids to create or update any of the records. These two parameters (INFOLIST and DECOMP) are not effected by scopes.

CREATING SCOPE RECORDS

To process scope records with ACF subcommands, establish the SCOPE setting by use of the SET subcommand:

```
     acf
     set scope(scp)
```

The scope-list-name may extend from one to eight characters and will be defined in the SCPLIST field of the Logonid record pointing to the associated name on the Infostorage database. Once in the SCOPE setting, you can insert a scope record by using a subcommand such as:

```
     insert limit1 dsn(payroll) lid(pay-) uid(1h*pr-)
```

This example illustrates the entries for a scope named LIMIT1. When applied to a SECURITY Logonid, that Logonid will only have authority to access the access rule set PAYROLL, and update Logonids beginning with

PAY that also match the UID entry. This scoped Logonid will not be able to access any other Logonids, any other access rule sets, or any Infostorage records.

Using an Existing Scope Record to Insert a New Record

The INSERT USING subcommand may be used to specify an existing scope list record as a model to create a new scope list record. The ADD, REPLACE, or DELETE subfunctions may be used to modify the new record from the model record. If omitted, the ADD subfunction is assumed. One of the four different types of entries must also be specified.

```
insert using(limit1) limit2 add lid(act-)
```

This example subcommand uses the scope record LIMIT1 to create a new record named LIMIT2. LIMIT2 will contain the same fields as does LIMIT1. In addition, LIMIT2 will contain a LID field with the values ACT- and PAY-.

SCOPE-RELATED CHANGES TO THE LOGONID RECORD FOR FUTURE ACF2/MVS RELEASES

The DSNSCOPE, LIDSCOPE, and UIDSCOPE fields of a user's Logonid record can achieve similar effects with a scope list record.

However, the DSNSCOPE, LIDSCOPE, and UIDSCOPE fields will probably be dropped in a future release of acf2/MVS. New installations should only use the SCPLIST field and corresponding scope list records.

Migration to using SCPLIST. If your installation is already using the DSNSCOPE, LIDSCOPE, and UIDSCOPE fields, you should begin converting to the use of the SCPLIST field and scope list records as soon as possible. During this conversion, the SCPLIST field will override the effects of the DSNSCOPE, LIDSCOPE, and UIDSCOPE fields. Thus, your installation can make this conversion without the user noticing any difference in ACF2 validation.

ACF SUBCOMMANDS UNDER THE SCOPE SETTING

Scope records can be processed after establishing the SCOPE setting of the ACF command:

```
set scope(scp)
```

After establishing the SCOPE setting, you can issue any of the following ACF subcommands:

*INSERT
*CHANGE
*LIST
*DELETE
END
HELP
SET
SHOW
SN

The common subcommands END, HELP, SET, SHOW, and SN operate under the SCOPE setting as explained in the cha.)ter on the ACF command. The following text describes the function, syntax, and parameters of the other subcommands, marked with asterisks (*).

INSERT Subcommand--SCOPE Setting

The INSERT subcommand, under the SCOPE setting, allows you to insert new scope records.

Syntax. The INSERT subcommand can have one of two basic syntax formats:

```
INSERT {record-name}   {[DSN(index,index...)]
                        [INF(storage-key,storage-key...)]
                        [LID(logonid-mask,logonid-mask...)]
or                      [UID(uid-mask,uid-mask...)]}

INSERT USING(model-record-name) {record-name} [ADD/REP/DEL]
                       {[DSN(index,index...)]
                        [INF(storage-key,storage-key...)]
                        [LID(logonid-mask,logonid-mask...)]
                        [UID(uid-mask,uid-mask...)]}
```

Example. In its simplest form, this subcommand requires a scope record name of 1 to 8 characters and one parameter such as INF. For example:

```
insert limit3 inf(esrc-,esgp-)
```

A scope record named LIMIT3 is created. It contains INF entries. When associated with a SECURITY Logonid, this scope record restricts the user's access to records on the Infostorage Database to only entry records. By default, complete restriction is placed on access to Logonids and access rules. If the scope contains DSN, LID, or UID parameters, both the LID and UID entries are required to allow the appropriate access.

Parameters. If multiple values are specified for the DSN, INF, LID, and UID parameters, these values must be separated by commas or blank spaces. The INSERT subcommand takes the following parameters:

USING(model-record-name)
The USING parameter specifies the name of an existing "model" scope record. The fields will be copied from that model record to create the new scope record. The ADD, REPLACE, and/or DELETE parameters alter the fields that will be in the new record. If you omit these parameters when specifying the USING parameter, then the ADD parameter is assumed and any values specified will be added to those already defined in the model record.

record-name

This is the name of the new scope record being inserted. The entry is from 1 to 8 alphanumeric characters. This parameter is required.

ADD

The ADD parameter indicates that any values specified by the DSN, INF, LID, and UID parameters of the INSERT subcommand will be added as a new field, or added to any corresponding fields copied from the model record. For example, take the following two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(act-) uid(1h*pr-)
insert using(scope1) scope2 add dsn(account)

list scope2
ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYROLL,ACCOUNT)
```

The first subcommand inserts a scope record named SCOPE1. The second subcommand uses SCOPE1 to insert a scope record named SCOPE2. As listed, SCOPE2 contains a DSN field with the value PAYROLL (as does SCOPE1) and also the value ACCOUNT. The ADD parameter is meaningful only when you specify the USING parameter. Also, the ADD parameter is assumed if you omit the ADD, REP, and DEL parameters when specifying the USING parameter,

REP

The REP parameter indicates that any values specified by the DSN, INF, LID, and UID parameters of the INSERT subcommand will replace any corresponding field copied from the model record. For example, take the following two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(pay-)
insert using(scope1) scope2 rep lid(act-)

list scope2
ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

The first subcommand inserts a scope record named SCOPE1. The second INSERT subcommand uses SCOPE1 to insert a scope record named SCOPE2. As listed, SCOPE2 contains a DSN field with the value PAYROLL (as does SCOPE1). SCOPE2 also contains a LID field with the value ACT- (replacing the value PAY-, which was copied from SCOPE1). The REP parameter is meaningful only when you specify the USING parameter.

DEL

The DEL parameter indicates that any values specified by the DSN, INF, LID, and UID parameters of the INSERT subcommand will be deleted from the corresponding fields copied from the model record. For example, take the following two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(pay-)
insert using(scope1) scope2 del lid(pay-)

list scope2
ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYROLL)
```

The first subcommand inserts a scope record named SCOPE1. The second INSERT subcommand uses SCOPE1 to insert a scope record named SCOPE2. As listed, SCOPE2 contains a DSN field with the value PAYROLL (as does SCOPE1), but the LID field that was copied has been deleted. The DEL parameter is meaningful only when you specify the USING parameter.

DSN(index,index-mask...)

The DSN parameter restricts the user's authority to access rule sets. These specified rule sets are identified by high-level indexes or a mask of high-level indexes. A mask can be 1 to 8 alphanumeric characters. Multiple masks or high-level indexes must be separated by commas. If this parameter is not specified, then dataset access is completely restricted except for access granted through access rule sets or special privileges. Scoped users with similar authority as listed on the DECOMP field of the GSO OPTS record will still be able to decompile access and generalized resource rules regardless of the limits imposed on their scope.

INF(storage-key,storage-key-mask...)

The INF parameter restricts Infostorage database access so that it includes access to only the specified generalized resource rule sets, entry records, scope records, shift/zone records, and control records. However, access to entry lists can also be granted by a pseudo dataset name. This is explained in the chapter in this manual about Delegating Authority. Resource names and Infostorage records are identified by a storage key of 1 to 44 characters. Scoped users with similar authority as listed on the INFOLIST field of the GSO OPTS record will still be able to list Information Storage records regardless of the limits imposed using this field on their scope. This storage key consists of:

1. A letter code indicating the storage class, as follows:

R	Generalized resource rule sets
E	Entry records
S	Scope records
T	Shift/zone records
C	Control (GSO) records

2. The three-character type code that identifies the type of record or generalized resource rule set. (This type code corresponds with the one used in establishing the RESOURCE, ENTRY, SCOPE, SHIFT, or CONTROL setting.)

3. Either the key of the generalized resource rule set or the name of the Infostorage database record.

For example, the storage key RCKCACFM consists of:

R	The storage class for a generalized resource rule set.
CKC	The type code for CICS transactions.
ACFM	The name of the transaction.

Multiple storage keys and storage key masks must be separated by commas or blank spaces. If this parameter is not specified and the DSN, LID, or UID parameter is, then Infostorage database access is completely restricted. However, access may be granted through generalized resource rule sets, special privileges, or pseudo dataset names.)

LID(logonid-mask,logonid-mask...)
The LID parameter restricts Logonid record access so that it includes access to only the specified Logonid records. These specified records are identified by Logonids or Logonid masks of 1 to 8 characters. Multiple Logonids or masks must be separated by commas or blank spaces. If this or the UID parameter is not specified, then Logonid record access is completely restricted. (However, access may be granted through special privileges.)

UID(uid-mask,uid-mask...)
The UID parameter restricts Logonid record access so that it includes access to only the specified Logonid records. These records are identified by UID strings or UID string masks of 1 to 24 characters. Unlike other UID parameters for ACF2, this UID parameter requires that any UID mask end with a dash (-) when padding is desired. Blank characters are not automatically inserted to complete the 24-character string. Multiple UID strings or masks must be separated by commas or blank spaces. If this or the LID parameter is not specified, then Logonid record access is completely restricted. (However, access may be granted through special privileges.)

CHANGE Subcommand--SCOPE Setting

The CHANGE subcommand, under the SCOPE setting, allows you to change existing scope records.

Syntax. The CHANGE subcommand has the following syntax:

```
CHANGE {*/record-name/LIKE(record-name-mask)} [ADD/REP/DEL]
                    {[DSN(index,index...)]
                     [INF(storage-key,storage-key...)]
                     [LID(logonid-mask,logonid-mask...)]
                     [UID(uid-mask,uid-mask...)]}
```

Example. In its simplest form, this subcommand requires a scope record name and at least one of the parameters DSN, INF, LID, or UID. For example, take the following INSERT and CHANGE subcommands:

```
insert limit3 dsn(payroll) lid(alt-) uid(1h*pr)
change limit3 lid(pay-)

list limit3
ACF60062 SCOPE LIMIT3 STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYROLL) LID(PAY-)
```

The example CHANGE subcommand changes a scope record named LIMIT3, which was created by the INSERT subcommand. After the change, LIMIT3 has a DSN field with the value PAYROLL, and a LID field with the value PAY-. (The ADD parameter is assumed, as discussed later.)

Parameters. The CHANGE subcommand takes the following parameters:

*
An asterisk specifies that the change apply to the last scope record processed since establishing the SCOPE setting.

record-name
This parameter specifies the name of the existing scope record to be changed. This name is 1 to 8 alphanumeric characters.

LIKE(record-name-mask)
The LIKE parameter specifies a 1- to 8-character mask for the names of scope records to be changed.

ADD
The ADD parameter indicates that any values specified by the DSN, INF, LID, and UID parameters of the CHANGE subcommand will be added to any existing fields copied from the model record. For example, the following CHANGE subcommand changes the scope record named SCOPE1, which was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 add dsn(account)

list scope1
ACF60062 SCOPE SCOPE1 STORED BY PAYSDH ON 07/15/85-11:43
DSN(PAYROLL,ACCOUNT) LID(PAY-) UID(1H*PR-)
```

As listed after this change, SCOPE1 contains a DSN field with both the values PAYROLL and ACCOUNT. The ADD parameter is assumed if you omit the ADD, REP, and DEL parameters.

REP

The REP parameter indicates that any values specified in the DSN, INF, LID, and UID parameters of a CHANGE subcommand will replace any values in the corresponding fields of the existing scope record. If a field specified in the CHANGE subcommand does not currently exist in the scope record, then it will be added.

For example, the following CHANGE subcommand changes the scope record named SCOPE1, which was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 rep lid(act-)

list scope1
ACF60062 SCOPE SCOPE1 STORED BY PAYSDH ON 07/15/83-11:43
DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

As listed after this change, SCOPE1 contains a DSN field with the value PAYROLL, and a LID field with the value ACT- (which replaced the value PAY-).

DEL

The DEL parameter indicates that any values specified by the DSN, INF, LID, and UID parameters of the CHANGE subcommand will be deleted from the corresponding fields in the specified scope record. For example, the following CHANGE subcommand changes the scope record named SCOPE1, which was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 del lid(pay-) uid(1h*pr)
```

As listed after this change, the scope record SCOPE1 contains only a DSN field with the value PAYROLL. Both the LID and UID field values have been deleted.

DSN(index,index-mask...)

The DSN parameter restricts any existing dataset access so that it includes access to only the specified datasets (and any datasets to which access has been granted through access rule sets or other special privileges). The specified datasets are identified by high-level indices or a mask of the high-level indexes. A mask can be 1 to 8 alphanumeric characters. Multiple masks or high-level indexes must be separated by commas. If the LID and UID parameter are not specified, then dataset access is completely restricted. However, access can be granted through access rule sets or special privileges. Scoped users with similar authority as listed on the DECOMP field of the GSO OPTS record will still be able to decompile rules, regardless of the limits imposed on their scope.

INF(storage-key,storage-key...)

The INF parameter restricts Infostorage database access so that it includes access to only the specified generalized resource rule sets, entry records, scope records, shift/zone records, and control records. This parameter also restricts access to generalized resources. However, access can be permitted by generalized resource rules, special privileges, or pseudo dataset names. Scoped users with similar authority as listed on the INFOLIST fields of the GSO OPTS record will still be able to list Information Storage records regardless of the limits imposed on their scope. These specified rule sets and records are identified by a storage key of 1 to 44 characters. This storage key consists of:

1. A letter code indicating the <u>storage class,</u> as follows:

R	Generalized resource rule sets
E	Entry records
S	Scope records
T	Shift/zone records
C	Control (GSO) records

2. The three-character <u>type code</u> that identifies the type of record or generalized resource rule set. (This type code corresponds with the one used in establishing the RESOURCE, ENTRY, SCOPE, SHIFT, or CONTROL setting.)

3. Either the <u>key of the generalized resource rule set</u> or the <u>name of the Infostorage database record.</u>

For example, the storage key RCKCACFM consists of:

R	The storage class for a generalized resource rule set.
CKC	The type code for CICS transactions.
ACFM	The name of the transaction.

Multiple storage keys and storage key masks must be separated by commas or blank spaces. If this parameter is not specified and the DSN, LID, or UID parameter is, then access to generalized resources and to the Infostorage database is completely restricted. However, access may be granted through generalized resource rule sets, special privileges, or pseudo-dataset names.

LID(logonid-mask,logonid-mask...)

The LID parameter restricts Logonid record access so that it includes access to only the specified Logonid records. These specified records are identified by Logonids or Logonid masks of 1 to 8 characters. Multiple Logonids or masks must be separated by commas or blank spaces. If this or the UID parameter is not specified, then Logonid record access is completely restricted. However, access may be granted through special privileges.

UID(uid-mask,uid-mask...)

The UID parameter restricts Logonid record access so that it includes access to only the specified Logonid records. These specified records are identified by UID strings or UID string masks of 1 to 24 characters. Unlike other UID parameters for ACF2, this UID parameter requires that any UID mask end with a dash (-). Blanks characters are not automatically inserted to complete the 24-character string. Multiple UID strings or masks must be separated by commas or blank spaces. If this or the LID parameter is not specified, then Logonid record access is completely restricted. However, access may be granted through special privileges.

LIST Subcommand--SCOPE Setting

The LIST subcommand, under the SCOPE setting, allows you to list existing scope records.

