CA ACF2™ for z/VM

Implementation Planning Guide

r12

Second Edition

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

This chapter provides general information about CA ACF2™ for VM (CA ACF2 for z/VM). New CA ACF2 for z/VM users, or current users who want to review basic information, should read this chapter.

This section contains the following topics:

Why CA ACF2 for z/VM?

CA ACF2 for z/VM is a systems software product that helps control the use of computer resources, including system access and data (for example, minidisks, CMS files, and MVS and VSE data sets). CA ACF2 for z/VM controls these resources by forcing you to enter a unique logonid and password to access the system. Your logonid can be up to eight characters long and identifies you to CA ACF2 for z/VM. You are assigned a unique logonid that you must enter every time you access the system. After you enter your logonid correctly, CA ACF2 for z/VM prompts you for your password. Passwords can be up to eight characters long. The difference between a logonid and a password is that any number of people can know your logonid, but only you and CA ACF2 for z/VM know your password. Never write down your password or share it with anyone. Keep it to yourself and change it often.

Your logonid and password are validated against a central CA ACF2 for z/VM database where a security administrator has previously defined one logonid record for every authorized user. A logonid record defines what you can and cannot do on the system. It also provides a history of what you have done so far, for example, how many password violations you have committed, and what types of data you have tried to access.

After you have successfully logged on, CA ACF2 for z/VM verifies that each data access request is authorized. CA ACF2 for z/VM determines whether a request is authorized by checking if:

- The user **owns** the data. Ownership is defined by the prefix field in your logonid record.
- An access rule has been written to specifically allow the access. An access rule defines the conditions when data can be accessed. Only the data owner or security administrator can write access rules. (In a centralized environment, only a security administrator can write rules.)

If your request is not authorized, CA ACF2 for z/VM denies and logs the action by creating a System Management Facility (SMF) record. You can review and edit these records with the CA ACF2 for z/VM reports.

A number of options let you tailor CA ACF2 for z/VM controls to your needs. An important aspect of preplanning is to decide which options to use and how to migrate your system from its current level of security to one of full CA ACF2 for z/VM security where all data is protected by default.For additional information on the other CA ACF2 for z/VM features, see the *Administrator Guide*.

System Integrity with CA ACF2 for z/VM

Correct implementation of CA ACF2 for z/VM provides a significant enhancement to data security because it greatly reduces unauthorized accesses and similar security exposures. Use CA ACF2 for z/VM as part of an overall approach to security. CA ACF2 for z/VM supports a number of management controls. These controls include:

- Separation of function
- Individual responsibility and accountability
- Access to data on a need-to-know basis
- Detailed auditability of system events.

For the VM environment, a proper installation of CA ACF2 for z/VM with appropriate rules and system options provides comprehensive protection. IBM has issued an integrity statement for the CP component of VM. As a result, CA ACF2 for z/VM uses the features of CP to secure data and resources. By controlling CP LINK and ATTACH commands, CA ACF2 for z/VM can tightly secure DASD accesses. CA ACF2 for z/VM offers further protection through the control of CP commands and diagnose instructions, and the control of the DASD Dump Restore (DDR) service program.

Under the constraints of the CMS environment, CA ACF2 for z/VM provides a high level of security. IBM has not yet issued an integrity statement for CMS and it is unlikely that they will issue such a statement. The CMS environment, with limited inherent security, lets the general user program execute in supervisor state, enter storage protect key zero, execute privileged instructions, issue input/output commands, and process interrupts independently of the CMS nucleus. The sophisticated user could use these CMS features to modify the CMS nucleus and compromise CMS and OS file-level security. This lack of integrity in the CMS environment limits any security system from providing absolute file-level protection. CA ACF2 for z/VM allows for separate READ and WRITE validations. WRITE does not imply READ. On the minidisk level, validations for read links (such as R, RR, and so on) are validated as READ, and write links (such as W, WR, MR, MW, and so on) are validated as WRITE. However, some write links can end up as R/O links (such as if another user has the disk linked R/W). Also, once a write link is allowed, CP establishes a R/W link. This means that CP, and CA ACF2 for z/VM allows data to be read and written at the minidisk level. File level security does allow for WRITE-ONLY files in the constraints of the CMS operating system.

CA ACF2 for z/VM greatly improves the limited file-level protection provided by CMS. When combined with CA ACF2 for z/VM DASD protection features, other features protect files against tampering and accidental data destruction. In addition, CA ACF2 for z/VM logs any attempts to violate security.

What Does CA ACF2 for z/VM Do?

CA ACF2 for z/VM:

- Protects computer resources by default
- Provides for system-wide access with one password
- Controlls access to data and resources
- Ensures individual accountability
- Provides for the separation of function
- Provides for the backup and recovery of its three databases.

Protection by Default

No action is required to secure data or resources. All data and resources are protected until the owner of the data or a security administrator takes action to allow access. This approach provides the most effective type of security because it reduces the likelihood of exposures through omission of security controls. It also reduces the amount of work required to secure the environment.

SystemWide Logon Access with One Password

CA ACF2 for z/VM has facilities that let you access any virtual machine without having to know the password for each user ID. Single password access enhances security because you do not have to write down the password for each user ID you need to access. CA ACF2 for z/VM keeps track of who is accessing the virtual machine and records this information in SMF records. You can get this information by issuing ACFSERVE QUERY commands.

The facilities for single password access are:

Autolog protection facility

Allows authorized users to autolog another virtual machine without a password.

Group logon facility

Requires users to identify themselves with their own logonid and password when they log onto a designated user ID. CA ACF2 for z/VM checks the access against the resource rules to verify if the access is allowed.

LOGON-BY

Allows a user to access another user ID by supplying his own logonid and password. Access is allowed only if the user is authorized by a resource rule.

Controlled Access to Data and Resources

In an CA ACF2 for z/VM-protected system, an individual, department, or group owns all data and resources. When data is owned, someone is responsible for it. Those who own data can access that data at any time. They can also allow others to use their data.

CA ACF2 for z/VM provides protection by default for all system and site-defined resources. Controlled access to data means that only the owner of the data or a security administrator can allow access to the data. Security administrators can create, change, or delete CA ACF2 for z/VM records. They can also modify the protection parameters of resources. They decide when, under what conditions, and with whom they want to allow access to the data.

Individual Accountability

Each CA ACF2 for z/VM user is assigned a system user ID called a logonid. The logonid identifies the user to CA ACF2 for z/VM. CA ACF2 for z/VM can trace suspicious activity, such as repeated attempts to access protected files or programs or excessive logon violations, to a specific individual. In this way, individuals are accountable.

Each logonid is protected by a password, known only to the owner. By correctly entering a password, a user lets CA ACF2 for z/VM know he is the person assigned to the logonid. At logon, a user must enter his logonid and password to gain access to the system. CA ACF2 for z/VM uses one-way encryption to secure passwords. The management and auditing principle of individual accountability makes reporting activities especially important. You can tell CA ACF2 for z/VM to maintain an audit trail for any selected activities, applications, or users you want to examine. CA ACF2 for z/VM reports reveal if individuals are abusing their privileges.

Separation of Function

CA ACF2 for z/VM has three online databases that contain information about system users, what they can do, and other records that describe CA ACF2 for z/VM and site options. CA ACF2 for z/VM records all changes to the databases in z/OS-compatible SMF records. These SMF records provide recovery of the CA ACF2 for z/VM databases in the event of a system disaster. You can also direct the ACF2VM service machine to take periodic backups of the databases, and initiate tasks to rebuild offline copies of the databases.

What Does CA ACF2 for z/VM Control?

CA ACF2 for z/VM controls access to your computer system, data, and resources. Unique CA ACF2 for z/VM records and rules define users to the system and control how data and resources are accessed. CA ACF2 for z/VM design is based on the philosophy that only authorized system users are granted access at the data and resource level.

System Access

CA ACF2 for z/VM controls access to your VM system. It lets an account manager assign a unique logonid for each user. CA ACF2 for z/VM stores the logonid and other information that defines the user’s privileges in logonid records on the Logonid database. In addition to the logonid, account managers can also specify a user’s specific privileges, such as whether he can create, update, and delete logonid records, and whether his access privileges are confined to a specific shift or time period.

CA ACF2 for z/VM protects logonid records in two ways. Although CA ACF2 for z/VM allows each user to display his own logonid record, he cannot grant himself sensitive privileges or view the logonid records of other users unless he is authorized. Only privileged users can display, create, update, and delete logonid records of other users.

CA ACF2 for z/VM also protects logonids by requiring users to enter their logonids and passwords to gain access to the system. After you enter your logonid and password, CA ACF2 for z/VM brings your logonid record into storage to verify your privileges. If you are authorized and your logonid and password match, you can access the system. If you are not authorized to access the system or your password does not match, CA ACF2 for z/VM aborts your request and you cannot log on. CA ACF2 for z/VM creates audit records that log the unsuccessful access attempt. To further protect your system from unauthorized access or prevent the misuse of logonids, CA ACF2 for z/VM provides an option that lets you deny access or suspend the logonid of a user who cannot match his password a specified number of times in a given day. Only a security administrator can reactivate a suspended logonid.

Data Access

CA ACF2 for z/VM gives a data owner unlimited access to his data. In the VM environment, each VM user ID is assigned a minidisk or SFS file space. The person who is assigned to a virtual machine is considered the owner of the virtual machine minidisks and SFS file space. You can redirect data ownership through the PREFIX field of the logonid record.