Syntax. The LIST subcommand has the following syntax:

```
LIST {*/record-name/LIKE(record-name)}
                    [ALL/DSN/INF/LID/UID]
```

Example. In its simplest form, this subcommand requires a scope record name. For example:

```
list scope1

ACF60062  SCOPE  SCOPE1  STORED  BY  PAYSDH  ON  07/15/83-11:43
DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

The example LIST subcommand lists the contents of a scope record named SCOPE1. By default, all existing fields are listed.

Parameters. The LIST subcommand takes the following parameters:

*

An asterisk specifies the listing of the last scope record processed since the current SCOPE setting was established

record-name

This parameter specifies the name of an individual scope record to be listed. This name is any 1 to 8 characters.

LIKE(record-name)

The LIKE parameter specifies a 1- to 8-character mask for the names of scope records to be listed.

ALL
The ALL parameter indicates that all fields of the specified scope record(s) will be listed. ALL is assumed by default unless the ALL, DSN, INF, LID, or UID parameter is specified.

DSN
The DSN parameter indicates that the DSN field of the specified scope record(s) will be listed.

INF
The INF parameter indicates that the INFO field of in the specified scope record(s) will be listed.

LID
The LID parameter indicates that the LID field of the specified scope record(s) will be listed. UID is required.

UID
The UID parameter indicates that the UID field of the specified scope record(s) will be listed. LID is required.

DELETE Subcommand--SCOPE Setting

The DELETE subcommand, under the SCOPE setting, allows you to delete existing scope records.

Syntax. The DELETE subcommand has the following syntax:

```
DELETE {*/record-name/LIKE(record-name)}
```

Example. In its simplest form, this subcommand requires a scope record name. For example:

```
delete scope1
```

This example DELETE subcommand deletes a scope record named SCOPE1. All fields are also deleted.

Parameters. The DELETE subcommand takes the following parameters:

*
An asterisk specifies deletion of the last scope record processed since the current SCOPE setting was established.

record-name
This parameter specifies the 1- to 8-character name of an individual scope record to be deleted.

LIKE(record-name)

The LIKE parameter specifies a 1- to 8-character mask for the names of scope records to be deleted.

SHIFT SETTING: SHIFT/ZONE RECORDS

Shift/zone records are used to validate accesses to the system, datasets, and resources. They can:

1. Define the days, dates, and times at which accesses can be made.

2. Define the offset from CPU time of the time zone in which the access attempt is being made.

This chapter discusses:

1. Examples of shift/zone records

2. Types and names of shift/zone records

3. Fields of shift/zone records

4. The user's LOGSHIFT attribute

5. Creating shift/zone records

6. ACF subcommands under the SHIFT setting

EXAMPLES OF SHIFT/ZONE RECORDS

This section shows and explains examples of a shift record and a zone record.

Example of a Shift Record

When listed, a shift record might look like:

```
ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) NTIME(1200-1300)
    INCLUDE(NHOLIDAY)
```

This record can be explained as follows:

1. SHIFT REGULAR indicates that the above example shows a shift record named REGULAR.

2. DAYS(MO,TU,WE,TH,FR) defines the shift as including the normal working days in a week.

3. TIME(0800-1700) further defines the shift as including only the hours 08:00 through 16:59 of the specified days.

4. NTIME(1200-1300) indicates that the shift will exclude the lunch hour (12:00 through 12:59) of the specified days.

5. INCLUDE(NHOLIDAY) futher defines the shift as including whatever days, dates, and times are specified or excluded by another shift record named NHOLIDAY.

Only when specified in the SHIFT field of a user's Logonid record will this shift record be used during access validation to make sure that the user accesses the system only during the specified days, dates, and times. This record can also be specified in access and generalized resource rules to define the days, dates, and times at which access to certain data and resources can be granted.

Example of a Zone Record

When listed, a zone record might look like:

```
ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/85-10:42
 ADJUST(+0600)
```

This record can be explained as follows:

1. ZONE SYD indicates that the above example shows a zone record named SYD (to represent a time zone for Sydney, Australia).

2. ADJUST(+0600) defines the time zone as being six hours ahead of the zone in which the CPU is running.

Only when specified in the ZONE field of a user's Logonid record will this zone record apply. It will allow for the proper time validation whenever the user makes an access attempt that involves validation according to a shift record.

TYPES AND NAMES OF SHIFT/ZONE RECORDS

As shown in the previous examples, two types of shift/zone records exist:

1. Shift records are identified by a type code of SFT and a name of any 1 to 8 characters.

2. Zone records are identified by a type code of ZON and a name of any 1 to 3 characters.

FIELDS OF SHIFT/ZONE RECORDS

This sections describes the fields of shift and of zone records.

Fields of Shift Records

A shift record must contain both a DAYS or NDAYS and TIME or NTIME fields since the defaults for these are NDAYS and NTIME.

DAYS
Specifies the days of the week or individual dates when access will be granted.

NDAYS
Specifies the days of the week or individual dates when access will not be permitted.

TIME
Specifies the times during the specified days/dates when access will be granted.

NTIME
Specifies the times during the specified days/dates when access will not be permitted.

INCLUDE
Specifies a another shift record that will be included as part of this shift record.

A shift may specify multiple days/dates, time periods, and other shifts as necessary. However, access will be denied unless one of the day fields and one of the time fields is specified on the shift record.

Fields of Zone Records

A zone record can have only an ADJUST field, which defines the offset between the time zone in which an access attempt is being made and the time zone in which the CPU is running.

THE LOGSHIFT USER ATTRIBUTE

The LOGSHIFT field of a user's Logonid record allows the user to access the system, data, and resources outside any shift specified in the SHIFT record of the user's Logonid record. However, any such access is logged and will be reported on the Invalid Password/Authority Log (ACFRPTPW). This report is described in the acf2/MVS Utilities Manual.

CREATING SHIFT/ZONE RECORDS

This section explains the creation of shift and of zone records by using the ACF command.

Creating Shift Records

A shift record is created by first issuing the ACF command, establishing the SHIFT(SFT) setting, and then issuing an INSERT subcommand:

```
acf
set shift(sft)
insert regular days(mo,tu,we,th,fr) time(0800-1700)
```

In the above example, the INSERT subcommand is used to create a shift record named REGULAR. This shift is defined as the weekdays Monday through Friday during the hours of 08:00 through 16:59.

If the INSERT subcommand cannot fit on one line, you can add other fields and values to the shift record by using the CHANGE subcommand:

```
change regular add ntime(1200-1300)
```

In the example, the CHANGE subcommand adds an NTIME field that denies access during the hours 12:00 though 12:59. (Access can take place at 13:00.)

Creating Zone Records

A zone record is created by first issuing the ACF command, establishing the SHIFT(ZON) setting, and then issuing an INSERT subcommand:

```
acf
set shift(zon)
insert syd adjust(+0600)
```

In the above example, the INSERT subcommand is used to create a zone record named SYD. This record defines Sydney time as being six hours ahead of the CPU time.

ACF SUBCOMMANDS UNDER SHIFT SETTING

To process shift and zone records, you must establish the SHIFT setting of the ACF command. Use one of the following formats of the SET subcommand:

```
SET SHIFT(SFT)   For shift records
SET SHIFT(ZON)   For zone records
```

After you have established the SHIFT setting, you can issue any of the following ACF subcommands:

```
*INSERT      *DELETE      SET
*CHANGE       END         SHOW
*LIST         HELP        SN
```

The common subcommands END, HELP, SHOW, and SN operate under the SHIFT setting as explained in the chapter on the ACF command. The following text describes the function, syntax, and parameters of the other subcommands, marked by the asterisks (*).

The operation of INSERT and CHANGE subcommands differ for shift record processing as opposed to zone record processing. Separate explanations of these subcommands are provided for these two different types of records.

INSERT Subcommand--SHIFT(SFT) Setting

The INSERT subcommand, under the SHIFT(SFT) setting, allows you to insert new shift records.

Syntax. The INSERT subcommand has the following syntax:

```
INSERT  {*/record-name} DAYS(mo,tu,we,th,fr,sa,su, -
        m1/d1/y1,m2/d2/y2,...,mn/dn/yn) -
        TIME(h1m1-h2m2,h3m3-h4m4,...,hnmn-hpmp) -
        [NDAYS([days(mo,tu,we,th,fr,sa,su, -
        m1/d1/y1,m2/d2/y2,...,mn/dn/yn)] -
        [NTIME(h1m1-h2m2,h3m3-h4m4,...,hnmn-hpmp] -
        [INCLUDE(shift-name1,shift-name2,...,shiftnamen)]
```

Parameters. The INSERT subcommand takes the following parameters:

*
Specifies the name of the last shift record processed since the SHIFT(SFT) setting was established.

record-name
Specifies a 1- to 8-character name of the shift record to be inserted.

DAYS(MO,TU,WE,TH,FR,SA,FR,mm/dd/yy,dd/mm/yy, or yy/mm/dd)
(Required) Specifies two-character abbreviations for days of the week to be included in the shift (i.e., SU for Sunday, MO for Monday, etc.). This parameter can also include numeric dates in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd. This format is defined in the DATE field of the OPTS record, as described in the chapter on GSO records. Multiple days and dates must be separated by commas or blank characters.

NDAYS(MO,TU,WE,TH,FR,SA,FR,mm/dd/yy,dd/mm/yy, or yy/mm/dd)
Specifies the days and dates that should be excluded from those specified in the DAYS parameter. This parameter may include two-character abbreviations for days of the week and numeric dates in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd. Multiple days and dates must be separated by commas or blank characters.

TIME(hhmm,hhmm,...)
(Required) Specifies the range of hours and minutes, based on a 24-hour clock, to be included in the shift. For example, TIME(0800-1700) represents the hours from 08:00 through 16:59. You can specify multiple time entries by separating them with commas, e.g., TIME(0800-1700,1800-1900). The TIME parameter must be specified, since it defaults to no allowable time period.

NTIME(hhmm,hhmm...)
Specifies the time period to be excluded from that specified by the TIME parameter. For example, if an installation wanted to allow access during the hours 09:00 through 16:59 but not during 12:00 through 12:59, the installation could specify the shift record parameters:

```
TIME(0900-1700) NTIME(1200-1300)
```

INCLUDE(shift-name,shift-name,...)
Specifies any existing shift record to be included as part of the one being defined. For example, a shift record name NORMAL can be included in a shift record named SPECIAL:

```
INSERT NORMAL DAYS(MO,TU,WE,TH,FR) TIME(0900-1700)

INSERT SPECIAL INCLUDE(NORMAL) DAYS(SA)
```

The shift record named SPECIAL will grant access not only during weekdays from 09:00 through 16:59, but also during those hours on Saturday.

Parameter Requirements. The INSERT subcommand requires all shift records to contain a DAYS, DATES, or NDAYS parameter. Furthermore, ACF2 does not automatically grant access for days, dates, and times excluded from the NDAYS and NTIMES parameters. Any access to be allowed by a shift record must be explicitly indicated.

Making an Inserted Record Become Active. When you insert or change any shift records, ACF2 automatically reloads the new information at midnight (24:00) each day. To reload that information sooner, the F ACF2,NEWSHIFT operator command must be executed from the console. For further information, see the chapter on console operator commands.

INSERT Subcommand--SHIFT(ZON) Setting

The INSERT subcommand, under the SHIFT(ZON) setting, allows you to insert a zone record.

Syntax. This INSERT subcommand has the following syntax:

```
     INSERT {*/zone-name} ADJUST(+hhmm/-hhmm)
```

Parameters. The INSERT subcommand takes the following parameters:

record-name
   Specifies a 1- to 3-character time zone name.

ADJUST(+hhmm/-hhmm)
   Specifies a positive or negative value in hours and minutes to be added or subtracted from the processing CPU time. For example, if the processing CPU is in London, then a time zone for Sydney might be specified with the parameter ADJUST(+0800). A time zone for New York might be specified by ADJUST(-0800).

CHANGE Subcommand--SHIFT(SFT) Setting

The CHANGE subcommand, under the SHIFT(SFT) setting, allows you to add, replace, or delete fields from an existing shift record.

Syntax. This CHANGE subcommand has the following syntax:

```
     CHANGE  {*/record-name/LIKE(record-name-mask)} -
             {ADD/REP/DEL}
             {[DAYS(mo,tu,we,th,fr,sa,su,mm/dd/yy,...)] -
             [NDAYS(mo,tu,we,th,fr,sa,su,mm/dd/yy,...)]} -
             [TIME(hhmm,hhmm,...)] [NTIME(hhmm,hhmm,...)] -
             [INCLUDE(shift-name,shift-name,...)]
```

Example. Take the following shift record named REGULAR:

```
     acf
     set shift(sft)
     list regular

      ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
       DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)
```

To add new values to a field in a shift record, you must issue the CHANGE subcommand with at least the name of the record and field and new value to be added. By default, if the ADD, REP, or DEL parameter is not specified, then the ADD parameter is assumed:

```
change regular days(sa)
```

This subcommand will add Saturday (SA) to any other days already specified in the DAYS field of the above shift record. After you issue the CHANGE subcommand, the system responds with the new contents of the record:

```
acf
set shift(sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular days(sa)
  DAYS(MO,TU,WE,TH,FR,SA) TIME(0800-1700)
```

If the value SA had already existed in the record, then that value would remain. Through the REP and DEL parameters, values can be replaced or deleted, respectively.

Parameters. The CHANGE subcommand takes the following parameters. The asterisk (*), the record-name, and the LIKE, ADD, REP, and DEL parameters are positional. The asterisk or the record-name must immediately follow the CHANGE subcommand keyword. If specified, the ADD, REP, DEL parameter must always come next.

*
: Specifies the name of the last shift record processed since the SHIFT(SFT) setting was established.

record-name
: Specifies a 1- to 8-character name of the shift record to be changed.

LIKE(shift-record-name)
: Specifies a mask for the names of the shift records to be changed. To mask the names of shift records, follow the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting).

ADD

Indicates that any values specified for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters will be added to any existing values in the corresponding shift record fields. If any new value already exists in a shift record field, then that value will remain. For example:

```
acf
set shift (sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular add days(fr,sa) ndays(03/24/86)
  DAYS(MO,TU,WE,TH,FR,SA) NDAYS(03/24/85) TIME(0800-1700)
```

The above CHANGE Subcommand adds Friday (fr) and Saturday (sa) to the DAYS field of the shift record. (Friday already exists in that field so this value remains.) An NDAYS field is also added.

REP

Indicates that any values specified for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters will completely replace the corresponding shift record fields. If any new field is specified, then that field and its value(s) will be added to the record. For example:

```
acf
set shift (sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular rep days(fr,sa) ndays(03/24/86)
  DAYS(FR,SA) NDAYS(03/24/85) TIME(0800-1700)
```

Through the above CHANGE subcommand, Friday (fr) and Saturday (sa) completely replace other values in the DAYS field of the shift record. The NDAYS field is added.

DEL

Indicates that any values specified for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters will be deleted from the corresponding shift record fields. If any specified value does not exist in a field, then that value is ignored. For example:

```
acf
set shift (sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular del days(fr) time(0800-1700,2200-2300)
  DAYS(MO,TU,WE,TH)
```

The above CHANGE subcommand deletes Friday (fr) as one of the values in the DAYS field of the shift record. The TIME field is completely deleted. The value 2200-2300 is ignored since it does not exist.

DAYS(MO,TU,WE,TH,FR,SA,FR,mm/dd/yy,dd/mm/yy, or yy/mm/dd,...)

Specifies two-character abbreviations for days of the week (i.e., SU for Sunday, MO for Monday, etc.). This parameter can also include numeric dates in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd. This format is defined in the DATE field of the OPTS record, as described in the chapter on GSO records. Multiple days and dates must be separated by commas or blank characters.

NDAYS(MO,TU,WE,TH,FR,SA,FR,mm/dd/yy,dd/mm/yy, or yy/mm/dd)

Specifies the days and dates that should not be included with those specified in the DAYS parameter. This parameter may include two-character abbreviations for days of the week and numeric dates in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd. Multiple days/dates must be separated by commas or blank characters.

TIME(hhmm-hhmm,hhmm-hhmm...)

Specifies a range of hours and minutes, based on a 24-hour clock. For example, TIME(0800-1700) represents the hours from 08:00 through 16:59. You can specify multiple time entries by separating each entry with commas, e.g., TIME(0800-1700,1800-1900).

NTIME(hhmm-hhmm,hhmm-hhmm,...)

Specifies the time period to be excluded from that specified by the TIME parameter.

INCLUDE(shift-name,shift-name,...)

Specifies any existing shift record to be included as part of the one being defined. For example, a shift record named REGULAR (listed below) includes the shift named NHOLIDAY. A CHANGE subcommand can be used to change the record REGULAR so that it includes the shift OFFTIME instead:

```
acf
set shift (sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/85-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) INCLUDE(NHOLIDAY)

change regular rep include(offtime)
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) INCLUDE(OFFTIME)
```

CHANGE Subcommand--SHIFT(ZON) Setting

The CHANGE subcommand, under the SHIFT(ZON) setting, allows you to change the offset specified in a zone record.

Syntax. This CHANGE subcommand has the following syntax:

```
CHANGE  {*/record-name/LIKE(record-name-mask)}
```

Example. The following zone record named SYD defines the user's time as being 8 hours ahead of the executing CPU time:

```
acf
set shift(zone)
list syd

 ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/85-10:52
  ADJUST(+0800)
```

To change this offset, issue the CHANGE subcommand with the name of the record and new offset:

```
change syd adjust(+0700)
```

The system responds with the new offset:

```
change syd adjust(+0700)
  ADJUST(+0700)
```

Parameters. The CHANGE subcommand takes the following parameters:

*
Specifies the name of the last zone record processed since the SHIFT(ZON) setting was established.

record-name
Specifies a 1- to 8-character name of the zone record to be changed.

LIKE(record-name-mask)
Specifies a mask for the names of the zone records to be changed. To mask the names of zone records, follow the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting).

LIST Subcommand--SHIFT Settings

The LIST subcommand, under the SHIFT(SFT) or SHIFT(ZON) settings, allows you to list shift or zone records, respectively.

Syntax. The LIST subcommand has the following syntax:

```
LIST  {*/record-name/LIKE(record-name-mask)}
```

Example. Under the SHIFT(SFT) setting, you can list a shift record by issuing the LIST subcommand with the shift record name:

```
acf
set shift (sft)
list regular

 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/84-10:38
  DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)
```

Under the SHIFT(ZONE setting), you can list a zone record by issuing the LIST subcommand with the zone record name:

```
acf
set shift(zon)
list syd

 ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/84-10:52
  ADJUST(+0800)
```

Parameters. The LIST subcommand takes the following parameters:

*
Specifies the name of the last shift or the last zone record processed since the current SHIFT setting was established.

record-name
Specifies a 1- to 8-character name of the shift or the zone record to be listed.

LIKE(record-name-mask)
Specifies a mask for the names of the shift or the zone records to be listed. To mask the names of shift or zone records, follow the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting).

DELETE Subcommand--SHIFT Settings

The DELETE subcommand, under the SHIFT(SFT) or SHIFT(ZON) settings, allows you to delete shift or zone records, respectively.

Syntax. The DELETE subcommand has the following syntax:

```
DELETE {*/record-name/LIKE(record-name-mask)}
```

Example. Under the SHIFT(SFT) setting, you can delete a shift record by issuing the DELETE subcommand with the shift record name:

```
acf
set shift (sft)
delete regular
```

If the deletion is successful, the system will respond with the message: DELETED.

Parameters. The DELETE subcommand takes the following parameters:

*
Specifies the name of the last shift or the last zone record processed since the current SHIFT setting was established.

record-name
Specifies a 1- to 8-character name of the shift/zone record to be deleted.

LIKE(record-name-mask)
Specifies a mask for the names of the shift or the zone records to be deleted. To mask the names of shift/zone records, follow the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting).

CONTROL SETTING: GLOBAL SYSTEM OPTION (GSO) RECORDS

Control records are of two types, GSO and TSO. GSO records define system options, such as operating and performance parameters, and installation exits. TSO records are used to provide the fullscreen logon/parameter retention facility available to TSO users. The following sections discuss GSO records and facilities. TSO records are discussed in a subsequent section.

Relationship to Options of the ACFFDR. In addition to the options defined by GSO records, the ACF2 Field Definition Record (ACFFDR) also defines some system options. However, these options are static in nature and are rarely modified by an installation after the install of ACF2. Any changes made to the options in the ACFFDR require reassembly and a system IPL to make them active, whereas changes to GSO records can be made active dynamically.

In this Chapter. A discussion of the following topics is contained in this chapter:

1. Example of a GSO record
2. GSO record names, referred to throughout as RECIDs
3. GSO record fields which are unique to each record
4. SYSID concept
5. GSO record maintenance
6. ACF subcommands for processing GSO records
7. Console operator commands for controlling GSO facilities

EXAMPLE OF A CONTROL RECORD

When listed, a Global System Option (GSO) record is formatted as shown below:

```
ABC1 / PSWD LAST CHANGED BY ADMISO ON 11/30/84-14:52
                ENCRYPT(XDES) MAXTRY(2) MINPSWD(5) PASSLMT(3) PSWDALT
                PSWDFRC PSWDJES WRNDAYS(3)
```

Each GSO record consists of three components, the SYSID (described later), the RECID or record name and data fields. Our example above is constructed as follows:

1. ABC1 represents the SYSID associated with the GSO record. The SYSID is explained in a subsequent section of this chapter.

2. PSWD is the record name or RECID. This record defines system options relating to password handling.

3. The data appearing on the second and third lines depict the actual record fields and their content. For example, the content of the ENCRYPT field is XDES.

The meaning of each field is fully explained in the section describing the specific RECID; the PSWD section describes this example.

GSO RECORD NAMES - RECIDS

RECIDs for GSO records are selected by SKK and are not installation definable or modifiable. These pre-defined RECIDs are listed below, together with their basic functions.

APPLDEF
Defines the format of secondary authentication information storage records.

AUTHEXIT
Contains the vendor or installation exit information supporting the secondary authentication facility.

AUTOERAS
Controls the automatic physical erasure of the VSAM, NONVSAM, or volumes named in this record.

BACKUP
Provides the ability to dynamically maintain the space allocation and location of the ACF2 sequential backup work files. This record also contains the CPU, command string information and time in which the automatic database backup utility is to occur.

BLPPGM
Specifies those programs that are authorized to use tape Bypass Label Processing (BLP).

EXITS
Specifies the module names of installation-written ACF2 exit routines.

LINKLST
Specifies one or more partitioned datasets that will be considered part of the system LINKLIST (SYS1.LINKLIB) during dataset access validation.

LOGPGM
Specifies those programs for which all dataset accesses will be logged.

MAINT
Specifies the Logonid, program, and library combinations used for system maintenance functions.

NJE
Specifies ACF2 validation options that apply to jobs submitted through a Network Job Entry subsystem (JES2, JES3, RSCS).

OPTS
Contains a number of global options that control logging, rule maintenance and SAF.

PPGM
Defines protected programs which may only be executed by privileged users.

PSWD
Defines the user password controls.

RESDIR
Specifies those generalized resource directories that are to be made resident at ACF2 initialization time.

RESRULE
Specifies those dataset access rules that are to be made resident at ACF2 initialization time.

RESVOLS
Defines those DASD and mass storage volumes for which ACF2 is to provide dataset-level protection.

SAFMAPS
Enables the installation to define generalized resource rule types for IBM System Authorization Facility initiated validation requests.

SAFSAFE
Defines subsystem, control point and class combinations for which the ACF2 System Authorization Facility (SAF) interface will bypass validation.

SECVOLS
Defines those DASD, mass storage, and tape volumes for which ACF2 is to provide volume-level protection.

TSO
Specifies MVS TSO and TSO/E system-wide options and default logon parameters.

TSOCRT
Defines a screen-clear string used to obliterate the logon password on ASCII CRT devices.

TSOKEYS
Defines installation-supplied keywords that ACF2 will allow at TSO logon time.

TSOTWX
Defines an x-out mask used to obliterate the logon password on TWX devices.

TSO2741
Defines an x-out string used to obliterate the logon password on 2741 devices.

WARN
Specifies a warning message to be issued to the user when the system is in WARN mode and a violation is deleted.

Each GSO record (RECID) has a unique set of fields. These records and their associated fields are discussed following the explanation of the structure of GSO records.

GSO RECORD FIELDS

This section lists and discusses the fields of each of the GSO records previously listed.

APPLDEF--Extended Authentication Infostorage RECID Definitions

RECORD-ID	FIELDS

APPLDEFqual	APPLDIV(division-mask)
	RECID(rsb-module-name1/recid-mask1,....)

Function: Defines the type (division), the record structure block module containing the data format and the record id for each user authentication application storing records on the Infostorage database.

Fields:

APPLDIV(division-mask)
Specifies an identifier to group a set of related Infostorage records. The division or mask must be one of the eight @CFDE macro entries for user authentication. Standard ACF2 masking conventions can be substituted.

RECID(rsb-module-name1/recid-mask1,...)
A multi-value field. The first RECID sub-argument specifies the Record Structure Block (RSB) module name describing the user authentication record format for the APPLDIV defined above. For user authentication, a single RSB record format is usually defined for users of the same application. Most authentication applications will store authentication data for each user in a separate database record. Since the data is unique for each user, the user's Logonid serves as the record id of the I-AUT information storage record. Therefore, the second RECID sub-argument will usually have a record id mask of '-' specified.

Specification: Optional. This record must be created when the INFOSTG option is selected on the AUTHEXIT record. A total of 8 different APPLDEF records can be created to support 8 different user authentication aplications. To differentiate each application, a qualifier should be appended to the record id. The qualifier can be as many as 8 characters.

Notes: An APPLDEF GSO record will define the division, data format (RSB) module and record id for each application's user authentication routine. The APPLDIV field is similar to the SYSID field for GSO records. DIVISION can be specified on the SET subcommand.

The Record Structure Block (RSB) defines the field composition of the records. Since record content will vary according to application, the application defines an ACF2 RSB. Any information the application needs can be placed in the data portion of the authentication record and referenced by the application-defined field name.

Further information can be found in the appropriate section of this manual, and in the System Programmer's Guide.

SHOW Command: The SHOW FIELDS command will display the Logonid attributes related to extended authentication. The SHOW ACTIVE command displays the records used for extended authentication.

SHOWALL will display the applications defined to the system.

```
   --APPLICATION DEFINITIONS: CLASS-TYPE/DIVISION/RECID/MODULE--
     I-AUT   OID          ********    ACFDROID
     I-AUT   LAZER        ********    LZR001
```

If no APPLDEF GSO records are defined, the SHOW APPLDEF command will display the title:

```
   --NO APPLICATION DEFINITIONS EXIST ON THIS SYSTEM--
```

AUTHEXIT--Extended User Authentication Exit

RECORD-ID	FIELDS

AUTHEXITqual	LIDFIELD(attribute name)
	PROCPGM(processing-program-name)
	INFOSTG/NOINFOSTG

Function: Defines the name of the user extended authentication program invoked after TSO logon. Logonids with the corresponding LIDFIELD attribute will be processed by this program.

Fields:

LIDFIELD(attribute-name)
The Logonid attribute used to trigger the extended user authentication application program named in the associated PROCPGM field. This attribute name can be a total of 8 characters. It must correspond with the associated Logonid attribute name on the @CFDE macro.

PROCPGM(processing-program-name)
The name of the extended authentication exit. This exit program is invoked after TSO logon validation if the Logonid contains the appropriate LIDFIELD attribute. This name can also be a total of 8 characters.

INFOSTG/NOINFOSTG
Indicates whether the extended authentication exit will store information on ACF2's Infostorage database. Information about the record format and type is defined on a corresponding APPLDEF GSO record. The actual data required by the authentication program is stored on the ACF2 Infostorage database. You can create these records under the IDENTITY(AUT) setting. Refer to the appropriate section of this manual or the System Programmer's Guide for further information.

Notes: One AUTHEXIT record must be defined per user authentication application. A total of 8 different applications are supported. To differentiate between each AUTHEXIT record, a qualifier can be appended to the record name. As many as eight characters can be used for the qualifier.

Further information about extended user authentication support can be found in the System Programmer's Guide.

Specification: Optional. A total of 8 different AUTHEXIT records can be specified. If the INFOSTG option is selected, a corresponding APPLDEF GSO record must also be defined.

SHOW Command: The SHOW ACTIVE and SHOWALL commands display the active GSO AUTHEXIT records in the installation.

```
--AUTHENTICATION EXITS ON THIS SYSTEM: LIDFLD/PROCESS PROGRAM/INFOSTG
  AUTHSUP2/ACFEAXIT/INFOSTG
  OID/ACFEAOID/INFOSTG
```

The SHOW FIELDS command displays the Logonid attribute names defined by the installation.

AUTOERAS Record--Automatic Erase Feature

RECORD-ID	FIELDS
AUTOERAS	NON-VSAM/NONON-VSAM VSAM/NOVSAM VOLS(volmask1,....volmask255)

Function: Specifies the types of data in which a physical erasure is to be performed during deletion prior to the release of that space to the system for later allocation.

Fields:

NON-VSAM/NONON-VSAM
Specifies whether or not acf2/MVS will automatically erase non-VSAM datasets before releasing the space for future use. The non-VSAM erase is invoked via JCL disposition processing, dynamic unallocation (SVC99), a system utility (IEHPROGM), or a user program. The default is NONON-VSAM which deactivates Automatic Erase for non-VSAM datasets. See also the VOLS description below.

VSAM/NOVSAM
Specifies whether acf2/MVS will automatically erase VSAM dataspaces. The VSAM erase is invoked during IDCAMS delete processing. If VSAM is specified, all VSAM dataspaces are automatically erased during IDCAMS delete processing. The default of NOVSAM deactivates the VSAM Automatic Erase feature.

VOLS(volser, vol-mask)
Identifies a set of DASD volumes. Automatic erasing of a non-VSAM dataset will be done only if the deleted dataset resides on one of the volumes specified. Note that VOLS applies only to non-VSAM datasets. This parameter may be specified as a list of 1-6 character volume serial numbers or as masked patterns. For example, if "NON-VSAM VOLS(PROD01,PROD06)" is specified, only non-VSAM datasets that reside on the DASD volumes PROD01 and PROD06 are automatically erased. Datasets that reside on other volumes (PROD03 or PROD04 for example) are not automatically erased.

Specification: Both the VSAM and non-VSAM automatic erase functions are dynamically implanted by acf2/MVS at system-provided exit points. They function either through batch or online and are not dependent on the use of an indexed VTOC.

The SHOW STATE subcommand displays the options and values defined in the AUTOERAS global system options record. SHOW ACTIVE indicates which dynamically implanted intercepts have gained control.

Removing Note 7

Installations using the NOTE 7 version of the Automatic Erase feature must de-install NOTE 7 during the Release 4.1 install. A jobstream named DENOTE7 is supplied on the Release 4.1 distribution tape for this purpose.

The following chart illustrates the various protection mechanisms and the effect on the data types shown.

PROTECTION MECHANISMS

DATA TYPE	JCL	SVC99	ACF2 Utilties	Automatic Erase
VSAM	I	I	U	S
Non-VSAM	I	I	U(ACF2)	S
Temporary Non-VIO	I	I	U(ACF2)	S
VIO	S	S	S	S

where:

I = impossible
U = user action required
U(ACF2) = possible with user-invoked ACF2 utilities
S = action automatically taken by system

BACKUP Record--Automatic Backup Options

RECORD-ID	FIELDS

BACKUP	CPUID(smfid)
	STRING(string)
	TIME(hh:mm/00:01)
	PRISPACE(nnn)
	SECSPACE(nnn)
	WORKVOL(volser#)
	WORKUNIT(VIO,device-type)

Function: Specifies the ACF2 automatic backup procedures for the databases (i.e., the Logonid, Rule, and Infostorage databases). Optionally, this record specifies an OS/VS console command which ACF2 issues internally upon successful completion of backup processing. It also provides the ability to dynamically allocate the ACF2 Backup Work files if they were not pre-allocated. Refer to the System Programmer's Guide for more information.

Fields:

CPUID(smfid)
The SMF-id of the CPU designated to take the automatic backups in a multi-CPU environment. If this field is specified, ACF2 will compare it with the actual MVS system SMF-id. ACF2 will bypass the automatic backup if the two do not match. Operator-initiated backups may be taken at any time from any CPU. Generally, a single CPU in a multi-system configuration should be designated as the sole automatic backup processor.

STRING(string)
Free-form text issued by ACF2 after completion of the backup. This text is usually an OS/VS start console command used to perform additional installation-required cleanup. As part of the ACF2 database recovery facility, a procedure named ACFBKUP is placed into SYS1.PROCLIB during the installation process. ACFBKUP or a similar facility can be used to REPRO the primary sequential backup datasets into the alternate VSAM clusters. Refer to Usage Notes, below.

If the STRING field is not specified, no console command is issued.

TIME(hh:mm/00:01)
Time of day (24-hour format) at which the backup is to be initiated. Default is 00:01 A.M. No automatic backup is performed if TIME(00:00) is specified.

PRISPACE(nnn)
Specify the amount of primary work space to be allocated for ACF2 backup processing. The default value is 5. The units are expressed in cylinders. This field will not be displayed if not entered.

SECSPACE(nnn)
Specify the amount of secondary workspace to be allocated for ACF2 backup processing. The default value is 5. Units are expressed in cylinders. This field will not be displayed if not entered.

WORKVOL(volser)
The volser of the volume to which the backup work files are allocated. There is no default value for the field. This field will not be displayed if not entered.

WORKUNIT (device type)
Indicates the device type on which ACF2 is to dynamically allocate its workfiles for backup processing. Device names are VIO, SYSDA, or DISK (VIO is the default). An installation defined common esoteric name can also be used. This field will not be displayed if not entered.

As ACF2 backup processing progresses from cluster to cluster, the number of records copied from each database is displayed. Two requirements must be met before ACFBKUP can be used:

(1) the JCL must be modified to reflect the installation's dataset names for both the primary sequential backup files and the alternate VSAM clusters.

(2) the alternate VSAM clusters must be initialized. Specify the STRING field as STRING(S ACFBKUP).

To deactivate the automatic backup, change the GSO BACKUP record to specify TIME(00:00). The NOBACKUP specified at ACF2 startup time may be used to deactivate the automatic backup facility for an individual CPU in a multi-CPU complex. Alternatively, the CPUID field can be used to designate the backup CPU. Note that backup processing may also be initiated independently by the console command:

```
F ACF2,BACKUP
```

Specification: Required.

SHOW Command: The SHOW SYSTEM and SHOW ACF2 subcommands of the ACF command display the values specified for fields in the BACKUP record.

BLPPGM Record--Tape Bypass Label Access Option

RECORD-ID	FIELDS

BLPPGMqual	LIBRARY(library)
	PGM(pgm1,pgm2,...,pgmn)

Function: Defines the programs and associated libraries which are authorized to use tape bypass label processing (BLP). A specified program from the designated library will be allowed BLP access regardless of the absense of TAPE-BLP or TAPE-LBC privileges in the Logonid record.

Fields:

LIBRARY(library)
Defines the fully-qualified name of the library in which the programs specified by the PGM parameter reside.

PGM(pgm1,pgm2,....,pgmn)
Defines up to 256 program names.

Usage Notes: If more than one BLPPGM record is required, a qualifier appended to the record name in the format BLPPGMqualifier to generate a unique RECID (e.g., BLPPGM001 or BLPPGM.LOADLIB). The qualifier can be up to eight characters.

Specification: Optional. A total of 256 unique program-name specifications may be defined.

SHOW Command: The SHOW PROGRAMS (or SHOW PGMS) and the SHOW ACF2 subcommands of the ACF command display all programs and library combinations which are authorized for tape bypass label access.

EXITS Record--Local ACF2 Exit Specifications

RECORD-ID	FIELDS
EXITS	DSNGEN(module)
	DSNPOST(module)
	EXPPXIT(module)
	INFOPRE(module)
	INFOPST(module)
	LGNIXIT(module)
	LGNPARMS(module)
	LGNPXIT(module)
	LGNTERM(module)
	NEWPXIT(module)
	RSCXIT1(module)
	RSCXIT2(module)
	RULEPRE(module)
	RULEPST(module)
	SRCXIT(module)
	STCXIT(module)
	SVCIXIT(module)
	VIOEXIT(module)
	VLDEXIT(module)

Function: Specifies the module name for each installation-written ACF2 exit. See the acf2/MVS System Programmer's Guide for complete information concerning each exit.

Fields:

DSNGEN(module)
Pseudo Dataset Name Generator exit.

DSNPOST(module)
Dataset Post-Validation exit. Note that if a DSNPOST exit is taken, then the VIOEXIT is not taken.

EXPPXIT(module)
Expired Password exit.

INFOPRE(module)
Information Storage Authorization Pre-processing exit.

INFOPST(module)
Information Storage Authorization Post-processing exit.

LGNIXIT(module)
TSO Logon Pre-Validation exit.

LGNPARMS(module)
Logon Parms exit.

LGNPXIT(module)
TSO Logon Post-Validation exit.

LGNTERM(module)
Terminal Identification exit.

NEWPXIT(module)
New Password exit.

RSCXIT1(module)
Resource Pre-Validation exit.

RSCXIT2(module)
Resource Post-Validation exit.

RULEPRE(module)
Access Rule Authorization Pre-processing exit.

RULEPST(module)
Access Rule Authorization Post-processing exit.

SRCXIT(module)
Source Name Modification exit.

STCXIT(module)
Started Task Validation exit.

SVCIXIT(module)
ACF2 SVC Initialization exit providing compatibility for those using ACF2 release 4.0 or earlier exits.

VIOEXIT(module)
Dataset Violation exit. Note that this exit will not be taken if a DSNPOST exit is specified.

VLDEXIT(module)
Dataset Pre-Validation exit.

Notes:

1. All exits must be linked into SYS1.LPALIB. LPA modules must be reentrant. To effect the new exit, a REFRESH can be done after it has been loaded into SYS1.LPALIB.

2. All exits, active or inactive, can be displayed with the SHOW ACTIVE and SHOW ACF2 subcommands of the ACF command.

3. Further information about these exits can be found in the System Programmer's Guide.

LINKLST Record--Logical Extension Of the System Linklist

RECORD-ID	FIELDS
LINKLST	LIBRARY(library-name1,library-name2,...,library-name64)

Function: Defines one or more program libraries that ACF2 will consider part of the system LINKLIST (SYS1.LINKLIB). Many installations use program pathing controls (e.g., PROGRAM and LIB access rule options) to define which programs are allowed access to a dataset. LINKLST provides installations with added flexibility with of ACF2's program pathing facility.

Fields:

LIBRARY(library-name1,library-name2,....library-name64)
Specifies up to 64 fully-qualified partitioned dataset (library) names. Each library-name may be up to 44 characters long. The default is LIBRARY(SYS1.LINKLIB).

SHOW Command: The SHOW LINKLST and SHOW ACF2 subcommands of the ACF command display all LINKLST option specifications.

Notes: The library name for TSO commands or programs not structured in the appropriate macro will be validated against the LINKLST record first. If the library name is not found, ACF2 will use the library(s) from which it executed in access rule validation. For further information about the program pathing macros, refer to the appropriate section of the System Programmer's Guide.

Specification: Required. All libraries must be cataloged before ACF2 startup.

LOGPGM Record--Dataset Access Logging Options

RECORD-ID	FIELDS
LOGPGM	PGMS(pgm1,pgm2,...,pgmn)

Function: Defines the set of programs for which all dataset access or execution is logged. These are programs for which the installation has an audit trail of activity, even though the installation elects not to restrict access to them by the GSO PPGM record. Refer to the section at the end of this chapter defining the relationship between the LOGPGM, PPGM and MAINT GSO records.

Fields:

PGMS(pgm1,pgm2,....,pgmn)
A maximum of 256 program names can be named. Programs AMASPZAP and IMASPZAP are the defaults.

Specification: Required. A total of 256 program names can be specified.

SHOW Command: The SHOW PROGRAMS (or SHOW PGMS) and the SHOW ACF2 subcommands of the ACF command display all program names defined in the LOGPGM record.

MAINT Record--System Maintenance Options

RECORD-ID	FIELDS

MAINTqual	LIBRARY(library)
	LID(logonid)
	PGM(pgm1,pgm2,...,pgmn)

Function: Defines the program, library, and Logonid that make up a maintenance environment. (Disk compression, archival, etc. are examples of standard system maintenance functions.) Those programs must reside in the specific library and be executed on behalf of the specified Logonid. When this environment is encountered, ACF2 rule validation is bypassed, thereby eliminating the generation of SMF logging records and the queueing of access rule sets in the address space. Any Logonid specified in the MAINT record must be defined as either non-cancellable (NON-CNCL) or maintenance (MAINT). If the Logonid is not marked MAINT or NON-CNCL, the maintenance program list will not be examined. Refer to the section at the end of this chapter describing the relationship between the MAINT, LOGPGM and PPGM GSO records.

Fields:

LIBRARY(library)
Defines a fully-qualified library dataset name in which the maintenance programs reside.

LID(logonid)
Logonid of an authorized maintenance user.

PGM(pgm1,pgm2,....,pgmn)
Specifies up to 256 maintenance program names.

Usage Notes: If more than one MAINT record is required, a qualifier can be appended to the record name to generate a unique RECID (e.g., MAINT.ABC). The qualifier can be up to eight characters.

Specification: Optional. A total of 256 MAINTqual program names may be specified.

SHOW Command: The SHOW PROGRAMS (or SHOW PGMS) and SHOW ACF2 subcommands of the ACF command display all program names defined by the MAINT record.

NJE Record--Network Job Entry Validation Options

RECORD-ID	FIELDS
NJE	INHERIT/NOINHERIT VALIN(YES/ONLY) VALOUT/NOVALOUT

Function: Specifies at which node(s) ACF2 job validation is to be performed in a Network Job Entry (NJE) environment. These NJE options apply only to ACF2 Version 3.1.4 (or higher) levels of the JES2 and JES3 interfaces. They have no effect if previous versions of the ACF2 interface for JES2 or JES3 are being used. Consult the acf2/MVS System Programmer's Guide for more information.

ACF2 can validate NJE jobs at the origin and/or execution node. If ACF2 is not active on the origin node, then the job will be sent without validation. In this case, the execution node will always perform validation. This may cause the job to fail if insufficient information is passed to the execution node to perform the validation.

Fields:

INHERIT/NOINHERIT

If INHERIT is specified, this execution node accepts network job inheritance. That is, a job sent to the execution node will inherit the Logonid and password of the person who submitted the job. If NOINHERIT is selected, then the job requires a Logonid and password if it is validated at the execution node. The job will not inherit the Logonid and password of the submitter. The default value is INHERIT.

VALIN(YES/ONLY)

If YES is specified, then this execution node will validate all incoming jobs. If ONLY is specified, then the execution node will not validate an incoming job that has already been ACF2-validated at the origin node. The default value is YES.

VALOUT/NOVALOUT

If VALOUT is specified, then this node will validate jobs going to other nodes for execution. If NOVALOUT is selected, then no validation is performed for outbound jobs. The default value is NOVALOUT.

Usage Notes: Each installation should review the implications of job inheritance when selecting these options. If a job is submitted without a password and the INHERIT option is active, the job will run under the submittor's Logonid. One word of caution: if network job inheritance is allowed and a duplicate Logonid exists on the execution node, different authorization (possibly including higher privileges than intended) may be granted to the job.

Specification: Required. One per system id.

SHOW Command: The SHOW SYSTEM and SHOW ACF2 subcommands display the NJE record.

OPTS Record--ACF2 Option Specifications

RECORD-ID	FIELDS
OPTS	$NOSORT/NO$NOSORT BLPLOG/NOBLPLOG CENTRAL/NOCENTRAL CHANGE/NOCHANGE CMDREC/NOCMDREC CONSOLE(NOROLL/ROLL/NONE/WTP CPUTIME(LOCAL/GMT) DATE(MDY/DMY/YMD) DECOMP(SECURITY,AUDIT/list) DFTLID(default-logonid) DFTSTC(ACFSTCID/logonid) INFOLIST(SECURITY,AUDIT/authority-list) JOBCK/NOJOBCK LABEXP(00:00/HH:MM) LABNUM(0/nnn) LIDRECL(512/1024) MAXVIO(10/nnn) MODE(ABORT/WARN/LOG/QUIET/RULE,no-rule,no-$mode) NOTIFY/NONOTIFY SAF/NOSAF SAFTRACE/NOSAFTRACE SHRDASD/NOSHRDASD STAMPSMF/NOSTAMPSMF STC/NOSTC TAPEDSN/NOTAPEDSN UADS/NOUADS VOLRULE/NOVOLRULE XBM/NOXBM

Fields:

$NOSORT/NO$NOSORT

Specifies whether the $NOSORT rule set control card will be processed. If the $NOSORT option is specified, then the $NOSORT control card will be recognized by ACF2 during rule compilation. During rule compilations, the $NOSORT control cards suppresses the normal ACF2 sorting of rules from most specific to most general. If the default option of NO$NOSORT is specified, ACF2 ignores any $NOSORT control cards during rule compilation, and automatically sorts rule sets.

BLPLOG/NOBLPLOG

Specifies whether BLP accesses, when they are authorized via the TAPE-BLP, TAPE-LBL, or BLPPGM attributes, should produce an ACF2 logging record. Default value is NOBLPLOG. Dataset ownership is determined by comparing the high-level index to the Logonid prefix value.

CENTRAL/NOCENTRAL

This option specifies whether the data owner is to have authority to store a set of access rules. By specifying CENTRAL, only Security Officers and users authorized by the %CHANGE or %RCHANGE feature will have this capability. Note that the NOSTORE attribute of the Logonid record and NOCENTRAL can be combined to give only selected users the ability to update their own rules. The default is NOCENTRAL. That is, all users will be able to update the access rule sets corresponding to datasets they own.

CHANGE/NOCHANGE

Specifies whether the access rule features, %CHANGE and %RCHANGE, are to be recognized. If NOCHANGE is specified, any %CHANGE or %RCHANGE cards in a rule set will be ignored when determining whether a user has the authority to replace any access rule. The default value is CHANGE, which activates %CHANGE and %RCHANGE authorization.

CMDREC/NOCMDREC

Specifies whether the TSO Command Statistics SMF records will be written for all users. Note that if NOCMDREC is specified, Command Statistics records can be written for individual users through the TSO-TRC attribute in the Logonid Record.

CONSOLE(NOROLL/ROLL/NONE/WTP)

Specifies how ACF2 dataset access logging messages (ACF99900) and dataset access security violation messages (ACF99913) will be generated, and where these messages are to be routed. Options are:

Option	Routcde	Descriptor Code	Action
NOROLL	1,9,11 (ACF99913)	2	Messages are issued to the user's joblog and to both the security console and operator's console as non-deletable.
NOROLL	9 (ACF99900)	2	Messages are issued to the security console as non-deletable.

ROLL	1,9,11 (ACF99913)	-	Messages are issued to the user's joblog and to both the security console and the operator's console as deletable.
ROLL	9 (ACF99900)	-	Messages are issued to the security console as deletable.
WTP	11 (ACF99913 and ACF99900)	-	Messages are issued to the user's joblog and to consoles selecting route code 11 messages. Console messages (if any) are sent as deletable.
NONE	- (ACF99913 and ACF99900)	-	Violation and logging messages are not issued (i.e., will appear neither on the console nor on the user's joblog).

NOROLL is the default.

CPUTIME(LOCAL/GMT)
Specifies the time setting of the CPU. This field determines how ACF2 will calculate a user's time of access when zone record processing is performed. If GMT (Greenwich Mean Time) is specified, ACF2 bases all time-zone calculations on the time-of-day (TOD) clock. In LOCAL setting, ACF2 first adjusts the TOD clock by the value stored in the CVTTZ field of the Communications Vector Table (CVT) and then bases all time-zone calculations on the adjusted TOD clock. The default is LOCAL.

DATE(MDY/DMY/YMD)
Indicates the date format to be used by ACF2: month-day-year (MDY), day-month-year (DMY), or year-month-day (YMD). Since dates are stored internally in Julian (YYDDD) format, there are no compatibility issues. Data formats may be freely changed. Default is MDY.

DECOMP(SECURITY,AUDIT/list)
Specifies which Logonid attributes are necessary to decompile an access rule or generalized resource rule which the user does not have the authority to change. The default value is SECURITY,AUDIT identifying that users with either attribute are authorized. This authority applies to all scoped or unscoped users with the privileges listed in this field. The Privileged Logonids listed in the DECOMP field are not affected by scoping.

DFTLID(default-logonid)
Specifies the default Logonid assigned to batch jobs entering the system that do not have a Logonid specified in their JCL. No default value is supplied for this option. A job entering the system without either a Logonid or a global default Logonid specified will be flushed.

DFTSTC(ACFSTCID/logonid)
Specifies the Logonid to be assigned to a started task if the procedure name is not defined as an ACF2 Logonid. This is only used if STC is also specified. Default value is ACFSTCID.

INFOLIST(SECURITY,AUDIT/authority-list)
Specifies which Logonid attributes are necessary to LIST Infostorage records which the user does not have the authority to change. It does not convey the authority to decompile generalized resource or access rules (such as entry lists, shift/zone records, and scope lists). Default value is SECURITY+AUDIT, indicating that users with either attribute are authorized. Logonids with the privileges listed in the INFOLIST field are not affected by scoping.

JOBCK/NOJOBCK
Indicates whether or not the Logonids submitting jobs via TSO are to be validated. When this option is used, Logonids with JOB privilege will be permitted to submit jobs via TSO. Does not affect foreground or XBM jobs, but does affect any background/batch jobs and any jobs submitted from these functions. The default value is NOJOBCK.

LABEXP(00:00/HH:MM)
Specifies the length of time, in hours and minutes, that each Lookaside Buffer (LAB) entry may remain in the LAB entry pool before it becomes eligible for automatic deletion. Note that the LAB entry is not automatically deleted unless the LABNUM value is reached and another LAB entry needs to be inserted into the pool. In addition, a system operator command can be used to manually delete all LAB entries, delete a group of LAB entries, delete all LAB entries that originated from a particular input source-id, or delete all LAB entries that originated from a specific NJE node. For additional information on LAB support, refer to the acf2/MVS System Programmer's Guide.

LABNUM(0/nnn)
Specifies the maximum number of ACF2 Lookaside Buffer (LAB) entries that can be stored on this CPU. LAB support provides base facilities for passing ACF2 Logonid records across multiple CPUs. The default is 0, which completely deactivates the support. The maximum number that can be specified is 32767. See the acf2/MVS System Programmer's Guide for complete information concerning LAB support.

LIDRECL(512/1024)
Specifies the length in bytes of the ACF2 Logonid record. The default is 1024. Note that only ACF2 Version 3.1.4 (or above) systems may specify 1024. See Usage Notes, below and the conversion section of the System Programmer's Guide.

MAXVIO(10/nnn)

The maxiumum number of security violations that may occur in a single job or TSO session before ACF2 will terminate the job. Default value is 10. Maximum value is 32767.

MODE(ABORT/WARN/LOG/QUIET/RULE,no-record,no-$mode)

The mode of the ACF2 system as it relates to dataset and resource access. The MODE determines what actions ACF2 will take when a request to access a dataset is considered a violation. The various MODEs are:

a) QUIET - disable ACF2 dataset access rule validations. Logonid, source, and similar system access validations will still take place.

b) LOG - log dataset access violations, but allow access to continue.

c) WARN - log dataset access violations, issue warning messages, and allow access to continue.

d) ABORT - log attempted violations, issue violation messages, and disallow the access. This is the default value.

e) RULE - An interim mode used by installations gradually migrating to full security. With this mode, the installation specifies what conditions exist in the absense of access rules or $MODE control cards in the access rule. Depending upon the condition, the dataset access will be validated accordingly. This method provides the ability to store access rules gradually while still permitting users access to the system while ACF2 is up and running. The value of the $MODE card may be QUIET, LOG, WARN, or ABORT as defined above. The $MODE control card has meaning only when the MODE(RULE,no-record,no-$mode) option is in effect and when ACF2 determines that a violation is to occur. The two positional parameters are:

no-record - specifies the action ACF2 is to take if NO access rule set is found when RULE mode is in effect. The "no-record" value may be QUIET, LOG, WARN, or ABORT as defined above.

no-$mode - specifies the action ACF2 is to take if no $MODE control card is found in the applicable access rule set when $MODE dataset access control is in effect. The "no-$mode" value may be QUIET, LOG, WARN, or ABORT as defined above.

For example, an installation selects MODE(RULE,WARN,ABORT). User ABC requests write access to dataset TEST.DATA. If no TEST rule set exists, ACF2 will base its decision on the no-record value, which in this case is WARN. If a TEST rule set does exist, but does not grant user ABC write access, ACF2 checks the $MODE control card in the access rule set and bases the allow or deny decision on the $MODE value. If $MODE(LOG) was specified in the access rule set, then user ABC is allowed to write into TEST.DATA and an ACF2 logging record is written. However, if no $MODE was specified in the rule set, the no-$mode value (in this case ABORT) would be used.

Always specify the REP command option when adding or modifying the $MODE parameter.

NOTIFY/NONOTIFY

Indicates if ACF2 will display the date, time, and input source-id of a user's last system access. If NOTIFY is specified, the message appears at logon time. Note that if a password change prompt or other ACF2 message must be displayed, the last access message will not be displayed. The default is NONOTIFY.

SAF/NOSAF

Activates or deactivates the ACF2/SAF (MVS System Authorization Facility) interface. Refer to the generalized resource rule section of the manual.

SAFTRACE/NOSAFTRACE

Determines whether the SAF exit will perform the trace function. NOSAFTRACE, the default, indicates that the SAF exit will not perform the trace function. If SAFTRACE is selected, information from the RACROUTE macro (control points, subsystem, class, request code, terminal id, user id, procedure, account number, and entity, where applicable) will be displayed at the security console.

SHRDASD/NOSHRDASD

Specifies whether or not ACF2 is to enforce shared DASD protocol to preserve the integrity of the VSAM clusters. Normally, ACF2 shared DASD logic is activated by the "shared" indicator in the UCB. It is recommended that SHRDASD be specified.

STAMPSMF/NOSTAMPSMF

Specifies whether ACF2 is to place the Logonid in the SMF User Identification Field at job initiation time. This will cause all SMF records (not just ACF2's SMF records) generated by the job to contain the Logonid. Care should be used in choosing the SMF stamp option because some installations and some other vendor supplied packages use this field for a communication area. If STC is specified, ACF2 will use the Started Task Name as the Logonid if it is present in the Logonid database. Otherwise the value in DFTSTC will be used. Dataset validation is performed, unless NOSTC is specified. NOSTC causes the DFLTSTC value to be used. Default value is NOSTAMPSMF.

STC/NOSTC

Specifies whether ACF2 is to validate dataset accesses by started tasks. Default value is NOSTC. Logonid records are not required for started tasks when NOSTC is in effect.

TAPEDSN/NOTAPEDSN

Specifies whether ACF2 is to protect datasets on tape volumes which do not match those listed in SECVOLS at the dataset level. MVS normally truncates a tape dataset name to the last 17 characters, so TAPEDSN should not be used unless a Tape Management System is present which will retain the entire dataset name. Default value is NOTAPEDSN, signifying no ACF2 protection for these tape volumes.

UADS/NOUADS

Specifies whether the TSO User Attribute Dataset (UADS) is used for TSO account, JCL procedures and user profile control. NOUADS, the default, implies that all TSO logon information is extracted from the user's Logonid record. UADS means that UADS is used. See the discussion in the acf2/MVS General Information Manual for further information.

VOLRULE/NOVOLRULE

Indicates the format to be used wherever ACF2 creates a pseudo dataset name for volsers access violations (SECVOLS). If VOLRULE is specified, then the psuedo dsn format will be VOLUME.@volser. If NOVOLRULE, is specified, then the format will be @volser.VOLUME. Note that @volser.VOLUME was the format used in releases prior to 4.1.

XBM/NOXBM

Specifies whether JES2 Express Batching Monitor jobs are to have their Logonids and passwords validated. If XBM is specified, validation will occur at beginning execution time. Default value is NOXBM.

ACF2 Sites Upgrading from 3.1.x Releases

If your site is currently running ACF2 Version 3.1.3 or below, the three ACF2 VSAM databases must be REPROed prior to specification of LIDRECL=(1024). See the "Install Procedure" chapter of the acf2/MVS System Programmer's Guide for instructions, and review the following notes.

Sharing the ACF2 Logonid Database Across Multiple CPUs

If the ACF2 Logonid database is shared by multiple CPUs, then the decision on whether to specify 512 or 1024 should be made carefully. For example, if CPUA uses 512-byte Logonid records and shares a Logonid database with CPUB which uses 1024-byte Logonid records, data integrity problems may be encountered on both CPUs. Several options are available to ensure compatibility:

1. Install ACF2 Release 4.1 on all systems simultaneously, specifying LIDRECL=(1024) for each system.

2. Specify LIDRECL=(512) on the target system when installing acf2/MVS Release 4.1 and convert later.

Installations Not Sharing Logonid Databases Across CPUs

Installations that do not share the ACF2 Logonid database across multiple CPUs can specify LIDRECL=(1024) and take immediate advantage of the larger Logonid record size.

First Time ACF2 Installations

Sites that have never installed ACF2 on any of their CPUs should specify LIDRECL=(1024). This is the default value.

Specification: Required.

SHOW Command: The SHOW STATE subcommand of the ACF command displays the values specified in the OPTS record for the BLPLOG, CENTRAL, CHANGE, CPUTIME, DATE, DECOMP, DFTLID, DFTSTC, INFOLIST, JOBCK, MAXVIO, MODE, $NOSORT, SAF, SAFTRACE, STC, TAPEDSN, VOLRULE and UADS fields.

The SHOW SYSTEM subcommand displays the values specified for the CONSOLE, LABEXP, LABNUM, LIDRECL, NOTIFY, SHRDASD, STAMPSMF, and XBM fields.

The SHOW ACF2 subcommand displays all of these values.

PPGM Record--Protected Program List

RECORD-ID	FIELDS
PPGM	PGM-MASK(pgm-mask1,pgm-mask2,...,pgm-maskn)

Function: Defines the programs that may be executed by privileged users. These programs can only be executed by unscoped users with the SECURITY privilege, or by Logonids with the non-cancellable (NON-CNCL) attribute. Refer to the section at the end of this chapter defining the relationship between the MAINT, PPGM, and LOGPGM GSO records.

Fields:

PGM-MASK(pgm-mask1,pgm-mask2,...,pgm-maskn)
One or more program-mask patterns, up to eight characters each.

Specification: Required. Up to 255 program-mask patterns can be specified.

Usage Notes: The ACF2 system is supplied with a PPGM record specifying IEHD**** (IBM's IEHDASDR), FDR*** (Innovation's Fast Dump/Restore), DRWD**** (IBM's Program Product), and ICKDSF** (also an IBM product). Programs can also be protected by moving them to a non-LINKLIST library and writing appropriate access rules.

FDR now has an ACF2-supplied interface. Refer to the Other Products Manual for further information.

SHOW Command: The SHOW PROGRAMS/PGMS and SHOW ACF2 subcommands of the ACF command display each program name specified in a PPGM record.

PSWD Record--Password Maintenance and Support

RECORD-ID	FIELDS

PSWD	ENCRYPT(R221/XDES)
	MAXTRY(1/nnn)
	MINPSWD(1/n)
	PASSLMT(2/nnn)
	PSWDALT/NOPSWDALT
	PSWDFRC/NOPSWDFRC
	PSWDJES/NOPSWDJES
	WRNDAYS(1/nnn)

Function: Defines the various Logonid password options and controls.

Fields:

ENCRYPT(R221/XDES)
Specifies which password encryption algorithm ACF2 will use to encrypt user passwords. R221 selects the algorithm used prior to Version 3.1.4 and is provided for compatibility only. XDES, the default, is the standard ACF2 encryption algorithm and is strongly recommended.

MAXTRY(1/nnn)
Specifies the maximum number of attempts, including the initial password entry, that are allowed before the terminal session is cancelled. Default value is 1. Maximum value is 255.

MINPSWD(1/n)
Specifies the minimum number of characters required in a new password. When ACF2 is first installed, MINPSWD should be set to 1 to allow conversion of the passwords currently in the UADS dataset. This minimum can be raised later and the old passwords will continue to be valid until they are changed or expire. Default value is 1. Maximum value is 8.

PASSLMT(2/nnn)
Specifies the maximum number of invalid password attempts to be allowed in a single day before ACF2 will deny all accesses to the system by the Logonid. The invalid password count can be reset to any number by the Security Officer, and may be reduced by one via the "F ACF2,RESET(logonid)" operator command. Default value is 2. Maximum value is 32767.

PSWDALT/NOPSWDALT

Specifies whether a new password may be entered by users at TSO logon time. The default is PSWDALT--allow password alteration. A user can also change his password by changing the PASSWORD field of his Logonid record by means of the ACF CHANGE command. To prevent such changes, the PASSWORD field can be redefined in the @CFDE macro of the ACF2 Field Definition record. See the acf2/MVS System Programmer's Guide for further details on defining the Logonid record.

PSWDFRC/NOPSWDFRC

Specifies whether or not a user will be forced to change the password at the next logon whenever someone other than the user, such as a Security Officer or Account Manager, changes the password. NOPSWDALT and PSWDFRC are conflicting and should not be used together. If PSWDFRC is set, ACF2 uses the PSWDALT option. Default is PSWDFRC.

PSWDJES/NOPSWDJES

Indicates if invalid passwords entered through batch jobs should be counted towards the invalid password limit (PASSLMT) in a single day (i.e., update PSWD-VIO count). The default value is NOPSWDJES.

WRNDAYS(1/nnn)

Specifies the number of days a warning is issued to the user prior to password expiration. This warning message is displayed on every TSO and batch system access. CICS and IMS users will not receive the warning message. Default value is 1.

Specification: Required.

Usage Notes: The ENCRYPT(R221) option should remain in effect until the installation is absolutely certain that acf2/MVS Release 4.0 or above is the production system with no possibility that a prior release of ACF2 will be reinstated as the production system.

When the ENCRYPT(R221) field is changed to ENCRYPT(XDES), individual passwords already encrypted using R221 will remain as such. The XDES encryption will not take place until the original passwords are changed. Changing this encryption technique in the PSWD record does not force the re-encryption of passwords.

Once a password has been encrypted and stored in a Logonid record through the XDES algorithm, that password will not be recognized under previous versions of ACF2. However, when the XDES algorithm is active, all passwords can be recognized regardless of which algorithm encrypted it. Encryption occurs when a user changes his password at logon time, or through a direct change to the PASSWORD field of the Logonid record.

Until XDES can be implemented on all CPUs simultaneously, multi-CPU sites that share the ACF2 databases should use the ENCRYPT(R221) option. In addition, the ACF2 JES2 and JES3 interfaces transmit partially encrypted passwords across NJE lines when the origin node uses the XDES password encryption algorithm. Consult the acf2/MVS System Programmer's Guide for complete information about the ACF2 JES2 and JES3 interfaces.

Sites that are installing ACF2 for the first time should specify the ENCRYPT(XDES) option. If this is a first-time install on a target system that interfaces with other ACF2-protected CPUs, then some coordination is required to maintain compatibility between CPUs.

SHOW Command: The password options that are in effect can be displayed with the SHOW STATE or SHOW ACF2 subcommands of the ACF command.

For installations migrating to Release 4.0 from Version 3.1.4, the XOUT17 field from the @PSWD macro has been placed in the TSOTWX record in Release 4.0.

RESDIR Record--Resident Resource Rule Directories

RECORD-ID	FIELDS
RESDIR	TYPES(c-type1,c-type2,...,c-typen)

Function: Determines whether generalized resource rules of a given type will be made resident.

Fields:

TYPES(c-type1,c-type2,...,c-typen)

Classes:

D-type
Resource rule sets are made resident in an address space on a demand basis.

R-type
Resource rule sets are made resident in global storage at ACF2 initialization time.

T-type
Resource rule sets are transient and never be made resident.

Usage Notes: Specifying (R-ITR) would cause all IMS transaction rules (i.e., resource rules with TYPE(ITR) specified) to be made globally resident along with the directory at ACF2 initialization time. Resident directories are rebuilt at each IPL, restart of ACF2, or when the console operator issues the F ACF2,REBUILD(type) command.

Regardless whether a given type of resource rule is resident on demand, resident in global storage, or transient, a directory is built for each type of resource rule specified in the RESDIR option and the directory itself is always made resident.

In order to use masks (asterisks or dashes within specifications) for resource names, the directory for the given type must be made resident. The maximum number of resource names allowed in a directory is 1024.

Specification: Required. A total of 256 generalized resource rule TYPES may be specified.

SHOW Command: The SHOW RESIDENT and SHOW ACF2 subcommands of the ACF command display the generalized resource types as specified in the RESDIR record.

RESRULE Record--Resident Rule Index List

RECORD-ID	FIELDS
RESRULE	INDEX(index1,index2,...,indexn)

Function: Defines a set of high-level indices identifying the access rule sets to be made resident in storage at ACF2 initialization time.

This function may be used to reduce the I/O operations required by ACF2 to obtain heavily used indices such as SYS1.

Fields:

INDEX(index1,index2,...,indexn)
Any number of RESRULE index fields may be used, up to 255 high-level indices.

Usage Notes: Once an index is made resident, changes to its rule set will not take effect until the next IPL, restart of ACF2, or until the console operator issues the F ACF2,RELOAD(index) command. It is therefore recommended that rules not be made resident until after rule modifications have stabilized.

Specification: Required. A total of 255 INDEX values can be supplied.

SHOW Command: The SHOW RESIDENT and SHOW ACF2 subcommands of the ACF command display the high-level index of each rule set specified in the RESRULE record.

RESVOLS Record--Dataset-Level Protection Volume List

RECORD-ID	FIELDS
RESVOLS	VOLMASK(mask1,mask2,...,maskn)

Function: Defines DASD and mass storage volumes for which ACF2 is to provide protection at the dataset-name level.

Fields:

VOLMASK(mask1,mask2,mask...,maskn)
Any number of volume serial masks up to six characters each. Standard ACF2 masking conventions are supported. By default, all DASD volumes are protected at the dataset-name level.

Specification: At least one VOLMASK is required. A total of 255 volume masks may be specified.

SHOW Command: The SHOW STATE and SHOW ACF2 subcommands of the ACF command display the volume serial number of all volumes specified in the RESVOLS record.

SAFMAPS Record--SAF Resource Rule Mappings

RECORD-ID	FIELDS
SAFMAPS	MAPS(SAF/-,type-1/c-1,...type-n/c-n)

Function: Associates various ACF2 user-defined generalized resource types and classes to the IBM standard group of SAF class names. The facility translates SAF requests into the appropriate generalized resource rule type prior to ACF2 rule validation. This record is active only when the SAF option is selected in the GSO OPTS record.

Fields:

MAPS(SAF/-,type-1/c-1...type-n/c-n)

This field can define as many as 128 different type and class combinations. SAF/- is the default. The default option provides the facility to group some or all resource rules under the type of SAF. Standard ACF2 masking conventions are supported for the CLASS entry.

The following example indicates that ACF2 will validate resource rules for the type TST for SAF requests under the class TCICSTRN. Resource type CKC rules will be validated for all other SAF classes (such as GCICSTRN). The standard ACF2 most specific to least specific search order is used.

```
     SAFMAPS MAPS(CKC/*CICSTRN, SAF/-, TST/TCICSTRN)
```

SAFSAFE Record--SAF Safelist

RECORD-ID	FIELDS

SAFSAFEqual	SUBSYS(subsystem-mask)
	CNTLPTS(point1,...,pointn)
	CLASSES(class1,...,classn)

Function: Defines the subsystem, control point, and class combinations for which the System Authorization Facility (SAF) will bypass validation. If more than one SAFSAFE record is required, a qualifier can be appended to the record name to generate a unique record id (e.g. SAFSAFE.OPENJ). The qualifier can be as long as 8 characters.

Fields:

SUBSYS(subsystem-mask)
Defines the subsystem for this SAFSAFE record. Standard ACF2 masking conventions are supported.

CNTLPTS(point1,...,pointn)
Defines up to 128 control points. Standard ACF2 masking conventions are supported.

CLASSES(class1,...,classn)
Defines up to 128 classes. Standard ACF2 masking conventions are supported.

Specification: Optional. For further information concerning ACF2 interaction with SAF, refer to the acf2/MVS System Programmer's Guide.

SECVOLS Record--Volume Mask Volume-Level Protection

RECORD-ID	FIELDS
SECVOLS	VOLMASK(volmask1,volmask2,...,volmaskn)

Function: Defines the DASD, mass storage, and tape volumes for which ACF2 is to provide volume-level protection.

Fields:

VOLMASK(volmask1,volmask2,...,volmaskn)
Any number of volume serials or masks, a maximum of six characters each. Standard ACF2 masking conventions are supported. The default of this record is null, no volume-level protection.

Specification: If ACF2 volume-level protection is to be used, then at least one volser is required. Additional VOLMASKS can be used, a total of 255.

Usage Notes: When a volume is defined in the SECVOLS record, ACF2 generates a pseudo dataset name of @volser.VOLUME for the purpose of validating an access request to that volume. However, if the VOLRULE field in the OPTS GSO record is specified, then the pseudo dataset name is generated as VOLUME.@volser.

SHOW Command: The SHOW STATE and SHOW ACF2 subcommands of the ACF command display the volume serial number of all volumes specified in the SECVOLS record.

TSO Record--Time Sharing Options and Defaults

RECORD-ID	FIELDS

TSO	ACCOUNT(1/string)
	BYPASS(#/character)
	CHAR(BS/character)
	CMDLIST(module-id)
	FSRETAIN/NOFSRETAIN
	LINE(ATTN/CTLX/character)
	LOGONCK/NOLOGONCK
	PERFORM(0/nnn)
	PROC(IKJACCNT/procedure)
	QLOGON/NOQLOGON
	REGION(512/nnnn)
	SUBCLSS(class)
	SUBHOLD(class)
	SUBMSGC(class)
	TIME(0/nnn)
	TSOSOUT(A/class)
	UNIT(SYSDA/unitname)
	WAITIME(0/nnn)

Function: Specifies global TSO usage and system parameters that define and control the TSO logon process and other system parameters.

Fields: (* = active only if UADS is bypassed.)

*ACCOUNT(1/string)
Specifies the system-wide default TSO Account Number. Default value is 1. If this string is set to blanks and there is no ACCOUNT specified in the Logonid record, an account prompt will occur at logon regardless of the UADS setting. This string is put in parentheses when it is moved to the JOB Card.

BYPASS(#/character)
Defines the TSO command list bypass character. Default value is a pound sign (#).

*CHAR(BS/character)
Defines the default TSO delete character. When entered at the terminal, this character indicates that the previous character entered should be ignored. This optional field has no default value. BS indicates that the backspace character will cause deletion of the previous character entered.

CMDLIST(module-id)

Specifies the default TSO command limiting list. If a module is specified, it will be impossible for anyone, even privileged Logonids, to run without the command list present in a LINKLIST library. This field is optional and has no default. It is effective in all modes with the exception of QUIET.

FSRETAIN/NOFSRETAIN

Controls the retention of logon values from session to session if TSO fullscreen logon is supported. NOFSRETAIN, the default, indicates that the user will have to provide applicable values at each logon.

*LINE(ATTN/CTLX/character)

Specifies the system-wide default TSO Line delete character. When entered at the terminal, this character indicates that the current line should be ignored. This optional field has no default value. ATTN indicates that an attention interruption will cause deletion of the current line. CTLX indicates that the X and CTRL keys pressed simultaneously cause deletion of the current line (for Teletype terminals).

LOGONCK/NOLOGONCK

Indicates if ACF2 is to check the TSO attribute in the user's Logonid record. If LOGONCK is specified and the TSO attribute is not on for that user, the logon attempt will be rejected. The default value is NOLOGONCK.

*PERFORM(0/nnn)

Specifies the system-wide default TSO performance group. If zero is specified, no performance group (PERFORM=) parameter will be placed on the job card. Default value is zero.

*PROC(IKJACCNT/procedure)

Specifies the default TSO cataloged procedure name. The default value for an individual user is specified through the TSOPROC field of the Logonid record.

QLOGON/NOQLOGON

Specifies whether a quick one-line logon is to be permitted. This allows ACF2 to accept the password specified on the first line instead of forcing a prompt. When QLOGON is in effect, password integrity can be jeopardized. Default value is QLOGON.

*REGION(512/nnnn)

Specifies the default TSO Region size. This value can be overridden by a TSOSIZE in the user's Logonid record, or by a size specification at TSO logon time. If this field is zero and no region size is specified at logon time or in the Logonid record, then ACF2 will assume that region has been specified in the TSO logon proc. Default value is 512.

*SUBCLSS(class)
Specifies the default TSO Job Submission Class. This parameter is only active if the TSO Command Package Program Product or TSO/E is also installed. This is an optional field and has no default value.

*SUBHOLD(class)
Specifies the default Submit Hold Class. This parameter is only active if the TSO Command Package Program Product or TSO/E is also installed. This is an optional field and has no default value.

*SUBMSGC(class)
Specifies the default Submit Message Class. This parameter is only active if the TSO Command Package Program Product or if TSO/E is also installed. Default value is null.

*TIME(0/nnn)
Specifies the default time estimate for TSO Sessions in minutes. If zero is specified, no TIME parameter will be placed on the job card. Default value is zero. Maximum value is 1439.

*TSOSOUT(A/class)
Specifies the default class for spun TSO Sysout. This parameter is only active if the TSO Command Package Program Product is also installed. Default value is A.

*UNIT(SYSDA/unit name)
Specifies the default UNITNAME to be used in TSO allocation requests. Default value is SYSDA.

WAITIME(0/nnn)
Specifies if ACF2 is to check that completion of TSO logons (responses to prompts) are within nnn seconds. The logon is aborted if waitime is exceeded. The value specified as nnn must be less than or equal to 120 seconds. Default value is 0 (no check takes place).

Specification: Required.

SHOW Command: The SHOW TSO and SHOW ACF2 subcommands of the ACF command display the TSO options as specified in the TSO record.

TSOCRT Record--ASCII CRT Clear String

RECORD-ID	FIELDS
TSOCRT	STRING(A12FA11C1A270C0D/hhhhhhhhhhhh...h)

Function: Defines a clear string used to obliterate the logon on ASCII CRT devices.

Fields:

STRING(A12FA11C1A270C0D/hhhhhhhh...h)
A 1 - 256 byte CRT clear string, in hexadecimal. The default is A12FA11C1A270C0D.

Specification: Optional. This record corresponds to and replaces both the XOUT17 option in the @PSWD ACFFDR macro and the NCP option in the @OPTS ACFFDR macro as they were used in acf2/MVS pre-Release 4.0 installations.

SHOW Command: This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

TSOKEYS Record--User Logon Keywords

RECORD-ID	FIELDS
TSOKEYS	KEYWORDS(keyword1,keyword2,...,keywordn)

Function: Defines installation-supplied keywords that ACF2 will allow at TSO logon time.

Fields:

KEYWORDS(keyword1,keyword2,...,keywordn)
Allows a total of 256 eight character keywords that the installation wants recognized by ACF2 as valid at TSO logon time.

Specification: Optional. A total of 256 keywords can be specified.

SHOW Command: The LIST subcommand of the ACF command will display the keywords from the TSOKEYS record.

TSOTWX Record--TWX X-Out String

RECORD-ID	FIELDS

TSOTWX	CR(15/hhhh)
	IDLE(17/nn)
	LENGTH(8/nn)
	M1(X/c)
	M2(N/c)
	M3(Z/c)
	M4(M/c)
	STRING(hhhhhhhhh...h)

Function: Defines an x-out mask used to obliterate the logon password on TWX devices.

Fields:

CR(15/hhhh)
The carriage return character, in hexadecimal. Acceptable values are 15, OD, or OD15. Default value is 15. For installations migrating to Release 4.1 from Version 3.1.4 or below, this field replaces the NCP operand in the pre-Release 4.0 @OPTS macro.

IDLE(17/nn)
The TWX idle character, in hexadecimal. Default value is 17.

LENGTH(8/nn)
The length of the x-out mask. Acceptable values are 8 or 17 bytes. Default value is 8. For installations migrating to Release 4.0 from Version 3.1.4, this field replaces the XOUT17 operand in the Version 3.1.4 @PSWD ACFFDR macro.

M1(X/c)
The first mask character. Default value is X.

M2(N/c)
The second mask character. Default value is N.

M3(Z/c)
The third mask character. Default value is Z.

M4(M/c)
The fourth mask character. Default value is M.

STRING(hhhhhhhhhh...h)

The 1 - 256 character x-out string in hexadecimal. The default is a null string, which causes the string to be built from values specified in the other fields of the TSOTWX record.

Specification: Optional.

SHOW Command: This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

TSO2741 Record--2741 X-Out Mask

RECORD-ID	FIELDS

TSO2741	BS(16/nn)
	LENGTH(8/nn)
	M1(X/c)
	M2(N/c)
	M3(Z/c)
	M4(M/c)
	STRING(nnnnnnnnn...n)

Function: Defines an x-out string used to obliterate the logon password on 2741 devices.

Fields:

BS(16/nn)
The backspace character. Default value is 16.

LENGTH((8/nn)
The length of the x-out mask. Acceptable values are 8 or 17. Default value is 8. For installations migrating to Release 4.0 from Version 3.1.4 or below, this operand replaces the XOUT17 operand in the @OPTS ACFFDR macro.

M1(X/c)
The first mask character. Default value is X.

M2(N/c)
The second mask character. Default value is N.

M3(Z/c)
The third mask character. Default value is Z.

M4(M/c)
The fourth mask character. Default value is M.

STRING(hhhhhhhhh...h)
The 1 - 256 character x-out string in hexadecimal. The default is a null string, which causes the string to be built from values specified in the other fields of the TSO2741 record.

Specification: Optional.

SHOW Command: This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

WARN Record--System WARN Mode Message

RECORD-ID	FIELDS
WARN	MSG(msg-text)

Function: Specifies text of a warning message to be displayed on the terminal and/or job log when a violation has taken place and the ACF2 system is in WARN mode.

Fields:

MSG(msg-text)
Any text string, up to 124 characters, enclosed in single quotes. The default warning string is:

AFTER JULY 1, 1999 THIS ACCESS WILL NOT BE ALLOWED

Note: The disposition of the WARN message is dependent upon the CONSOLE parameter of the OPTS record.

Specification: Required. One per system.

SHOW Command: This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

RELATIONSHIP BETWEEN LOGPGM, MAINT AND PPGM

The installation can define which programs must be executed by privileged Logonids using the MAINT, LOGPGM or PPGM records in GSO. The programs on these records must be carefully selected to avoid the problem of excessive loggings or the absence of adequate audit trails. Programs listed on the LOGPGM record generate a logging each time the program accesses a dataset, regardless of any trace or violation. The programs on this record are secured by normal access rules and they are generally not run very frequently. The PPGM and MAINT records can list the names of programs which must be run by privileged users. The programs listed on these two records are generally executed frequently, making it desirable to partially or totally suppress loggings.

This diagram shows relationships between the LOGPGM, MAINT and PPGM GSO records.

	Logonid Attributes	Access Rule Validation	SMF Loggings
LOGPGM	N/A	YES	All Dataset Accesses
*MAINT	MAINT or NON-CNCL	BYPASSED	NONE
PPGM	Unscoped SECURITY or NON-CNCL	BYPASSED	Only at Step Initiation

* The MAINT record requires that the program, the library in which it resides, and the Logonid that will execute it all be included on this record.

CONTROL SETTING: TSO FULLSCREEN LOGON RETENTION RECORDS

The Infostorage TSO type records associated with the CONTROL class are maintained internally by the ACF2 TSO Preprompt exit (IKJEFLD). They retain the logon parameters specified during the most recent TSO logon. When the fullscreen retention facility is active, ACF2 will assume that the logon parameters desired are those specified in the user's Logonid record. When those parameters differ from the Logonid record, a retention record is created in Infostorage.

Use of the logon parameter retention facility requires that:

1. Fullscreen TSO logon is selected in the individual Logonid record. The TSOFSCRN attribute introduced in Release 4.0 selects the fullscreen logon.

2. The FSRETAIN attribute is specified in the TSO Global System Options (GSO) record.

By default, the logon parameter retention facility is dormant unless explicitly activated by the installation as described above. Once activated, management of the supporting CONTROL(TSO) records is automatic and requires no administrative attention. However, the retention records may be listed by a security officer or INFOLIST privileged user.

Because the management of retention records is performed internally, no external facility is provided to update the fields contained in these records. ACF command support for CONTROL(TSO) records is provided primarily to furnish the administrator with a mechanism to review the usage of the retention facility. In fact, when using the SET command, SYSID does not need to be stated. Additionally, if the installation discontinues the use of the retention facility for any reason, the ACF command can be used to delete any or all of the unused retention records.

When a Logon Retention Record is created, the user's Logonid becomes the name of that user's retention record identifier. The fields contained in the user's record are described below. When listed, only those parameters which differ from the defaults found in the Logonid record are displayed.

ACCOUNT - Identifies the TSO logon account string. This field corresponds to the TSOACCT field of the Logonid record.

ATTR1 - Flags defining the status of the retention record. These flags are for internal use only and have no counterpart in the Logonid record.

ATTR2 - Flags defining the status of the retention record. These flags are for internal use only and have no counterpart in the Logonid record.

ATTR3 - Flags defining the status of the retention record. These flags are for internal use only and have no counterpart in the Logonid record.

COMMAND - When the command string feature is activated by the installation system programmer, this field contains a TSO command issued immediately after logon has completed. This field has no counterpart in the Logonid record.

MSGCLASS - Identifies the SYSOUT class to which the TSO session is assigned. This field has no counterpart in the Logonid record.

PERFORM - Identifies the MVS performance group the TSO session will run under. This field corresponds to the TSOPERF field of the Logonid record.

PROC - Identifies the TSO JCL procedure name. This field corresponds to the TSOPROC field of the Logonid record.

REGION - Identifies the MVS region size in Kbytes. This field corresponds to the TSORGN field of the Logonid record.

TIME - Identifies the CPU time limit associated with the TSO session. This field corresponds to the TSOTIME field of the Logonid record.

UADSNDX - When running ACF2 using the UADS option, UADSNDX identifies the SYS1.UADS index (password) which determines the user's logon procedure and account string. This field corresponds to the UADSINDX field of the Logonid record.

UNIT - Identifies the default MVS unit name selected during dataset allocations. This field corresponds to the TSOUNIT field of the Logonid record.

SYSID CONCEPTS

The SYSID is a string of 1 to 8 characters that is used to group GSO records. The content of the SYSID string is arbitrary and may be defined by the installation.

When ACF2 is started, the SYSID string selected will remain in effect until explicitly changed by the operator. The initial SYSID selection logic is as follows:

1. The initial SYSID string is extracted from the SYSID(sysid) operand of the START command PARM field. For example, if the system is started with the command S ACF2,PARM='SYSID(PROD)', then PROD is selected.

2. If the SYSID operand is omitted from the START command, the SMF system ID value defined at IPL time via SYS1.PARMLIB(SMFPRMxx) is selected.

3. At any time after ACF2 startup, the operator may change the SYSID string using the SETSYSID(sysid) operand of the MVS MODIFY ACF2 command. For example, if the operator enters F ACF2,SETSYSID(TEST), the SYSID string is changed to TEST.

Under the CONTROL (GSO) setting of the ACF command a SYSID string is selected for the session according to the following logic:

1. After the user establishes the CONTROL setting for the ACF command, the current ACF2 SYSID string (obtained originally as described above) is selected as the default SYSID for the session.

2. The SYSID and MSYSID parameters of the ACF SET subcommand may be used at any time to change the session default. MSYSID is similar to SYSID, but indicates the use of SYSID masking. For example, the system may be started with a SYSID string of CPU1, but the user began his ACF command session by entering:

   ```
   SET CONTROL(GSO) SYSID(CPU2)
   ```

   As a result of this subcommand, the default SYSID becomes CPU2 for the duration of the CONTROL (GSO) setting.

3. Finally, the SYSID or MSYSID parameter can be specified with any of the ACF subcommands under the CONTROL setting (INSERT, CHANGE, LIST, and DELETE). These parameters override the SYSID default for the execution of that command only. For example, assume the default SYSID is CPU1 when the following subcommand is entered:

   ```
   CHANGE PSWD MINPSWD(5) SYSID(CPU2)
   ```

   The MINPSWD field change applies to the PSWD GSO record defined for CPU2, not for the GSO record defined for the system CPU1.

MULTIPLE SYSTEM SYSID USAGE

Installations using multiple systems can secure each system exclusively by using a different SYSID name for each system. In turn a different set of GSO records may then be created for each SYSID. When symmetry is desired, one or more GSO records can be shared between systems.

Naming conventions for the SYSID string are installation defined. For example, to start ACF2 for different CPUs, the MVS start command may look like the following:

```
     S ACF2,PARM='SYSID(CPU1)'
     S ACF2,PARM='SYSID(CPU2)'
```

To share a GSO record between multiple CPUs, masking characters can be used in the SYSID field. For example, to share the OPTS record between CPU1 and CPU2, the security officer sets the SYSID to the CPU-id which is controlled by the desired OPTS record. The GSO OPTS records for CPU1 and CPU2 must first be deleted using the standard ACF subcommand. Then a new GSO OPTS record can be created with a SYSID containing masked characters.

```
ACF
SET CONTROL(GSO) SYSID(CPU1)
CHANGE OPTS SYSID(CPU*)
```

An asterisk is substituted for the last character. Since the asterisk is a standard ACF2 mask, the OPTS record will apply to all CPUs with the SYSID of CPU followed by any alphanumeric character.

Similar concepts can be applied to qualified GSO records (e.g. MAINTqual, BLPPGMqual, SAFSAFEqual, etc.). For example, an installation has 2 SYSIDs, CPU1 and CPU2. CPU1 has three MAINT records, MAINT.C1A, MAINT.C1B, and MAINT.CU3. CPU2 also has three MAINT records, MAINT.C2A, MAINT.C2B, and MAINT.CU3. The two systems also share several MAINT records, MAINT.CU1, MAINT.CU2, and MAINT.CU3. These three records would then be created with a SYSID(CPU*) and qualified as shown.

MSYSID USAGE

Installations utilizing multiple SYSIDs to secure their systems can make a change to one GSO record for multiple SYSIDs. This is performed using the standard ACF2 masking technique and a special parameter, MSYSID, with the CHANGE subcommand of the ACF command system.

```
ACF
SET CONTROL(GSO) SYSID(CPU1)
CHANGE OPTS MSYSID(CPU*) DATE(MDY)
```

Although the SYSID shown above is CPU1, the MSYSID parameter applies the change to the OPTS records for all SYSIDs beginning with CPU and ending with any alphanumeric character. This parameter can also be applied to the LIST subcommand. To LIST the OPTS records for all SYSIDs, the following command can be entered:

```
ACF
Set control(gso)
LIST MSYSID(-) OPTS
```

To LIST all MAINT records for all SYSIDs, the following command can be used:

```
ACF
Set control(GSO) sysid(CPU1)
LIST MSYSID(-) LIKE(MAINT-)
```

Since MAINT records are qualified, the specific name with the qualifier is entered or the LIKE parameter is used with a mask.

Use of '?'

When a '?' is used as a sysid value, it indicates that the default sysid value for this system is desired. The default system sysid is the SYSID selected when the system is initialized. Either of the ACF2 MVS commands in the right-hand column can be used to set the default system sysid.