To allow others to access data, you must write access rules. Access rules specify who can access specific data and under what conditions access can occur. If your site runs under the centralized security option, security administrators must write the access rules. If you are running under the decentralized option, data owners can write access rules. If no access rules exist, only the data owner or security administrator can access the data. For more information on centralized and decentralized security, see Centralized and Decentralized Security Administration in “How Does CA ACF2 for z/VM Work?”

Access rules are stored on the Rule database in records called rule sets. The first request to access data brings the appropriate rule set into memory and places it in a cache for future reference. CA ACF2 for z/VM validates the request against the rule set to determine whether the level of access to the data is prevented (the default), allowed but logged, or allowed. If the rule set instructs CA ACF2 for z/VM to prevent the access, or if no rule set exists, the access is automatically prevented and CA ACF2 for z/VM writes an SMF record. If the rule set instructs CA ACF2 for z/VM to log the access, the access is allowed, and CA ACF2 for z/VM writes an SMF record. If the request is allowed by a rule entry, access is allowed and CA ACF2 for z/VM does not create an SMF record.

Resource Access

Unlike data sets, which can be owned by a user, the resources of a computer system cannot be owned. Resources are the AUTOLOG command, group user IDs, the DIAL command, or any other resource your site wants to define. To control the use of resources, security administrators can write resource rules. Resource rules specify whom and under what conditions resources can be shared. CA ACF2 for z/VM stores the rule sets for accessing logical resources in the Infostorage database.

CA ACF2 for z/VM validates requests against resource rule sets to determine whether access to a specific resource is prevented (the default), allowed but logged, or allowed. If the rule set instructs CA ACF2 for z/VM to prevent the access or if no rule set exists, CA ACF2 for z/VM denies access to the resource and creates an SMF record. If the rule set instructs CA ACF2 for z/VM to log the access, CA ACF2 for z/VM allows access to the resource and creates an SMF record. If the request is allowed by an entry in the rule set, CA ACF2 for z/VM lets the user access the resource.

Access Protection

CA ACF2 for z/VM stores other types of records in the Infostorage database that further restrict access to the system, data, and resources. For example, you can create infostorage records that:

- Specify when users can log onto the system (shift records)
- Specify what terminals a user can use to log onto the system (source records)
- Limit the privileges of special users (scope records)
- Define system-wide options (VM Option records).

Components of CA ACF2 for z/VM

CA ACF2 for z/VM consists of the following components:

CA ACF2 for z/VM Service Machine

This service machine contains the internal control tables that CA ACF2 for z/VM uses to validate logon requests and requests for access to minidisks, CMS file IDs, MVS data sets, and ATTACH commands. It processes all requests to update the databases and generates logging records.

CA ACF2 for z/VM Modules in CP

CA ACF2 for z/VM modules are inserted into CP to control CP-related functions. These functions include validation calls, system status changes, communication with the service machine, and command limiting rule interpretation and compilation.

CA ACF2 for z/VM Modules in CMS

CA ACF2 for z/VM intercept code is inserted into CMS and becomes part of the CMS system. These modules provide functions such as communication with the CA ACF2 for z/VM service machine and CMS file-level security.

CA ACF2 for z/VM Database Recovery

The CA ACF2 for z/VM ACFRECVR utility recreates one or all of the databases if they become damaged. ACFRECVR accepts one or more of the database backup files and selected database maintenance logging records and merges them to create a current database.

CA Earl™

The CA Earl™ facility is provided with CA ACF2 for z/VM. It's is a powerful, easy-to-use reporting language with 24 straightforward commands. With CA Earl™, you can:

- Create custom CA ACF2 for z/VM reports quickly
- Combine sequential files created on VM and MVS systems and process all the data in a single run
- Write entirely new reports using advanced features
- Include local information in SMF records through an SMF preprocessor exit.

CA ACF2 for z/VM Full-screen Feature

This is a menu-driven, easy-to-use method of maintaining logonids, writing access rules and resource rules, and running CA ACF2 for z/VM reports. To access the full-screen primary menu, enter the following command from CMS:

```
ACFFS
```

CA ACF2 for z/VM Security and Maintenance Reports

We provide a report utility (ACFRPTS) to format CA ACF2 for z/VM reports. Four basic types of reports are available:

Data and Resource Logging Violation Reports

These reports contain logging and violation records for the five reports below.

ACFRPTCL

Reports each command limiting and diagnose limiting logging or violation record.

ACFRPTCT

Reports each ACFSERVE command issued, the type of command, and the user issuing the command.

ACFRPTDL

Reports the violation and logging records for all commands issued to the DirMaint service machine.

ACFRPTDS

Reports the logging and violation records for minidisks, CMS files, OS data sets, DOS files, and attachable DASD devices.

ACFRPTRV

Reports the resource violation and logging records for all activity related to user-defined logical resources.

Database Maintenance Reports

These reports document updates to the CA ACF2 for z/VM databases.

ACFRPTEL

Reports modifications made to resource rule sets and other Infostorage database records.

ACFRPTLL

Reports modifications to the Logonid database.

ACFRPTRL

Reports modifications to the Rule database.

Cross Reference Reports

ACFRPTPW

Journals each unsuccessful system access attempt.

See the *Reports and Utilities Guide* for complete information on these and all reports.

CA ACF2 for z/VM Commands

CA ACF2 for z/VM provides a powerful set of tools to maintain the following:

- logonid records
- entry records
- scope record
- shift and zone records.

CA ACF2 for z/VM also provides commands to compile, store and decompile the following:

- Access rule sets
- resource rules
- Command limiting rule sets
- Models
- Diagnose limiting rule sets

ACF

Primary interface with the ACF2 service machine for creating, deleting, and maintaining CA ACF2 for z/VM records.

ACFCOMP

Compiles an access rule set with one command. You can enter input for the rule set directly from the terminal or from a CMS file.

ACFDCMP

Decompiles an access rule set at the terminal. Alternately, you can write the decompiled rule set to a CMS file for easy modification.

ACFNRULE

Inserts or deletes a single rule entry in an access rule set. It provides a convenient way to quickly edit a rule set.

ACFSERVE

A CP command that lets you communicate directly with the ACF2 service machine. The ACFSERVE commands can instruct CA ACF2 for z/VM to take a database backup, reset a password violation count, manage the SMF records created by CA ACF2 for z/VM, reload resident rules, display the status of CA ACF2 for z/VM, or switch SMF files.

CA ACF2 for z/VM Utilities

We provide many utilities to assist in the initial implementation and installation of CA ACF2 for z/VM. We also provide utilities to help streamline ongoing maintenance activities, recover databases, and SMF management.

CA ACF2 for z/VM Help Files

A set of online help files can assist you when using the ACF command, full-screen, and other CA ACF2 for z/VM-supplied commands.

Message Help

Help is also available for all CA ACF2 for z/VM messages. If you need help with a message, enter the following command:

```
HELP message_number
```

A help screen displays with information on that message.

What is the System Request Facility?

The System Request Facility (SRF) provides facilities that let your application programs interface to CA ACF2 for z/VM. You can also use SRF to interface other products to CA ACF2 for z/VM. In many instances, the appropriate place for security validation processing is inside the application program. By using SRF, you can invoke most types of CA ACF2 for z/VM security processing directly from your own application programs. SRF includes the macros and subroutines you need to interface application programs to CA ACF2 for z/VM. In addition, you can use all the other facilities of CA ACF2 for z/VM, such as the databases, ACF subcommands, ACFSERVE commands, and reports.

SRF also includes a comprehensive resource facility that defines logical resources and validates user access to these resources. See the *System Programmer's Guide* for more information.

Modifying CA ACF2 for z/VM

There are two types of local modifications you should consider: modifications to standard VM components and standard CA ACF2 for z/VM components.

Existing or Planned Local Modifications

Many sites modify standard VM components (CP, CMS, exits) to provide added functions. You should review these local modifications before you install CA ACF2 for z/VM. In some cases, such as VM logon processing, the local code may perform some security-related function that is replaced by CA ACF2 for z/VM. In these cases, you can remove the local code. In other cases, the code must remain to perform some needed local function and must reside in the same exit or front-end the same intercept-point that CA ACF2 for z/VM uses. This requires a decision as to which processing (CA ACF2 for z/VM or the local modification) should come first. CA ACF2 for z/VM should come first because it determines whether the requested function is legitimate. If CA ACF2 for z/VM lets the process continue, the local code is useful.

Local Modifications to CA ACF2 for z/VM

CA ACF2 for z/VM is designed to provide a high level of flexibility to reduce the need for modification. Yet many sites find new ways of using functions and sometimes tailor them beyond what can be done with the provided options. This usually requires producing local code for one or more of the provided exits. Most sites find these modifications can wait until the system is installed, tested, and operational. Most sites install without using the local exits. While you should consider possible local modifications (due to some special naming conventions or transitional implementation plans) and plan for this during installation planning, you can perform the actual coding and activation of these changes after the initial installation and IPL has established the base system.

At What Maintenance Level Should VM Be Operating?

Each distribution tape provides the current requirements for operating system maintenance levels. In general, VM should be at a relatively recent level (maintenance provided by the system hardware vendor applied in six months of the release date). It also should have been running at that level in production a minimum of one week to provide a valid, stable base for code installation and testing.