```
S ACF2,PARM='SYSID(sysid)'
F ACF2,SETSYS(sysid)
```

For example, the system is started with a SYSID of PROD. A security officer enters the GSO environment, using a SYSID of TEST. Commands entered at this point would reference records with a SYSID of TEST.

```
S ACF2,PARM='SYSID(PROD)'
set control(gso) sysid(test)
```

However, the security officer remembers that he has to change a record used in the production, not test, environment. He uses a SYSID of '?' to reference the system default, in this case PROD.

```
change sysid(?) opts maxvio(5)
```

GSO-RELATED CONSOLE OPERATOR COMMANDS

Several console operator commands process GSO records and related GSO facilities. More information about these commands can be found further on in this manual.

F ACF2,REFRESH(recid/ALL)
Dynamically applies changes made to GSO records without a system IPL. ACF2 automatically applies any change made to GSO records whenever the system is IPLed. Also, if you stop and then start ACF2 with a new SYSID, a total REFRESH occurs.

F ACF2,SHOWSYS
Displays at the console current SYSID and time the SYSID was last changed.

F ACF2,SETSYS(sysid)

Establishes the current SYSID. The Global System Options of the new SYSID will not take effect until a REFRESH takes place.

F ACF2,TRACEGSO(SMF/CONSOLE/OFF/SYSLOG/SECURITY)

This sets the trace facility 'or GSO records. It is used in conjunction with the SHOW GSO console command and generally serves as a debugging tool. Each of the options specifies the type of GSO trace records that will be generated. A SMF record is generated regardless of the option selected when the trace is on.

F ACF2,SHOWGSO

When entered on the console, displays the effective TRACEGSO option. The actual loggings will display on the appropriate console if entered in the TRACEGSO command.

IDENTITY SETTING: STRUCTURED EXTENDED USER AUTHENTICATION

Identity records are of a single identifier, AUT. AUT records contain the installation's extended authentication information for each specified user. Since these records contain data that is unique for each system user, the Logonid is used as part of the key to identify each record.

A maximum of eight different applications can exist in an installation, the authentication type represents the DIVISION. The concept of DIVISION is similar to SYSID. This allows a user to have information on the Infostorage database for several authentication types.

Formats for each division of AUT records are provided in the APPLDEF GSO record. The requirement for the AUT record is established by the application and defined in the AUTHEXIT GSO record. If the INFOSTG option is not selected, the AUT records for that application do not need to be created. Further information about these records can be found in the GSO section of this manual. Installations utilizing OID card processing can convert their system to this. Refer to the Utilities Manual for complete information about the conversion.

Topics presented in this chapter are:

1. AUT Record Examples

2. ACF Subcommands for the IDENTITY Setting

AUT RECORD EXAMPLES

An example of the subcommands used to establish the IDENTITY setting to list a record for a user of a specific device type (OID card reader) is shown below. The fields listed below must be pre-defined to ACF2 by the installation. Therefore, the example assumes OID card authentication records have an OIDCARD field defined without data encryption.

```
acf
  ACF
set identity(aut)  division(oid)
  IDENTITY
show mode
  MODE:  IDENTITY  TYPE: AUT  DIVISION: OID
  IDENTITY
list user1
  OID  USER1  LAST CHANGED BY USER5 ON 5/25/85-12:52
              OIDCARD(12345678)
  IDENTITY
end
```

ACF COMMANDS FOR THE IDENTITY RECORDS

User authentication interface options are defined both via the CONTROL(GSO) setting and via changes made to the ACFFDR. Records containing the data the authentication application needs for verification are created via the IDENTITY setting.

The ACF command and subcommands follow the format of the CONTROL setting commands. To differentiate between groups of AUT records, a DIVISION is entered with either the SET subcommand or one of the subcommands (INSERT, CHANGE, LIST).

DIVISION follows a similar format as SYSID. When the IDENTITY setting is used, it groups the data for a specific authentication application. The APPLDIV field on the APPLDEF GSO RECID contains the appropriate DIVISION to be used in the IDENTITY setting for your installation. The same conventions used for SYSID in the CONTROL setting can be used for the IDENTITY setting.

The examples in this section only show possible entries for AUT records. The field names will differ depending upon the authentication applications established at your installation. Usage of the subcommands to add, change, or delete records is essentially the same as those used for CONTROL (GSO) records.

ACF SUBCOMMANDS UNDER THE IDENTITY SETTING

You can process AUT records after establishing the IDENTITY setting of the ACF commands and specifying the type code AUT:

```
     set identity(AUT)
```

After establishing the IDENTITY setting, you can issue any of the following subcommands:

```
    *INSERT      *DELETE      *SET
    *CHANGE       END         *SHOW
    *LIST         HELP         SN
```

The common subcommands END, HELP, and SN operate under the IDENTITY setting as explained in the chapter on the ACF command. The other subcommands, marked by asterisks(*), are explained in the following text.

INSERT Subcommand--IDENTITY(AUT) Setting

The INSERT subcommand under the IDENTITY(AUT) setting allows you to create an AUT record for a specific Logonid.

Syntax. The INSERT subcommand has the following syntax:

```
     INSERT [USING(recid)] [MDIV(?/appldiv)] [DIVISION(?/appldiv)] */recid
            [field,field,...][ADD/REP/DEL]
```

Example.

For the sake of this illustration, assume that the installation has an extended authentication routine which requires specific users to enter a second password after normal TSO logon. The installation has chosen to name their DIVISION AUTRTN1 representing their first authentication application. Both the AUTRTN1 and the routine's record structure block are named in the GSO APPLDEF control record. The AUTHEXIT GSO record contains the Logonid attribute name and associated processing routine for this application. Since the routine will use records for each user, the ACF2 Infostorage Database AUT records can be created for each user as follows:

```
acf
set identity(aut)  division(AUTRTN1)
  insert user1 password (-)
  USER1  CREATED BY USER2 ON 08/28/85-15:32
         PASSWORD ( )
```

This example of the INSERT subcommand shows:

1. An AUT RECID of USER1 has been created for the installation-defined AUTRTN1 division.

2. The USER1 Logonid will be required to enter a second password after normal TSO Logon via the AUTRTN1 routine. The routine chosen by the installation will store the PASSWORD field entry in encrypted format on the ACF2 Infostorage Database.

3. DIVISION can be entered at the SET subcommand or on the same line as the INSERT subcommand of the ACF command.

Parameters. The INSERT subcommand supports the following parameters:

USING(recid)
(Optional) Identifies a model Logonid record to be copied in creating the new AUT record. All values from the model record are inserted in to the new record. Any other fields and values specified in the INSERT subcommand add to or replace the fields and values in the new record.

DIVISION(?/appldiv)

Specifies the 1 to 8 character DIVISION for which this inserted record applies.

The question mark (?) indicates that the division default value for this system should be used. If this operand is not specified, the DIVISION(appldiv) value specified when the IDENTITY setting was established will be used. This field is defined on the APPLDIV field in the APPLDEF GSO record. DIV is an acceptable abbreviation for the DIVISION parameter name.

MDIV(division-mask)

Specifies a mask indicating the division for which the specified IDENTITY record will be changed. This must contain at least one asterisk or a trailing dash. AUT records changed through this parameter will have the same record identifier (RECID), but different DIVISION. MDIV is an acceptable abbreviation for the MDIVISION parameter name.

*/recid

Specifies the identifier of the record(s) to be changed. An asterisk (*) will refer to the last record processed since the IDENTITY setting was established. (The asterisk will not work in the case of multiple records or masking.)

field,field,...

Specify the fields and any values to be added to the new record (or to replace fields copied from a model record). For the specific field names in each record, see the previous description of the APPLDEF records and their respective field names.

These general rules apply to field names:

1. Bit fields are turned on by stating the field name, and turned off by prefixing the field name with 'NO'.

2. Fields with variable or character values must be followed immediately by the desired value in parentheses. For example, TIME(12:30).

3. Fields which are defined with the capacity to contain multiple values must be followed by those values in parentheses. For example: PASSWORD(123 132 145). Either a space or comma is a valid delimiter within the parentheses.

ADD

Indicates that the specified field values will be added to those copied from the model record. The new value will be added and any existing values will remain. ADD is the default value.

REP
: Indicates that any field value specified will be replaced the value of that field as indicated.

DEL
: Indicates that the specified field value will be deleted from the record.

CHANGE Subcommand--IDENTITY(AUT) Setting

The CHANGE subcommand, under the IDENTITY(AUT) setting, allows you to change an existing AUT record.

Syntax. The CHANGE subcommand has the following syntax:

```
CHANGE [DIVISION(?/appldiv)/MDIV(division-mask)] */recid [ADD/REP/DEL]
       [field,field,...]
```

Example. The following CHANGE subcommand alters the user's authentication record by adding more DEVICE information.

```
acf
set identity(aut)  division(appldiv1)
change user1 device(34567890)
```

ADD is the default of this command.

Parameters. The CHANGE subcommand takes the following parameters:

DIVISION(?/appldiv)
: Specifies the division-id of the IDENTITY record to be changed. If ? is specified as the DIVISION value, then the previously established setting for the session will be used. DIV is an acceptable abbreviation for DIVISION.

  Any value specified with the DIVISION parameter may contain asterisks or a trailing dash; however, these characters will be treated as part of the DIVISION itself and not as an indication of masking.

  If the DIVISION parameter is specified, then the DIVISION parameter cannot be specified.

MDIV(division-mask)
: Specifies a mask indicating the division for which the specified IDENTITY record will be changed. This must contain at least one asterisk or a trailing dash. AUT records changed through this parameter will have the same record identifier (RECID), but different DIVISION. MDIV is an acceptable abbreviation for the MDIVISION parameter name.

*/recid
Specifies the identifier of the record(s) to be changed. An asterisk (*) will refer to the last record processed since the IDENTITY setting was established. (The asterisk will not work in the case of multiple records or masking.)

The following parameters apply only to multi-value fields in an AUT record.

ADD
Indicates that any fields and values specified with this CHANGE subcommand will be added to the existing field values in the specified AUT record(s). The specified values will be added to any existing values in the field. If the ADD, REP, or DEL parameter is not specified, then the ADD parameter is assumed to be the default.

REP
Indicates that any fields specified with this CHANGE subcommand will completely replace the corresponding field in the specifed IDENTITY record(s).

DEL
Indicates that any field value specified with this CHANGE subcommand will be deleted from the specified IDENTITY record(s). If any specified value does not exist in a field, then the field will remain unchanged.

field,field,...
Specify the field value to be changed as directed by the ADD, REP, or DEL parameter. For the specific field names in each record, see the previous description of the APPLDEF record and its fields.

LIST Subcommand--IDENTITY(AUT) Setting

The LIST subcommand, under the IDENTITY setting, allows you to list the contents of AUT record(s).

Syntax. The LIST subcommand has the following syntax:

```
LIST [DIVISION(?/appldiv)/MDIV(division-mask)] */recid/LIKE(recid-mask)
```

Parameters. The LIST subcommand takes the following parameters:

DIVISION(?/appldiv)
Specifies the DIVISION of the AUT record to be listed. If ? is specified as the DIVISION value, then the default value for the system will be used. DIV is an acceptable abbreviation for DIVISION.

Any value specified with the DIVISION parameter may contain asterisks or a trailing dash; however, these characters will be treated as part of the DIVISION itself and *not* as an indication of masking.

If the DIVISION parameter is specified, then the MDIV parameter cannot be specified.

MDIV(division-mask)
Specifies a mask indicating the divisions for which the specified IDENTITY record will be listed. This division-mask must contain at least one asterisk or a trailing dash. AUT records listed via this parameter will have the same record identifier(RECID), but different divisions. MDIV is an acceptable abbreviation for the MDIVISION parameter name.

*
Specifies the DIVISION and identifier of the last AUT record processed since the IDENTITY setting was established. The asterisk is not effective if multiple records were last specified by masking.

recid
Specifies a 1 to 8 character identifier of the AUT record to be listed.

LIKE(recid-mask)
Specifies a mask for the identifiers of the AUT records to be listed. This masking follows the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting).

DELETE Subcommand--IDENTITY(AUT) Setting

The DELETE subcommand, under the IDENTITY setting, allows you to delete the specified AUT record(s).

Syntax. The DELETE subcommand has the following syntax:

```
DELETE [DIVISION(?/appldiv)/MDIV(division-mask)]
       */recid/LIKE(recid-mask)
```

Parameters. The DELETE subcommand takes the following parameters:

DIVISION(?/appldiv)
Specifies the DIVISION of the AUT record to be deleted. If ? is specified as the DIVISION value, then the previously established setting for the session will be used. DIV is an acceptable abbreviation for DIVISION.

Any value specified with the DIVISION parameter may contain asterisks or a trailing dash; however, these characters will be treated as part of the division itself and not as an indication of masking.

If the DIVISION parameter is specified, then the MDIV parameter cannot be specified.

MDIV(division-mask)
Specifies a mask indicating the DIVISIONs for which the specified AUT record will be deleted. This sysid-mask must contain at least one asterisk or a trailing dash. Use extreme care when deleting multiple AUT records with this parameter. MDIV is an acceptable abbreviation for the MDIVISION parameter name.

*
Specifies the division-id and RECID of the last AUT record processed since the IDENTITY setting was established. The asterisk is not effective if multiple records were last specified by masking.

recid
Specifies a 1 to 8 character identifier of the AUT record to be listed.

LIKE(recid-mask)
Specifies a mask for the identifiers of the AUT records to be deleted. This masking follows the same conventions that apply to Logonids, as explained in the chapter on Logonid records (LID setting). Use extreme care when deleting multiple AUT records with this parameter.

SET Subcommand--IDENTITY(AUT) Setting

The SET subcommand allows you to establish the IDENTITY(AUT) setting for the ACF command. It also allows you to establish the active DIVISION plus other parameters that affect how certain ACF subcommands operate.

Syntax. The SET subcommand has the following syntax:

```
     set [division(appldiv)/mdiv(division-mask)]
         identity(aut)
```

Parameters. Only the DIVISION and MDIV parameters are unique to the IDENTITY(AUT) setting, and are explained below. The other parameters operate as explained in the chapter on the ACF command.

DIVISION(appldiv)
Specifies the DIVISION of the AUT records to be processed under the IDENTITY(AUT) setting. This DIVISION will remain active for the duration of the IDENTITY(AUT) setting but can be overriden through the DIVISION parameter of individual ACF subcommands.

If the DIVISION parameter is not specified when the IDENTITY(AUT) setting is established, then the system default is used. The DIVISION is established in the creation of the APPLDIV field of the APPLDEF GSO record.

DIV is an acceptable abbreviation for the parameter name DIVISION.

MDIV(division-mask)

Specifies a mask indicating the DIVISIONS of the AUT records to be processed under the IDENTITY(AUT) setting. These division-masks will remain active for the duration of the IDENTITY(AUT) setting but can be overriden through the DIVISION parameter of individual ACF subcommands.

MDIV is an acceptable abbreviation for the parameter name MDIVISION.

UTILITIES: REPORTS, BATCH PROGRAMS, AND TSO COMMANDS

The ACF2 system package provides various report generators, batch programs, and TSO commands, which are described in detail in the acf2/MVS Utilities Manual. The following sections of this chapter provide a brief description of these utilities for the user's general information:

REPORT GENERATORS

ACFRPTCR csk This utility formats the statistical TSO information collected by ACF2 interface routines. CPU time, service units, and various other resources utilized by each TSO command are presented in the output report of ACFRPTCR.

ACFRPTDS

ACFRPTDS produces the Dataset/Program Event Log. SMF records which journal dataset loggings, dataset access violations, dataset access trace requests, and program use loggings/violations are formatted and edited by the ACFRPTDS utility. The information is selected and sorted to produce various reports. Field descriptions and sample reports are available in the acf2/MVS Utilities Manual.

ACFRPTEL

The report generator ACFRPTEL uses SMF records to provide the Information Storage Update Log. This log contains a listing of each change made to the ACF2 Information Storage database, including updates to entry records, Global System Option (GSO) records, and generalized resource rule sets.

ACFRPTIX

The Dataset Access Index Report is produced by the ACFRPTIX utility. This report is of particular aid to the security officer or auditor in that it helps to determine when the access environment for a particular dataset prefix has changed.

ACFRPTJL

The Restricted Logonid Job Log produced by this utility lists all system accesses made by Logonids with the RESTRICT attribute. A RESTRICTed Logonid does not require a password, and is intended for production job use. In order to carefully monitor the use of such Logonids, ACFRPTJL provides a summary of their system accesses.

ACFRPTLL

An update activity report of the ACF2 Logonid database - The Logonid Modification Log - is the output of this utility. It identifies each addition, change, or deletion of any Logonid record.

ACFRPTNV

This report generator produces the ACF2 Environment Report, which shows the date and time of each START (S ACF2), STOP (P ACF2), and MODIFY (F ACF2) operator command. The report also shows the date and time of each system IPL and possible losses of SMF data.

ACFRPTPP

This pre-processor utility selects SMF records that will be used by the various other ACF2 utilities in order to reduce the number of times the same records must be read. Generally, its purpose is one of cost reduction in that it cuts the overhead of report generating. It also provides various general statistics. This utility accepts any input file whose DDNAME begins with 'REC', thus allowing as many separate input files as needed.

ACFRPTPW

This utility provides the security officer with a list of each unsuccessful attempt to gain access to the system, and the reason the access was not allowed. Accesses made by use of the LOGSHIFT privilege are also journalled. The report generated is called The Invalid Password/Authority Log.

ACFRPTRL

Updates to the access rule database are reported by this utility on the Ruleid Modification Log. This report identifies who is changing which rules and is also useful in the event of recovery cross-checking.

ACFRPTRV

Using SMF records issued by the generalized resource facility, this utility produces a report describing the resource accesses, the user requesting the access, and the final disposition of the access. Loggings, violations, and trace requests are the types of resource events categorized in The Generalized Resource Event Log, produced by the ACFRPTRV report generator.

ACFRPTSL

The JCL parameters of this utility allow specified Logonid records to be selected for printed output. A full display of each record may be obtained, or a shortened version of the record can be specified. The Selected Logonid List is the report produced.

ACFRPTXR

For a specified dataset, volume, or resource name, ACFRPTXR produces a list of applicable rules and the Logonids of the users who have access to the data/resource (this access is determined either by the rules, or by special ACF2 authorization). In other words, a Cross-Reference Report is created to determine which users, based on the standard ACF2 security controls, will have access to the specific dataset, volume, or generalized resource.

ACFRPTRX

Similar to the cross-reference report described above, ACFRPTRX produces a report in Logonid order, linking users with associated access and generalized resource rules.

OTHER BATCH PROGRAMS

ACFBATCH

The ACFBATCH utility allows for execution, in a batch environment, of a sequence of ACF subcommands.

ACFBCOMP

The ACFBCOMP utility compiles access or generalized resource rule sets. It accepts rule entries specified in the jobstream itself or residing in a file. It accepts the same parameters (e.g., STORE, FORCE) as does the ACF COMPILE subcommand.

ACFBDCMP

The ACFBDCMP utility decompiles the ACF2 access or generalized resource rule set specified as a parameter. Only one rule set can be decompiled per execution of this utility.

ACFBSYNC

The ACFBSYNC utility synchronizes the Logonid database with the BRODCAST dataset. Its effect is similar to that of the ACF SYNCH subcommand.

ACFNRULE

The ACFNRULE utility adds a rule entry, and then compiles the access or generalized resource rule set specifed by the parameters. It can also delete those entries within a rule set that contain a certain character string. It can be executed both as a batch program and as a TSO command.

ACFERASE

The Data Disposal Utility, ACFERASE, will remove data from a direct access dataset and/or erase a tape volume. ACFERASE may be executed in batch or as the ACFDEL command under TSO.

ACFRECVR

The ACFRECVR utility processes SMF update records for the ACF2 databases. It merges those records to build VSAM clusters and, thus, restore the ACF2 databases.

JOBCOPY

The JOBCOPY utility allows for submission of production and other special types of job streams that need to run under a Logonid other than that of the TSO operator. This utility verifies that the user submitting the job stream has the authority to submit job streams via JOBCOPY from the referenced JCL library. It performs the same function within a batch environment as does ACFSUB in the TSO environment.

TSO COMMANDS

ACFCOMP

The ACFCOMP utility compiles access or generalized resource rule sets. It accepts rule entries directly from the terminal or from a file. It accepts the same parameters (e.g., STORE, FORCE) as does the COMPILE subcommand. ACFCOMP can also be used in batch under the Terminal Monitor Program (IKJEFT01).

ACFDEL

The ACFDEL utility (the Data Disposal Utility) will remove data from a direct access dataset and/or erase a tape volume. ACFDEL may be executed under TSO or via the ACFERASE batch utility.

ACFNRULE

The ACFNRULE utility adds a rule entry, and then compiles the access or generalized resource rule set specified by the parameters. It can also delete those entries within a rule set that contain a certain character string. It can be executed both as a batch program and as a TSO command.

ACFSUB

ACFSUB is a TSO command issued from TSO ready mode. This utility may be used to submit controlled production-type and other special job streams under a Logonid other than the one of the TSO operator. ACFSUB verifies that the TSO operator submitting the job has the authority to submit job streams via ACFSUB out of the referenced JCL library. In addition, this utility dynamically creates the Logonid that the job will run under. ACFSUB is similar to the JOBCOPY utility for batch production job submissions.

CONSOLE OPERATOR COMMANDS

The various console commands applicable to the ACF2 system are described below.

F ACF2,BACKUP

This command is used to manually initiate the backup of ACF2 control databases from any ACF2 CPU console regardless of the NOBACKUP parameter on the MVS startup command for ACF2. Even if the automatic backup process is in effect, you might use this command to initiate a backup at a point when a near-disaster situation was evident, the system appeared likely to go down, etc.

F ACF2,NEWSHIFT

This command is used to reload the resident matrix tables containing SHIFT information. ACF2 will also automatically reload these tables every twenty-four hours (at 12:00 A.M. - midnight).

F ACF2,NEWXREF

The new cross-reference command causes an input source cross-reference table to be rebuilt. This table is used to identify input sources and device names, and to cross-reference these to their input source groups. If new input sources or new input source names have been specified, or if you have reorganized input source groups, this command should be entered to ensure that the most current version of these tables are resident. Further information about source entries can be found in the chapter about source records.

F ACF2,REBUILD(type-code)

The REBUILD command recreates the directory of resident generalized resource rule sets of the specified type code. The type-code entry refers to the generalized resource type. ACF2 normally rebuilds this directory during each system IPL; this operator command recreates the directory in between IPLs. If an installation changes any masked or unmasked generalized resource rule set of the specified type-code, or if the installation changes the list of resident generalized resource rule sets, this command should be used to rebuild the directory.

The RESDIR record of the Infostorage database allows the installation to specify the list of resident generalized resource rule sets. For further information on resident rules and the RESDIR record, refer to the chapter on GSO records.

F ACF2,REFRESH(recid/ALL)

The REFRESH command should be executed whenever changes are made to CONTROL records in the Infostorage database and need to be applied to the system without a system IPL. ACF2 automatically applies the current CONTROL records whenever the system is initialized. To use this command, a console operator must have the REFRESH attribute specified in his Logonid record. ACF2 will prompt that operator for his Logonid and password.

Entries for the RECID field can be any valid GSO record. If ALL is specified, a complete refresh of all GSO records takes place.

A REFRESH of the RESDIR or RESRULE records does not affect the running system. In the event of a system crash, a discrepancy between the current directory or rules may be created if the REFRESH command was used to update the existing ones, we recommend that the RELOAD or REBUILD commands be used instead of the REFRESH for the RESDIR or RESRULE GSO record types.

F ACF2,RELOAD(ruleid)

The RELOAD command reloads the resident dataset access rule sets. ACF2 reloads these rule sets during each system IPL; this operator command reloads these rule sets between IPLs. If an installation changes any of the resident access rule sets or changes the list of resident access rule sets, this command should be used to reload the updated rule set. Valid entries for the RULEID field can be the high-level index of the dataset access rule.

The RESRULE GSO record of the Infostorage database allows the installation to specify the list of resident generalized resource rule sets. For further information on resident rules and the RESRULE record, refer to the chapter on GSO records.

F ACF2,RESET(logonid)

The RESET command reduces the password violation count field by one for the Logonid specified.

F ACF2,SHOWGSO

The SHOWGSO command displays the active GSO options at the console.

F ACF2,SHOWSYS

The SHOWSYS command displays at the console the startup and the current, active sysids as well as the time at which the sysid was last changed.

F ACF2,SETSYS(sysid)

The SETSYS command sets the current system id (SYSID). A REFRESH of the Global System Options (GSO) must be done to enact any changes to the GSO records for the new SYSID.

F ACF2,TRACEGSO(ALL/CONSOLE/OFF/SECURITY/SMF/SYSLOG)

The TRACEGSO command controls the destination of trace records for GSO events. (It can be used to record or display all changes to the ACF2 GSO records.) The parameter specified with this command can be:

ALL to write messages to all options listed.

CONSOLE to display records at the console and also write them in the SYSLOG file.

OFF to turn off the trace facility.

SECURITY to write records to the SECURITY console and create SYSLOG records.

SMF to write records in the SMF files.

SYSLOG to write records in the SYSLOG file (but not display them).

P ACF2

This command terminates ACF2. If ACF2 is stopped after it has been started or if the system is brought up without ACF2 started, numerous messages to the operator will be displayed. When ACF2 is in the system, it makes its presence known. Every request to start a job or to access a dataset will need operator verification (to allow access).

S ACF2[OOE] [,PARM=]

This command starts ACF2. When the START command is issued, the following options may be entered in the PARM= field:

NOBACKUP - indicates that the automatic twenty-four hour backup dump of the ACF2 databases should be bypassed.

COMMAND(string) - specifies an OS console command string. After the initialization (start) of ACF2 has completed, ACF2 will automatically execute any tasks you have specified in the string (such as sending a message or starting Special Started tasks).

DDSN(dsn-group) - specifies the dataset name group names that are defined in the ACFFDR. This can be used to initiate ACF2 using an alternate set of ACF2 databases, if necessary.

SYSID(system-id) - specifies 1 to 8 characters defining the default SYSID for the system on which ACF2 is being started. This is the default id that will be used if none is specified for entry of ACF subcommands under CONTROL(GSO) setting and for the writing of ACF2 SMF records. To find out what this default system id is at any given time, use the ACF SHOW SYSTEMS subcommand.

OOE - directs any 'SYSUDUMP' data for ACFMAIN abends to the installations printer named OOE.

TRACEGSO(destination-option) - sends TRACEGSO records to the destination indicated. Refer to the explnation of the operator command F ACF2,TRACEGSO mentioned earlier in this section of the manual.

The following operator commands pertain to the ACF2 LOOKASIDE BUFFER (LAB) feature:

F ACF2,DELLABL(logonid/logonid-mask)

This command is used to clear all lookaside buffers that correspond to the specific Logonid entered, or to the group of Logonids matching the specified Logonid mask.

F ACF2,DELLABN(node/node-mask)

Lookaside buffer entries for a specific node or group of nodes may be cleared by use of this command.

F ACF2,DELLABS(source-id/source-id-mask)

Similarly, lookaside buffer entries corresponding to the specified source-id or mask will be cleared by use of this command.

SPECIAL ACF2 PROCEDURES

This chapter discusses the following procedures:

- Special procedures and considerations for TSO fullscreen logon
- Backup Procedures
- System access without ACF2 being active

SPECIAL PROCEDURES AND CONSIDERATIONS FOR TSO FULLSCREEN LOGON

This section discusses the following procedures and considerations surrounding TSO fullscreen logon:

- Authorization for fullscreen logon.
- Retention of logon values from session to session
- Accommodations for the User Attribute Dataset (UADS)
- Modification of the logon screen format.

Authorization for Fullscreen Logon and Modification of Values of Logon Operands

Authorization to use the fullscreen logon feature is determined by the TSOFSCRN field of the Logonid record. Authorization to change information displayed on the logon screen is determined by the following fields of the Logonid record. The following table describes these fields.

Parameter/Option	Field in Logonid Record
PROCEDURE	LGN-PROC, PMT-PROC, TSOPROC, VLD-PROC
ACCT NMBR	LGN-ACCT, PMT-ACCT, VLD-ACCT
SIZE	LGN-SIZE, TSOSIZE
PERFORM	LGN-PERF, TSO-PERF
UNIT	LGN-UNIT, TSOUNIT
TIME	LGN-TIME, TSOTIME
NOMAIL	MAIL
NONOTICE	NOTICES
RECOVER	LGN-RCVR, RECOVER

* Fields whose names begin with "LGN" indicate permission to change a particular operand at logon time. For instance, if the LGN-PROC field of the Logonid record has been specified, the user has permission to specify a TSO logon procedure name at logon time.

* Fields whose names begin with "PMT" determine whether the user will be forced to specify a value for the corresponding operand during each logon. For instance, if the PMT-PROC bit field is on, the user will be prompted for a procedure name at logon time.

* Fields whose names begin with "TSO" or other characters may provide default values to be displayed when a value for an option is not saved from the previous session. For instance, the TSOPROC field contains the user's default TSO logon procedure name.

* Only ACCOUNT and PROC can be treated as "resources". The VLD prefix is used to indicate Resource Validation. Refer to the section on resource rules in this manual.

For a description of each of the Logonid record fields just mentioned, refer to the chapter on Logonid records. For information on the retention of logon values refer to the section "TSO Fullscreen Logon Retention Records".

Retention of Logon Values from Session to Session

An installation may choose to retain individual logon values from session to session. This option is specified by the FSRETAIN field of the GSO TSO record in the Infostorage database. Note that this retention of logon values is a system-wide option rather than an individual option for each user. For further information on the TSO record, refer to the chapter on GSO records.

Retention of logon values only occurs if the user has a fullscreen attribute. Furthermore, only logon values that differ from the default values are saved. This information is in the Infostorage database. See the section in this manual about the TSO CONTROL setting.

Automatic deletion of the logon values retained for a user can occur when either:

* The user's Logonid record is deleted.

* All retained values are identical to those already specified in the user's Logonid record.

Accommodations for UADS

If an installation is also using UADS processing for TSO, the ACF2 TSO fullscreen screen will not appear. The installation may be using multiple account numbers/procedure names that make up a "tree structure" for each user. UADS "passwords" act as index pointers to the account number/procedure requested for the session. To specify the logon account number or procedure to be used, the user enters his UADS password with the INDEX keyword at logon time.

An individual user's ability to specify a UADS password at logon time is determined by the LGN-INDX field in the Logonid record. See the chapter on the Logonid Record for further information on these fields.

If the installation wishes to have UADS in effect, the UADS field in the OPTS GSO record must be specified. See the chapter on GSO records for further information.

BACKUP PROCEDURES

Beginning in Release 4.1, the installation has the option of dynamically allocating backup work files and backup files. The @DDSN macro specifies a group of dynamic allocation dataset names. Datasets are dynamically allocated at startup time and dea:located immediately to insure allocation can be completed. At backup time they are dynamically allocated and backup processing is performed. Information for the backup files is supplied via the GSO BACKUP record (see "Fields of GSO Records" in this manual). Any pre-allocated DD will override the GSO BACKUP record or the @DDSN description. The 'SHOW DDSN' subcommand of the ACF command displays the values specified in the @DDSN macro and the datasets in actual use.

ACF2 can be started with AUTOBACKUP, for dynamic or pre-allocation, or NOBACKUP, meaning allocation will not take place. At start-up time console messages indicate the status of the allocation:

PRE - pre-allocated
FDR - dynamically allocated
NOA - not allocated

If NOBACKUP is specified, console messages will not be displayed for backup files.

Work and backup files are dynamically allocated based on file information in the GSO BACKUP record. To dynamically allocate files:

> Remove the SYSUT1 DD card from the ACF2 start-up procedure as it will override the GSO BACKUP description. Work file parameters are given in this manual under the GSO BACKUP record.
>
> At backup time, if a work file or a backup file cannot be allocated, a warning message is issued to the console. The backup will stop. ACF2 processing continues, and the next clusters will be processed. If allocation is successful, the backup is performed and the backup datasets are freed.
>
> If running under Release 4.0, the GSO BACKUP record does not contain the work file support parameters. You MUST expand the BACKUP record before using this enhancement. Delete the old GSO BACKUP record and insert a new one. Modify the new GSO BACKUP record to fit the installation's requirements, and do a REFRESH to update the file. Refer in this manual to the DELETE, INSERT, and REFRESH commands. Do not attempt to use dynamic allocation without changing the GSO BACKUP record. Do not share the expanded BACKUP record with an acf2/MVS Release 4.0 system.

SYSTEM ACCESS WITHOUT ACF2 BEING ACTIVE

Normally, if ACF2 has been stopped or has not been started initially, new jobs cannot enter the system. However, if it is necessary to access the system without ACF2 active, there are procedures to do so. For example, if a hardware or software failure destroyed the ACF2 databases, the ACF2 recovery jobs which recreate these databases would have to be run without ACF2 active.

When ACF2 is not active, all jobs which are executing, plus new jobs entering the system, require operator intervention. Exactly when operator intervention is required depends on the following:

1. ACF2 was not started

   In this case, the operator receives a query at each step initiation. The operator can instruct the system to either:

   a) allow the step to continue

   b) cancel the step and, therefore, the entire job.

   c) new jobs are allowed to enter the system.

2. ACF2 was started but was removed (via "P ACF2" command)

   In this case, the operator receives a query when the job begins execution, at each step initiation, each time a job attempts to access a dataset, and whenever a started task attempts to access a dataset (if the STC field is specified in the GSO OPTS record). No new jobs can enter the system. One of three actions can be taken:

   a) allow the job (or started task) to continue,

   b) cancel the job,

   c) instruct the job to wait for ACF2 to start. Note that the operator should enter the wait response only if ACF2 is to be restarted.

3. ACF2 started after a job started

   Once ACF2 has been started, no further operator intervention is needed.

4. ACF2 has not been brought up yet. The 'FORCE' parm used at TSO logon time allows a TSO session to be initiated if ACF2 has not been brought up. This requires that the user be defined in the UADS database.

IMPORTANT: Whenever a job or task is allowed to access the system without ACF2 active, it should be noted that automatic dataset or resource checking cannot be performed!

JOBS ON THE READER

Jobs read via the reader when ACF2 is not active will be flushed from the system.

INDEX