Planning

You should be planning for the technical installation of CA ACF2 for z/VM and its ongoing systems maintenance concurrently with the general implementation planning. Some of the topics that you should consider in the technical portion are listed below. These are in general chronological sequence but, with careful planning and coordination, you can do many of these steps in parallel.

- Planning and organizing for installation, including schedules, responsibility assignments, and check points.
- Establishing a stable operating system at the minimum required maintenance level.
- Reviewing special program products (data or operational management packages) that already exist on the system and planning for the resolution of any possible conflicts.
- Reviewing local system modifications (local modifications to CP, CMS, or other system components) and planning for the resolution of any logic or physical (locational) conflicts.
- Planning for the creation of logonid records so users can access the system. This includes executing the ACFLIDGN utility to create logonid records and modifying these logonids with the ACF command.
- Planning for and processing the distribution tape and steps, such as:
  - Running CA-ACTIVATOR
  - Applying maintenance from pertinent genlevel maintenance updates (if any).
- Unloading the tape is usually done early to obtain other information that you might need to proceed with some of the other steps described here.
  - Selecting the ACFFDR and VMO options (as applicable) and preparing the assemblies and modules for CA ACF2 for z/VM processing.
- Completing the installation process, including:
  - Creating the final target system with all CA ACF2 for z/VM CP and CMS modules
  - Making final adjustments for the local running environment
  - Defining and initializing the CA ACF2 for z/VM databases or any additional groups of databases
  - IPLing the system (normally in LOG mode)
  - Establishing the initial system users (through the ACFLIDGN utility)
  - Testing the system and rules.

- Tailoring and testing the database recovery procedures.
- Running and reviewing CA ACF2 for z/VM reports.
- Establishing practices and procedures for:
  - The timely, ongoing application of CA ACF2 for z/VM and system maintenance.
  - The periodic review and modification of CA ACF2 for z/VM system options (updating the ACFFDR, including reassembly and reload of the new version into storage using the ACFSERVE RELOAD FDR command).
  - Handling new features and migration of controls to other areas, such as DirMaint and VMBACKUP.

Other Product Interfaces

There might be several other software products running under CA ACF2 for z/VM at your site. These can include database systems, disk management and archiving systems, tape management systems, sorts, and other utilities and packages. Many of them can continue to operate without any changes or special information related to implementing CA ACF2 for z/VM. With some software products, however, you may notice some minor differences and might want to consider using special CA ACF2 for z/VM interfaces to provide the best overall security level.

Other Documentation and Publications

Interfaces to other products are available through CA, CA ACF2 for z/VM users, or other software vendors. Review other CA ACF2 for z/VM guides (*System* Programmer's *Guide* and the *Installation Guide*) and CA periodicals for the most current information. "Planning the Implementation" provides information you need to know to prepare for CA ACF2 for z/VM implementation.

Chapter 2: How Does CA ACF2 for z/VM Work?

The previous chapter introduced you to the concepts of default protection, controlled access, individual accountability, and separation of function. It also defined some of the key terms, such as logonids, access rules, and resources that we use to describe the components of CA ACF2 for z/VM.

This chapter builds on your understanding to provide more details about how CA ACF2 for z/VM secures your computer system.

This section contains the following topics:

Controlling System Access

CA ACF2 for z/VM uses the logonid record to control access to your computer system. This section explains the following topics:

- Logonid records
- Passwords
- System access
- CA ACF2 for z/VM access validation.

Logonid Records

The logonid record is the most important CA ACF2 for z/VM record. It identifies a user on a particular system protected by CA ACF2 for z/VM. Account managers define users by creating a unique logonid record that enforces individual accountability. Account managers are also responsible for assigning special privileges to users. They specify these privileges in the logonid record fields. The fields contain information to identify a user's attributes, such as:

- The user's privileges
- The user's system history (such as the number of security violations to date)
- Other CA ACF2 for z/VM system-related information such as the maximum number of days allowed between password changes or the time allowed for system access.

Logonid Record Fields

The logonid record is variable in length, with a maximum of 1024 bytes. CA ACF2 for z/VM reserves 640 bytes for its use. Your site can use the remaining bytes to define your own fields. The standard fields of the logonid record are organized into the following sections:

Identification

Contains information such as the user's logonid, name, phone number, and user identification string (UID).

Cancel/Suspend

Specifies if a logonid has been canceled or suspended. This section is only displayed if the user's logonid has been canceled or suspended.

Privileges

Specifies what auser can do, such as defining his ability to process other CA ACF2 for z/VM records.

Access

Specifies the number of accesses a user has made and the time, date, and source of the last access.

Password

Contains statistics on the number of violations, expiration date, and the date the password was last changed.

Statistics

Contains the total number of security violations and the date and time the logonid was last updated.

Restrictions

Contains information about access to data and conditions for logon, such as shift.

You can define fields for your data center in any of these sections.

Sample Logonid Record

Below is a sample logonid record for Ann Smith, an auditor in the accounting department.

```
TLCAMS        ACCTGAUDTLCAMS  ANN SMITH   EXT.413
              DEPT(ACCTG)   FUNCTION(AUD)

CANCEL/SUSPEND EXPIRE(12/29/03)

PRIVILEGES    DUMPAUTH JOB VM

ACCESS        ACC-CNT(133) ACC-DATE(9/15/03) ACC-SRCE(LV248)
              ACC-TIME(09:21)

PASSWORD      MAXDAYS (30) PSWD-DAT(9/15/03)
              PSWD-TOD(9/01/03-13:23) PSWD-VIO(1)

TSO           DFT-PFX(TLCAMS)

STATISTICS    SEC-VIO(1) UPD-TOD(8/11/03-09:21)

RESTRICTIONS  PREFIX(TLCAMS)
```

Field Descriptions

TLCAMS

Specifies the user’s logonid.

ACCTGAUDTLCAMS

Specifies the user identification string (UID). This example has defined the UID as the DEPT field, followed by the FUNCTION field, followed by the logonid. The values ACCTG, AUD, and TLCAMS are taken from these fields to form the UID ACCTGAUDTLCAMS. The DEPT and FUNCTION fields have been defined by the site and do not appear in the logonid record supplied with CA ACF2 for z/VM.

ANN SMITH

Specifies the user's name.

EXT. 413

Specifies the user's telephone number.

DEPT(ACCTG)

Indicates the user is in the Accounting department.

FUNCTION(AUD)

Indicates the user is an auditor.

CANCEL/SUSPEND

Indicates if the logonid has been canceled or suspended.

EXPIRE(12/29/03)

Indicates the expiration date. In this example, Ann Smith's logonid record is temporary because it expires on December 29, 2003.

PRIVILEGES

Indicates what privileges the user has been granted.

DUMPAUTH

Indicates that the user can generate a storage dump.

JOB

Indicates that the user can submit jobs.

VM

Indicates that the user can use VM.

ACCESS

Indicates how many times the user has accessed the system, and when and where last access attempts were made.

ACC-CNT(133)

Indicates that USER01 has made 133 system accesses.

ACC-DATE(9/15/03)

Indicates that USER01's last access was on September 15, 2003.

ACC-SRCE(LV248)

Indicates that USER01’s last access was from a terminal identified as LV248.

ACC-TIME(09:21)

Indicates USER01’s access was made on September 15, 2003 at 09:21.

PASSWORD

Indicates the last time the user entered an incorrect password, the last time the password was changed, and how many password violations were made to date.

MAXDAYS (30)

Indicates that 30 days must elapse before the user’s password must be changed.

PSWD-DAT(9/15/03)

Indicates the user’s last invalid password attempt was made on September 15, 2003.

PSWD-TOD(9/01/03-13:23)

Indicates that the last time the user changed their password was September 1, 2003 at 1:23 p.m.

PSWD-VIO(1)

Indicates that on September 15, 2003, the user made one invalid password attempt. CA ACF2 for z/VM automatically resets PSWD-VIO to one on the first invalid password attempt on a new day.

STATISTICS

Indicates how many security violations the user has and when their logonid record was last updated.

SEC-VIO(1)

Indicates that, to date, the user has one security violation.

UPD-TOD(8/11/03-09:21)

Indicates that the user’s logonid record was last updated on 8/11/03 at 09:21.

RESTRICTIONS

Indicates what records the user can access.

PREFIX(TLCAMS)

Identifies what PREFIX the user owns. The user can access data owned by TLCAMS without validation. The prefix is TLCAMS (same as the logonid). This field gives the user ownership of all records with a high-level index of TLCAMS.

You can see that a logonid record contains a great deal of information about a user. For more detailed information about logonid records, see the *Administrator Guide.*

CA ACF2 for z/VM Privileges

Logonid record fields also specify special user privileges. Privileges give the user access to data, resources, and CA ACF2 for z/VM records. The CA ACF2 for z/VM privileges are listed below.

ACCOUNT

Indicates that this user is an account manager. An account manager can insert, delete, and change logonid records in the limits defined by his scope. (Scope is explained later in this chapter.) Account managers usually establish, maintain, and delete logonid records. The ACCOUNT privilege grants no authority for writing rules or processing other CA ACF2 for z/VM records.

SECURITY

Indicates that this user is a security administrator and can access all data, protected programs, and resources. A security administrator can insert, change, list, and delete access and resource rules and list and change certain fields of logonid records. He cannot insert or delete logonid records unless he also has the ACCOUNT privilege. He can insert, change, list, and delete any infostorage records in his scope. You can use a scope record to protect any access granted by the SECURITY privilege.

AUDIT

Indicates that this user can list logonid records, access and resource rules, entry records, shift records, zone records, scope records, and VM Option (VMO) records. He cannot update or delete logonid records, or access any resources other than those authorized to him through rules.

CONSULT

Indicates that this user can display most fields of logonid records, and update only certain nonsecurity-related fields. The CONSULT privilege is usually given to individuals who assist other users on the system so they can answer questions about logonid record information.

LEADER

Indicates that this user has the same privileges as CONSULT, however he can only display and modify certain fields of other logonid records as defined by the SCPLIST field. (For detailed information on SCPLIST, see the *Administrator Guide.*)

USER

Indicates that this user can display their logonid record. You can specify whether a user with only the USER privilege can write access rules for their own data with a system-wide option.

A user can have more than one privilege. For example, he can have both SECURITY and ACCOUNT, which gives him all authorities associated with the SECURITY privilege and all authorities associated with the ACCOUNT privilege.

Limiting a User's Authority

As you have read, privilege fields grant users certain authorities to access data, rule sets, logonid records, and other CA ACF2 for z/VM records. Scope records restrict that authority. For example, a user with the ACCOUNT privilege can insert, change, and delete logonid records. With an assigned scope, you can limit a user with the ACCOUNT privilege to insert, change, and delete logonid records for only a certain group of users, such as his department, site, or project group. A scope is associated with a user's logonid through the SCPLIST field of the logonid record.

Other CA ACF2 for z/VM Privileges

You define all logonid record fields in macros in the Field Definition Record (ACFFDR). The ACFFDR defines other system options besides the logonid record fields. The macro entry for each logonid field contains the field name, its attributes, and information that indicates what authorization is required to modify or list that field. This gives the system field-level control over each logonid record field, whether defined by CA ACF2 for z/VM or added locally. You define your own logonid record fields by specifying new macro entries in the ACFFDR. See the *Installation Guide* for more information.

Creating Logonid Records

Users with the ACCOUNT privilege can create, change, display, and delete logonid records with the ACF command and its subcommands. They can issue these commands using the CMS ACF command or the full-screen panels. You can find complete details on how to use the ACF subcommands to process logonid records in the *Administrator Guide.*

Passwords

Every logonid must have an associated password. Before processing the password information, CA ACF2 for z/VM encrypts it in a one-way method that cannot be deciphered. CA ACF2 for z/VM never displays a password, either in its encoded or clear text form. You can specify password controls in the PSWD VMO record. The following describes some of the controls you can implement:

- The number of times you can enter an incorrect password in a single day for a given logonid before the logonid is suspended.
- The maximum number of days allowed before you must change a password to a different one.
- The minimum number of days that you must keep your password before you can change it again.
- The number of days before password expiration when CA ACF2 for z/VM issues a warning at logon to change the password on each system access.
- The minimum number of characters required for a password.

- Whether you can modify your own password at system entry time.
- Whether to use a password exit to gain control when you enter a new password at system entry or with the ACF command. This exit can deny the new password.
- Decide how many previous passwords are to be stored in history.
- Support z/OS UNIX and mainframe Linux uses of mixed-case passwords.

Accessing the System

When you try to log onto VM, CA ACF2 for z/VM verifies whether you are authorized to access the system. During the logon process, the following is verified:

- Did you enter the correct password?
- Is your logonid suspended for excessive password or security violations?
- Is your password expired?
- Is your logonid expired?
- If you are accessing another user ID through group logon or logon-by, does a resource rule allow access?

CA ACF2 for z/VM also optionally checks the following:

- Are you accessing the system from an authorized source?
- Are you accessing the system during an authorized time or day?
- Is the account number used for this logon valid?
- Did you enter your password from a nondisplay field?

You can activate these items on an as needed basis.

Whenever access is denied, CA ACF2 for z/VM displays an error message that explains why the access was denied. CA ACF2 for z/VM also creates an SMF record to log the violation.

CA ACF2 for z/VM supports the following LOGON line command:

```
LOGON logonid password
              password/new password
              BY logonid
```

From the VM logo screen, CA ACF2 for z/VM supports the following formats:

```
1.  USERID ===>  userid ________

    PASSWORD ==> oldpassword/[newpassword]

    COMMAND ===> [[BY userid ] [other-optional-operands]] ___

2.  USERID ===>  userid ________

    PASSWORD ==> [BY userid] [oldpw/newpw] ______________

    COMMAND ===> [other-optional-operands] __________

3.  USERID ===> userid ________

    PASSWORD ==> oldpw[/newpw] [BY userid] ___________

    COMMAND ===> [other-optional-operands] ________
```

Note the following:

- If you log on from the VM logo screen or through the LOGON line command and do not enter your password, CA ACF2 for z/VM responds with the following message:

  ```
  ACFpgm244R ENTER ACF2 PASSWORD
  ```

- You can abort a logon by entering a plus sign (+) or the **LOGOFF** command in response to any CA ACF2 for z/VM logon prompt.

You can also use any of the following formats to log onto CA ACF2 for z/VM:

- Where users specify a password, they can specify a new password (the password can be changed) by using the following notation:

  ```
  old-password/new-password
  ```

  Your site can prevent this. See the *Administrator Guide* for details.

- You can specify the GRPLOGON attribute with the LOGON command. Resource rules validate this parameter to determine whether CA ACF2 for z/VM allows the access as a member of a group or project.

  You can also instruct CA ACF2 for z/VM to check that any user attempting to log onto VM has the VM attribute in his logonid record.

Controlling Access to Data

CA ACF2 for z/VM protects all data by default. Data is information that you store on the system. To allow access to data, you must first write access rules. An access rule contains the information CA ACF2 for z/VM needs to determine who has access to data and under what conditions access can or cannot occur. These rules are stored on the CA ACF2 for z/VM Rule database.

What are Access Rules?

CA ACF2 for z/VM access rules define the conditions for data access. They specify who can access data and the terms under which the access can occur. Only a data owner or security administrator specifies when access is allowed. For example, a security administrator can specify a specific week or month, (5/22/99 to 5/29/99), or a time period, (9:00 A.M. to 5:00 P.M.) A data owner or security administrator can also specify if the access attempt is logged, allowed, or prevented. They can specify the type of access allowed. For example, a security administrator can specify whether the user has read, write, or execute access. You can find detailed information about writing access rules in the *Administrator* Guide.

What are Access Rule Sets?

An access rule set is a group of related access rules. Rule entries specify the data, access types, and permissions. CA ACF2 for z/VM maintains a rule set for each data set name (DSN) high-level index, although some rule sets can apply for entire volumes of DASD or tape data sets. CA ACF2 for z/VM stores rule sets in the Rule database by their key or high-level index. The key for a data set access rule set can be up to eight-characters long. The key for a volume access rule is the volume serial number (VOLSER), which can be six characters long. Rule sets are compiled and stored much like programs.

The following is a sample rule set:

```
$KEY(TLCAMS)  MODE(ABORT)
%CHANGE ACCTGMGRTLCMGR
 V0191.VOLUME UID(ACCTGAUD) R(A) E(A) W(A)
 V0191.ACCOUNTS.DATA UID(ACCTGAUD) R(A) E(A) W(A)
```

This rule set allows all accounting auditors to read and write to the ACCOUNTS.DATA file on the TLCAMS 191 disk. UIDs that start with ACCTGAUD can also link to the TLCAMS 191 minidisk because the V0191.VOLUME rule allows this. UIDs are explained later in this chapter.

This sample rule set contains:

- Control statements
- Access rule entries
- Access types
- Access permissions.

Control Statements

Control statements are parameters that begin a rule set. They specify conditions that apply to the whole rule set or rule entries. You can use two types of control statements in an CA ACF2 for z/VM rule set, those that begin with a $ (dollar sign) and those that begin with a % (percent sign). The only required control statement is $KEY, which identifies the high-level qualifier of the data set this rule set protects. In the sample rule set, $KEY indicates that this rule set applies to files whose names begin with TLCAMS.

The control statements of the rule set specify such attributes as:

- The user ID that owns the minidisk governed by this rule set. (You can use the PREFIX field to designate ownership other than the user ID.)
- User information to supply to locally-defined exits.
- The CA ACF2 for z/VM mode of the rule set. Through this control statement and the system option of RULE mode, a rule set can have its own mode. See the “Migrating to Full Security” chapter for more details.
- Which users have the authority to change the entire rule set or can only change rule entries.
- Whether CA ACF2 for z/VM sorts rule entries.

Access Rule Entries

Access rule entries are statements that specify the conditions for data access, such as the names of the files you want to allow other users to access. In a rule entry, you can specify the following parameters:

EXECUTE

Defines a specific type of access. When a user has execute only access, he cannot read or write to the data. That is, he cannot update or even look at the data, but he can use it (for example, to run a program).

FOR

Specifies the number of days that a user can access this data.

NEXTKEY

Specifies the name of an alternate rule set that CA ACF2 for z/VM checks to determine if the access is authorized.

PSEUDO DSN

Defines the information that describes the minidisks, CMS files, or OS and DOS data sets.

READ

Defines a specific type of access. When a user has read access to data, he can only read the data. He cannot write to (or update) the data.

SHIFT

Specifies the time-of-day when a user can access the system, specific data, or resources.

SOURCE

Specifies the logical input source name or source group name where this logonid must enter the system, for example, a terminal.

UID

Defines a 1- to 24- character string that is formed when CA ACF2 for z/VM combines logonid record fields that you specify. UIDs specify the user or groups of users the rule entry applies to. By using a masked UID to group users, you reduce the number of rule entries you need to write.

UNTIL

Specifies a time limit to allow only temporary access to this data.

WRITE

Defines a specific type of access. When a user has write access to data, he can write to (or update) the data.

ZONE

Specifies the time zone where this user accesses the system (the user's local time zone).

In the sample rule set shown in the What are Access Rule Sets? section, all accounting auditors (UIDs that start with ACCTGAUD) can read and write to the ACCOUNTS.DATA file on the TLCAMS 0191 disk and link to the TLCAMS 0191 minidisk as specified by the V0191.VOLUME rule.

Access Types

Access types are parameters in a rule entry that specify the type of access a user can have. A rule can specify that users can have read-only (READ), read/write (WRITE), or execute-only (EXEC) access to data. You can use a one-character abbreviation for the access type such as R for read-only, W for read/write, and E for execute-only.

Access Permissions

Access permissions are parameters in a rule entry that specify the action CA ACF2 for z/VM takes when all of the required conditions are met. Access rules instruct CA ACF2 for z/VM to enforce one of the following access types:

Allow (A)

Allow the user to access the data.

Log (L)

Allow the user to access the data, but record the access so that CA ACF2 for z/VM creates an audit trail. Specify LOG for sensitive data.

Prevent (P)

Do not allow the user to access this data. Log the attempt and create an audit trail.

Permissions are matched with types to fully describe the access you want to grant. For example, R(A), W(L) lets these users read this data and write to the data, but logs that access. In our example, UIDs that begin with ACCTGAUD can read and write to the ACCOUNTS.DATA file. By default, all other UIDs are denied access.

What is the User Identification String?

The user identification string (UID) groups users to make rule writing easier and more efficient. The UID identifies individual users or groups of users in rules for access to data and resources. Choosing the proper UID is the most important task you will do when implementing CA ACF2 for z/VM.

The UID is built at logon time for every user who accesses the system. When a user requests access to a protected object, the user's UID is matched against the rule to check if the user can access the object. The UID is a 1- to 24-character pseudo field constructed of logonid record fields. It enables you to add information such as department, group or project name, job responsibility, employee ID number, or other data necessary for determining access privileges. Your site can select which fields it uses, but it must use the same UID format for all users. The UID can include the fields supplied with CA ACF2 for z/VM or fields defined by your site. For example, you could specify that the UID be composed of five logonid record fields:

Byte	**Description**
1	Site
2	Division
3-4	Department
5-6	Job Function
7-14	Logonid

The first four fields are site-defined and the logonid is an CA ACF2 for z/VM-defined field. With this UID layout, each user defined to the system has a unique UID.

Suppose you had to define a UID for Ann Smith. She is employed with the True Lock Company in Chicago. Her user ID is TLCAMS. She works in the Marketing division in Department 02, with a Job function of 44. Ann's UID would be CM0244TLCAMS.

The advantage of the UID is that you can define subsets of users in ways that the logonid cannot. The UID allows grouping of users by any character fields defined in the logonid record. When the data or resource owner writes rules, he determines who he wants to allow access to the object.

The following examples illustrate the power and flexibility the UID provides:

- UID(C)
- UID(****86)
- UID(*M**44)
- UID(******TLCAMS)

In this example, all users in Chicago match the UID value; all users with job function 86 in the True Lock Company match the UID value. All users in the Marketing division with function 44 match the UID value. Line number four of this example shows a UID value for logonid TLCAMS. If the UID is specified this way, Ann Smith (TLCAMS) matches this value regardless of what department or site she is transferred to, and what functions she performs there.

As you can see, the UID is very powerful and simplifies rule writing. To use the UID more effectively, it is critical that you design it carefully and correctly. After users have started creating rules, you cannot modify the UID without impacting many of your existing rules. You must also consider both the length and content of the UID. The format you select must apply to your entire organization, though you can use selected fields in the string to define different functions in different identifiable groups. The UID should be long enough to contain all meaningful data that is required for the proper generation of all rules. However, redundancy or using excessive characters can make the UID too long to use effectively.

What is Masking?

You can also simplify rule writing and improve system performance by masking. In access rules, masks can represent multiple data set names or UIDs. You can replace almost all character strings used in CA ACF2 for z/VM with a character string pattern. For example, you can create a pattern to mask the data set, volume, and UID in an access rule set.

You can substitute two special symbols for characters— the asterisk (*) and the dash (—). CA ACF2 for z/VM processes these symbols differently depending upon the type of character string, that is, fixed-length strings (such as the UID) or variable-length strings. The basic function of each symbol is described below:

Asterisk (*)

Matches any character in this position. You can place any number of asterisks in a string pattern. For example, AB*D matches any string containing AB in positions 1-2 and D in position 4. ABCD, AB1D, ABXD, and ABBD match AB*D. An asterisk embedded anywhere in a string does not match a null character. For example, AB*D does not match AB D.

If you place an asterisk at the end of a string, the trailing asterisk matches a null character and every other character. For example, ABC* matches ABC, ABCD, ABC1, and ABC2.

Dash (—)

Translates into asterisks to create the maximum length of the character string. Maximum length varies according to the string type. A dash is only valid when placed at the end of a character string or when used by itself. You cannot imbed it between other characters.

If you omit a dash from a string specification with a variable length, the remainder of the name is assumed to be blanks and only matches blanks. Also, when specifying data set names, you can specify a dash as the only character of a data set name index level to indicate that any number of index levels (including zero) can be present in the target data set.

Our sample rule set did not contain any masks, but we could add a new rule entry to mask a data set name as follows:

```
$KEY(TLCAMS) MODE(ABORT)
%CHANGE ACCTGMGRTLCMGR
 V0191.VOLUME UID(ACCTGAUD) R(A) E(A) W(A)
 V0191.ACCOUNTS.DATA UID(ACCTGAUD) R(A) E(A) W(A)
 V0191.—. UID(ACCTGMGRTLCMGR) R(A) W(A) E(A)
 V—.— UID(ACCTG) R(A) E(A) W(P)
```

This new rule entry allows any user who’s UID begins with ACCTG to read and execute any file.

How Do I Use CA ACF2 for z/VM Access Rules?

Before you can write an access rule, you must be the owner of the data, a security administrator, or someone with **change** authority in the control statement of the rule set.

To write effective access rules, you must ask the following questions:

- What data do I want to allow access?
- Who do I want to access this data? What are the UIDs of the users I want to allow access? Can they be grouped? Can I mask them?
- How do I want others to use the data? Do I want them to be able to read and write to the file?

In the rule set example below, the minidisk rules allow access to the 0191 minidisk, but not to any of the files on the minidisk. Access to specific files is granted by individual rule entries. The lines that begin with an asterisk (*) are comments. They are included to help you understand the rule set.

```
* ALLOW ALL APPL-PROGRAMMERS TO READ MY COBOL FILE
* FOR THE NEXT 7 DAYS
* ALLOW MY MANAGER TO READ AND WRITE TO MY COBOL FILE
* MY INDEX/LOGONID = APP002, MGR = APP001
$KEY(APP002)
* LINKS TO MY MINIDISK MUST BE AUTHORIZED
 V0191.VOLUME UID(APP001) READ(A) WRITE(A)
 V0191.VOLUME UID(APP) READ(A)
* ACCESS TO THE SPECIFIC FILE FOLLOWS
 V0191.TEST1.COBOL UID(APP001) READ(A) WRITE(A)
 V0191.TEST1.COBOL UID(APP) READ(A) FOR(7)
```

In the above access rule, the manager (APP001) is allowed WRITE access to the TEST1.COBOL FILE on the minidisk on virtual device 0191. However, the application programmers (APP) can only read this file for seven days because the rule only specifies READ, not WRITE.

Many sites need to limit access even further. They should consider such issues as:

- What specific conditions must the user meet to access data?
- Must the users access the data from a specific group of terminals during a specific shift?

After you decide how you want to allow others access to your data, you must write the rule sets. Rule sets are like computer programs. Both are written in human-readable form (known as source code) and then converted (compiled) into machine-readable form (known as object code). You can compile the source code directly from the terminal and store it on the rule database. Or, you can create the source code in a CMS file, then compile it into object code and store it on the rule database.

You can use the ACF command and its subcommands to compile, test, store, maintain, and delete access rules on the rule database. Issue the ACF command from the CMS READY prompt. Additionally, CA ACF2 for z/VM provides full-screen panels that you can use to process access rules. You must have special privileges to issue these commands. You can find complete details on how to use the ACF subcommands to process access rules in the *Administrator* Guide.

How Does CA ACF2 for z/VM Sort Rules?

CA ACF2 for z/VM sorts access rules alphabetically by data set name and then from most specific to most general. CA ACF2 for z/VM grants privileges as determined by the first rule entry that matches the environment. The rule selection algorithm for CA ACF2 for z/VM access rules is mapped out below.

- DSN (data set name)—CA ACF2 for z/VM selects patterns from most specific to most general
- VOL (volume)—CA ACF2 for z/VM selects patterns from most specific to most general
- UID (user identification string)—CA ACF2 for z/VM selects patterns from most specific to most general
- PGM/PROG (program)—CA ACF2 for z/VM selects patterns from most specific to most general
- UNTIL—CA ACF2 for z/VM selects dates from earliest to latest.

Controlling Access to Resources

CA ACF2 for z/VM controls access to the resources of your computer system. Resources are objects or system services, such as group IDs, VM accounts, and the DIAL command. CA ACF2 for z/VM also defines some system resources. You can define your own resources for protection.

After CA ACF2 for z/VM grants a user access to the system, that user can access any data that he owns. However, because he does not own system resources, he cannot use them or allow others to use them. CA ACF2 for z/VM protects all defined system resources by default. Because no user owns system resources, a security administrator must write resource rules to allow use.

What are Resource Rules?

Security administrators write resource rules to control access to resources. CA ACF2 for z/VM stores resource rules in records up to 4K on the Infostorage database. CA ACF2 for z/VM finds resource rules on the Infostorage database by finding their type and name. Resource types specify the category of resource that is being requested, such as group IDs. Resource names are one to 40 characters long and specify an individual resource in a given resource type. Resource rules specify who can access the type and name of a particular resource and the terms under which CA ACF2 for z/VM allows access.

Like access rules, a security administrator specifies when you can use a resource. For example, a security administrator can specify a specific week or month, (5/22/99 to 5/29/99), or a time period, (9:00 A.M. to 5:00 P.M.) A security administrator can also specify if using the resource should be logged, allowed, or prevented.

What are Resource Rule Sets?

Resource rule sets are groups of related resource rules that control access to system resources. Resource rule sets are very similar to access rule sets. A resource rule set can contain all of the following:

- Control statements
- Resource rule entries
- Access permissions.

Control Statements

Control statements are parameters that begin a rule set. They specify conditions that apply to the whole rule set or to the rule entries. You can use two types of control statements in a rule set, those that begin with a $ (dollar sign) and those that begin with a % (percent sign). The only required control statements in resource rules are the $KEY and the $TYPE. The $KEY control statement is the name of the resource you want to protect. The $TYPE is the three-character type code of the resource. CA ACF2 for z/VM provides six type codes that you can modify:

ACT

Controls account use

ALG

Controls the target of the AUTOLOG command

GRP

Controls access to group IDs

DIA

Controls the target of the DIAL command

IUC

Controls access to the Inter-User Communications Vehicle (IUCV)

VMC

Controls access to the Virtual Machine Communications Facility (VMCF).

These are the default resource types. You can also define your own resource type codes. The control statements of the rule set determine such attributes as:

- The resource name and type of the record
- User information to supply to locally-defined exits
- The users who have the authority to change the entire rule set.

Resource Rule Entries

Resource rule entries are statements that specify the conditions for resource access. In a rule entry, you can specify any of the following keywords to control the access:

DATA

Specifies an optional keyword for any comments you want to enter about the rule entry. Comments can be up to 64 characters long. DATA does not affect CA ACF2 for z/VM validation.

SHIFT

Specifies an optional keyword that specifies the name of a shift record that resides on the Infostorage database. The specified shift record defines the dates and time-of-day that this user can access the system. If you do not specify this keyword, the rule entry applies to all shifts.

SOURCE

Specifies an optional keyword that defines an input source. To use the resource, the user request must come from the source ID specified in the rule entry. If you do not specify this keyword, the rule entry applies to all sources.

UID

Specifies the user identification string of the user that the rule entry applies to. You can mask the UID.

UNTIL|FOR

Specify optional keywords. UNTIL specifies the date that the rule expires and becomes invalid. FOR specifies the number of days that the rule entry is valid. FOR is automatically translated into an UNTIL date when you compile and store the rule entry. For example, if you compile and store a rule entry on October 1, 1999 with FOR (10) specified, the rule entry automatically expires on October 10, 1999. If you do not specify UNTIL or FOR, the rule entry applies to all dates

Access Permissions

The next four keywords (ALLOW, LOG, PREVENT, and SERVICE) are related. They specify whether access to the resource is allowed if the rule environment matches a user-access environment.

These keywords determine how CA ACF2 for z/VM validates the user's request. Keyword meanings are:

ALLOW

The access is allowed.

LOG

The access is allowed, but logged.

PREVENT

The access request is prevented and logged. This is the default.

SERVICE (READ|ADD|UPDATE|DELETE)

SERVICE extends the level of control that a rule entry specifies. The SERVICE level is application-specific and controls access to the data.

How Do I Use Masking in Resource Rules?

You can also use masking in resource rules. You can mask the $KEY and UID parameters in resource rules, but you cannot mask the $TYPE control statement. If you mask the $KEY, you must create a resource typelist. A resource typelist is an in-storage index of $KEYs that CA ACF2 for z/VM searches to find a resource rule. Because resource typelists are loaded into memory, CA ACF2 for z/VM does not have to search the entire Infostorage database for the $KEY. It searches the typelist, and goes directly to the rule that applies. CA ACF2 for z/VM uses resource typelists to enhance performance when you mask resource names for a resource type.

An example of an AUTOLOG command resource rule is illustrated below.

```
$KEY(RSCS1)
$TYPE(ALG)
 UID(TLCAMS) SOURCE(GRAF-4A1) SHIFT(FIRST) ALLOW
 UID(TLCPJM) SOURCE(GRAF-4A1) LOG
 UID(TLC*)  SHIFT(FIRST) UNTIL(11/30/99) ALLOW
```

In the above example, the last line contains a masked UID. UID masking allows one rule entry to apply to several users. Masking reduces the number of rule entries that you need to write. The last line applies to all users (except for TLCAMS) that attempt to autolog the RSCS1 machine during the first shift time period if the access date is on or before November 30, 1999. The first rule entry (line 3) limits TLCAMS to autologging the RSCS1 machine during the first shift and only from GRAF-4A1. Because this rule entry is more specific that the last rule entry, any attempt by TLCAMS to autolog RSCS1 is validated by this rule entry.

When CA ACF2 for z/VM cannot find matching rule entry, access to the resource is prevented (by default) and logged. For example, if user TLCGLB tries to autolog RSCS1 on December 1, 1999, CA ACF2 for z/VM does not find a matching rule entry and denies the access.

How Do I Use CA ACF2 for z/VM Rules to Protect Resources?

Security administrators can use the ACF command and its subcommands to build, maintain, and delete resource rules from the Infostorage database. Issue the ACF command from the CMS READY prompt. You can also use the CA ACF2 for z/VM full-screen panels to process resource rules. You must have special CA ACF2 for z/VM privileges to issue these commands. Only security administrators and users with special privileges can write resource rules. You can find complete details on how to use the ACF subcommands to process resource rules in the *Administrator Guide*.

How Does CA ACF2 for z/VM Sort Rules?

The CA ACF2 for z/VM rule compiler converts the rule set input into a form usable by the rule interpreter when CA ACF2 for z/VM verifies it. The compiler also orders the rule according to the following criteria:

- UID patterns, from most specific to most general
- SOURCE parameters in alphabetical order, with “none specified” last
- SHIFT parameters in alphabetical order, with “none specified” last
- UNTIL dates, from earliest to latest.

The first active rule whose environment matches the actual access attempt determines the access permission.

CACIS and the CA Standard Security Facility (CAISSF)

The CA Standard Security Facility (CAISSF) controls and validates access to all system and application resources. Products or applications can ask CAISSF if access is allowed for a specific user. Products that use CAISSF provide sites with an unprecedented level of product integration. CAISSF provides such benefits as centralizing resource access control and reducing administrative costs for training and day-to-day administration.

How Do I Use CAISSF to Protect Resources?

CAISSF lets you:

- Specify that access to sensitive resources is logged
- Turn on event logging to see who is using a resource and determine if you can remove that resource from the system
- Scan the databases to determine which users have access to what resources
- Write customized reports against the audit data using CA-Earl.

For more information on how you can protect access to resources with CAISSF, see the *Administrator Guide.*

What is Shared File System?

Shared File System (SFS) lets CMS users use CMS files that do not reside on their minidisks. Instead, these files reside in hierarchical directories that are similar to the directories under MS-DOS.

SFS files physically reside in a database a SFS file server virtual machine owns. CMS users never link to the minidisks containing their SFS files. They simply use the access command to access these directories. The user works using the normal CMS command and macro set.

What is a Filepool?

Each SFS file server manages a collection of SFS files known as a filepool. There is only one filepool per SFS server machine. Every user must know the name of the filepool where the files they want to access reside.

How to I Access an SFS Directory?

Most CMS commands require you to refer to files as filename, filetype, and filemode. Normally, users access the directory where the file resides at a CMS filemode exactly as they do for a minidisk.

To access a file called MY DATA that is owned by TLCAMS in filepool PAYROLL, the user enters:

```
ACCESS PAYROLL:TLCAMS.B
XEDIT MY DATA B
```

Protecting VM Data Spaces

CA ACF2 for z/VM validation of data spaces occurs at PERMIT time, not actual access time. Data space rules must distinguish between READ and WRITE access to data spaces. You therefore need to use the SERVICE keyword of resource rules to distinguish between READ access and UPDATE access.

To protect data spaces, you must specify a new keyword of DSPACE(xxx) on the RESCLASS VMO record. The default resource type for DSPACE is DSP.

For more information on VM data space security, see the *Administrator Guide.*

Controlling CP Commands and Diagnose Instructions

You have already learned how you can write access rules that define who can access information and resources and how that access is controlled.

Command limiting rules also control access, but provide further flexibility in validating users and the information they can use on the system. Through command limiting rules, you can control which Control Program (CP)commands and operands a user can execute.

What Is a Command Limiting Rule Set?

A command limiting rule set contains the rules for a particular CP command. For example, one rule set can control the SPOOL command while another rule set controls the SET command. Each command limiting rule set consists of control statements and rule entries.

Control Statements

The control statements in a command limiting rule set include:

$KEY

Specifies the full name of the CP command that the rule set applies to.

$MDLTYPE

Associates the rule with a particular release of VM. It can also provide separate rule sets when systems share the same databases.

$MODE

Specifies the mode for command limiting validation. It provides another way of changing the system-wide mode for command limiting validation.

$NOSORT

Prevents CA ACF2 for z/VM from sorting the rule set.

$OWNER

Specifies the owner of the rule set for tracking purposes.

$USERDATA

Stores rule set information.

Rule Entries

Each rule entry specifies an environment where a CP command can occur. These parameters can indicate:

Command operands

Specifies the operands of the CP command being issued

UID

Specifies those users that can issue the CP command with the specified operands

UNTIL|FOR

Specifies any time limitation (you can specify the number of days or a date) when a user can issue the command and operands.

Each rule entry also specifies an access permission, which determines how CA ACF2 for z/VM handles an attempt to issue a CP command under a particular environment. These permissions are:

ALLOW

Allows execution of the CP command under the specified conditions

LOG

Allows execution but writes a record to log the attempt

PREVENT

Denies execution and logs the attempt.

Masking in Command Limiting Rules

Masking in command limiting rules reduces the number of rule entries that you need to write. The number of rule entries is reduced because you can write a rule to allow:

- A group of users to issue a particular CP command in a particular environment
- Particular users to issue a number of different combinations of CP command operands in a particular environment.

For example, the following rule set applies to users in the Payroll Department at the True Lock Company. This rule set lets those users issue the SPOOL command with the PRINT operand and one other operand of one to four characters. (Command limiting rules usually contain more than one rule entry.)

```
$KEY(SPOOL)
 PRINT **** UID(TLCPAY-) ALLOW
```

In this example, each asterisk can match any character including trailing blanks. The dash (—) represents any characters that completes the character string. In this case, UID(TLCPAY—) is the same as UID(TLCPAY*************) or UID(TLCPAY), because any specified UIDs are treated as if they were masked with a trailing dash.

Sample Command Limiting Rule Set

The following example illustrates a command limiting rule set for the CP SET command:

```
$KEY(SET) MDLTYPE(H50)
FAVORED – UID(AUTOLOG1) ALLOW
FAVORED – UID(OPERATOR) LOG
FAVORED – UID(*) PREVENT
PRIORITY – UID(AUTOLOG1) ALLOW
PRIORITY – UID(OPERATOR) LOG
PRIORITY – UID(*) PREVENT
SRM   – UID(MAINT) LOG
SRM   – UID(*) PREVENT
– UID(*) ALLOW
```

This example allows user AUTOLOG1 to issue the SET FAVORED and SET PRIORITY commands. The OPERATOR user ID can issue the SET FAVORED and SET PRIORITY commands, but they will be logged. MAINT can issue the SET SMR command, but it will be logged. By default, all other users are prevented from issuing the SET FAVORED, SET PRIORITY and SET SRM commands. The last rule entry allows users to issue all other operand combinations of the SET command without logging or preventing them.

You can determine the CP commands that CA ACF2 for z/VM validates. When a CP command is validated, command limiting rules determine who can issue the command, under what conditions the command can be issued, and with what operands the command can be issued.

What is a Diagnose Limiting Rule Set?

A diagnose instruction is a routine that lets any virtual machine communicate directly with CP. A diagnose limiting rule set contains the rules for a particular VM diagnose instruction. For example, one rule set can control the use of diagnose 14 while another rule set can control diagnose 84. Each diagnose limiting rule set consists of control statements and rule entries.

Control Statements

Each rule entry specifies an environment where a diagnose instruction can occur. These parameters can indicate the following:

UID

Specifies who can issue the diagnose instruction named in the $KEY control statement

UNTIL|FOR

Specifies any time limitation (you can specify the number of days or an ending date) when a user can issue the diagnose instruction.

Each rule entry also specifies an access permission that determines how CA ACF2 for z/VM validates an attempt to issue the diagnose instruction. These permissions include

ALLOW

Allow execution of the instruction under the specified environment

LOG

Allow execution but write a record to log the attempt

PREVENT

Deny execution and log the attempt.

Sample Diagnose Limiting Rule Set

The following example shows a diagnose limiting rule set for the x'14' diagnose instruction:

```
$KEY(DIAG0014)
 UID(TLCOPR) ALLOW
 UID(TLCTEC) ALLOW
```

These rule entries let users TLCOPR and TLCTEC issue the diagnose instruction with diagnose code x'14'. You can determine the diagnose codes that CA ACF2 for z/VM validates. When a diagnose code is validated, diagnose limiting rules determine who can use the diagnose instruction.

Masking in Diagnose Limiting Rules

In diagnose limiting rules; you can mask the UID values just as you do in access rules and command limiting rules. Masking in diagnose limiting rules reduces the number of rules you need to write to protect diagnose instructions. The following sample rule set applies to users in the Payroll department and the Personnel department at the True Lock Company.

```
$KEY(DIAG0014)
 UID(TLCPAY–) ALLOW
 UID(TLCPER–) LOG
```

In the above example, all users in the Payroll Department can issue the x’14’ diagnose instruction. All users in Personnel can also issue the x’14’ diagnose. However, every time a user in Personnel issues the diagnose, the attempt is logged and CA ACF2 for z/VM creates an SMF record.

Providing Additional Controls

Access rules and resource rules contain parameters that let you control the access environment. Parameters such as SHIFT and SOURCE let you restrict access to a particular shift or input device. If you specify SHIFT(NORMAL) in a rule, you are instructing CA ACF2 for z/VM to limit access to that data or resource according to the shift record named NORMAL.

Entry Records

Entry records are class E records that identify input devices. They are stored on the Infostorage database. Every logonid that is validated by CA ACF2 for z/VM has a physical input source designation, such as a specific terminal or remote cluster associated with it. For example, the standard terminal IDs used by VM can be GRAFxxxx and LINExxxx.

Physical source names are subject to random changes, such as recabling of terminals or swapping terminals for repair. This makes them a poor choice for access and resource rules and for limiting logonid use. CA ACF2 for z/VM translates the physical source name to a logical source name that is up to eight characters long. It does this by looking up a corresponding record on the Infostorage database having a type code of SRC (for source entry records) and identified by the physical source name. It then uses the first data item as the logical source name.

The logical source name is composed of location information, such as Terminal Room #1, Terminal #3 (designated as TR1T3), or a building and office number designation, such as BDG2RM32. They can be more oriented to the way a security administrator would think about them than the way an operating system views them.

Source Group Support

Source group records are type SGP records. They identify groups of input devices. In other words, entry records control the source from which certain users can access the system. For example, you might limit certain logonids access to the system from a specific set of remote terminals or a set of interactive terminals. Unrestricted security administrators can create and update entry records. Full scope auditors can display these records.

Creating Entry Records

To limit a logonid to specific input devices, use the ACF subcommands to set the SOURCE field of the logonid record to a logical source ID or a source group name. To make the input source part of the environment specification for resource or access rules, specify the name of the source record in the SOURCE parameter.

See the *Administrator Guide* for more information on how to define entry records.

Scope Records

You can grant special privileges to a user in their logonid record. You can restrict these privileges with scope records. Scope records have a record class of S and a type of SCP. They are stored on the Infostorage database. They can limit a user's ability to access any or all of the following:

- Files and the access rules for those files
- Logonid records
- Infostorage records, including resource rule sets, entry records, scope records, shift and zone records, VMO records, and ACFSERVE privilege records. (We explain VMO records and ACFSERVE privilege records in the Defining System Options section.)

A scope record does not become effective for a given user until you specify the name of the scope record in the SCPLIST field of that user's logonid record.

The special logonid record privileges are designed to promote separation of function. The most often used privileges are:

SECURITY

Gives a user authority to write access rules for all data, to modify all Infostorage database records, and to modify most fields of the logonid record.

ACCOUNT

Gives a user authority to insert, list, change, and delete logonid records.

AUDIT

Gives a user authority to display all access rules, all Infostorage database records, and all logonid records. An AUDIT user can display records, but cannot change or create records.

A scope record is from 1 to- 44 characters long and contains specific information depending on the type of record. A scope record name is from one- to eight-characters long and is specified in the SCPLIST field of the logonid record. The scope record name specified in the SCPLIST field points to the associated scope record on the Infostorage database. A scope entry that limits a user's authority over logonid records can contain the actual logonid and UID or masked logonid and UID.

For example, in a decentralized environment, the Director of the Financial Division for the True Lock Company may have the ACCOUNT privilege to create and maintain logonid records for his division, but only his division. If the UID format at the True Lock Company is defined as company (TLC), division (FIN), and the user's logonid, a scope record for the Finance director would specify UID(TLCFIN—) and the associated logonid (or masked logonid). Logonid entries limit him to only the financial group of logonid records. You can also limit a SECURITY user to particular groups of filenames for rule writing.

The presence of any SCPLIST value, regardless of the actual entries in the related scope record itself, limit that user. A security administrator whose scope record contains only entries that govern logonid records is limited, even though his scope record contains no entries for access rules. A logonid record must match both LID and UID SCPLIST entries for that user to modify the record. If a user has SCPLIST, you must specify all matching scope lists for that user.

If there is no scope definition present (LID, UID, or DSN), no records match.

Note: We will not support DSNSCOPE, LIDSCOPE, and UIDSCOPE in future releases. We recommend that you use SCPLIST to impose restrictions.

A restricted security administrator or account manager has limited power to perform specific CA ACF2 for z/VM functions. An account manager or security administrator with no scope list is unrestricted. The SCPLIST determines the user's restriction.

Creating Scope Records

Security administrators can create scope records and specific scope entries by using the ACF command and its subcommands under the SCOPE setting in a way similar to the creation of other CA ACF2 for z/VM records. Use the INSERT, LIST, CHANGE, and DELETE subcommands of the ACF command to create and maintain scope records. You can find complete details on how to use the ACF subcommands to process scope records in the *Administrator Guide.*

Shift Records

CA ACF2 for z/VM lets you specify time and shift controls in the SHIFT field of the logonid record. You specify entries that define the specific shift on the Infostorage database. You can grant a user system access, data access, and resource access for a specific time of day, days of the week, or actual dates (specified as mm/dd/yy, yy/mm/dd, or dd/mm/yy, depending on your site’s option). Shift records have a record class of T and the type code SFT.

The SHIFT field in the logonid record contains the name of the shift record in the Infostorage database. Fields in the shift record let you specify the times and dates that CA ACF2 for z/VM should use to allow or deny access.

The DAYS field can contain days of the week (MO, TU, or WE) and dates (10/10/99, 12/20/99) when a user can access the system. The NDAYS parameter indicates specific days or dates not allowed. For example, you can name a shift record NORMAL and define DAYS as (MO,TU,WE,TH,FR). This record allows a user access during the regular work week. With NDAYS, you can specify 12/25/99 to prevent access on Christmas Day, regardless of whether it falls on a weekday.

Similarly, the TIME and NTIME fields specify the allowed and prevented times when a user can access the system. For example, the same shift record NORMAL specifies TIME(0800-1700) to allow access during the standard business day, with NTIME(1200-1300) specified to deny access during the lunch period. Or, you could define the same time frame by specifying only the TIME field as TIME(0800-1200,1300-1700). You must specify the DAYS field to specify TIME or NTIME.

The LOGSHIFT field of the logonid record grants system access to a user outside the shift specified in the logonid record, but this access is always logged. If you set the LOGSHIFT field but do not specify a shift name, CA ACF2 for z/VM ignores the LOGSHIFT field. For example, a programmer whose shift is defined as NORMAL but who must access the system on a weekend can be given access with the LOGSHIFT privilege. The LOGSHIFT privilege applies to a user’s access to the system and not to shift controls in resource rules or access rules.

Access rules can contain the SHIFT parameter to indicate the shift when a rule applies. In this way, you can limit access to specific data to particular days and times. Shift records can also protect resources. For example, you can govern group logons with a shift record that limits the hours of access to only morning or afternoon.

Creating Shift Records

Security administrators can create shift records using the ACF command and its subcommands under the SHIFT setting in a way similar to creating other CA ACF2 for z/VM records. Use the INSERT, LIST, CHANGE, and DELETE subcommands of the ACF command to create and maintain shift records.

A shift record name is from one- to eight-characters long. You specify the shift record name in the SHIFT field of the logonid record to point to the associated record in the Infostorage database. You can find complete details on how to use the ACF subcommands to process shift records in the *Administrator Guide*.

Zone Records

CA ACF2 for z/VM lets you create and store zone records on the Infostorage database. Zone records have the record class of T and type code of ZON. They specify time zones using an offset in hours and minutes to the local CPU time (hhmm). The ZONE field of the logonid record contains the name of the zone record on the database that applies to that user.

For example, a site in Chicago that is running under local time defines a zone called EST for Eastern Standard time. Because Eastern Standard time is one hour ahead of Chicago time, the EST zone record contains an adjustment of +0100 to offset the local time ahead by one hour. A zone record for Pacific Standard time named PST is also defined and contains an adjustment of —0200, to subtract two hours from local time.

You can use the ACF command and its subcommands to create and maintain zone records. See the *Administrator Guide* for complete details on how to use ACF subcommands to process zone records.

Defining System Options

Most CA ACF2 for z/VM system-wide options are defined in VMO records and ACFSERVE privilege records that are stored on the Infostorage database. Other system-wide options of a more technical nature are defined in the CA ACF2 for z/VM Field Definition Record (ACFFDR) macros.

System Options in CP Nucleus

CA ACF2 for z/VM generates a number of options you need before the ACF2 service machine is activated into the CP nucleus. The options are described in the VMXAOPTS macro in module HCPAC0. These options can also be set in the ACF2VM config file. The options maintained in CP are:

- The name of the CA ACF2 for z/VM service machine
- A minidisk address to hold the CA ACF2 for z/VM startup options across IPLs and whether to prompt the operator for RESTART IPLs
- The size of the CP command and diagnose limiting cache
- The list of autologged users allowed on the system before CA ACF2 for z/VM is active
- The list of user IDs allowed on the system when VM is brought up without CA ACF2 for z/VM active
- The list of user IDs allowed to update the CA ACF2 for z/VM databases when VM is running without CA ACF2 for z/VM active
- Whether VM directory passwords are required when VM is brought up without CA ACF2 for z/VM active
- IUCV and VMCF protective parameters
- CMS protective parameters
- Unit record logging activation for C2 sites
- Minidisk directory password handling options
- Names of the CP resident CA ACF2 for z/VM exits
- Hacker trap activation
- Names of the primary and alternate message repositories and names of available languages
- Name of the CMS filename and filetype translation table.

VMO Records

VMO records define system-wide options. You can change the system options you define in VMO records with the ACF subcommands. No reassembly is required. Some of the VMO records CA ACF2 for z/VM provides are listed below. Use VMO records to specify the following system-wide options:

- The format of infostorage application records
- The time for automatic backups, and who should be notified when CA ACF2 for z/VM takes a backup

- Which CP commands are protected and the CA ACF2 for z/VM mode used during validation
- Which diagnose instructions are protected and the CA ACF2 for z/VM mode used during validation
- The exits loaded by the service machine
- The VM options available to the system
- The password options and controls, such as the maximum number of password attempts allowed in a session, in a day, and minimum password length
- The resource rule keys and the size of the cache
- The options important to access rule and resource rule maintenance and where resident resource rule directories are built
- The text of a warning message to display at the terminal or job log when CA ACF2 for z/VM is in WARN mode and a security violation occurs.

You can find complete descriptions of all the VMO records and how to define them in the *Administrator Guide.*

System Options in the ACFFDR

The CA ACF2 for z/VM Field Definition Record (ACFFDR) is an assembly of a module that defines the fields in the logonid record and specifies other system options not defined in the VMO records. Detailed information on how to specify these options is included in the *Installation Guide*. Macros in the ACFFDR define:

- The name of each field in the logonid record.
- The filenames for each group of CA ACF2 for z/VM databases. Your site can define up to 16 groups of databases.
- Users who can use the System Request Facility (SRF).
- The logonid record fields that make up the user identification string (UID).
- VM attributes.

ACFSERVE Privilege Records

The ACFSERVE command is controlled by structured infostorage records for ACFSERVE privilege records.

ACFSERVE privilege records let you define your own privilege attributes for the ACFSERVE commands.

ACFSERVE privilege records control the ACFSERVE commands. In these records, you must define which users can issue certain ACFSERVE commands and what special privileges they need in their logonid records to issue them. Most ACFSERVE commands require a user to have the SECURITY and ACCOUNT privileges, but some also require the AUDIT privilege.

ACFSERVE privilege records also allow you to specify if a user needs a combination of these privileges. For example, for the ACFSERVE RELOAD FDR command, an authorized user must have **both** SECURITY and ACCOUNT in his logonid record. For the ACFSERVE ARCHIVE SMF command, an authorized user can have **either** the SECURITY or ACCOUNT privilege.

You can also specify whether an authorized logonid can be scoped. Finally, you can also specify whether CA ACF2 for z/VM creates an SMF record when the specified user issues the given ACFSERVE command.

For complete information on defining user privileges for ACFSERVE commands, see the *Administrator Guide.*

User Exits

There are two types of installation exits in VM:

- CP-resident routines
- Service machine resident routines.

This section provides general descriptions of these exit points. You can find complete descriptions of each of the CA ACF2 for z/VM user exits in the *Systems Programmer Guide.*

CPResident Routines

There are certain installation subroutines that you can add to CA ACF2 for z/VM (at provided exits) that supplement certain routines. The exits are specified in the CP module HCPAC0 by the VMXAOPTS macro, or in the ACF2VM CONFIG file.

The CP-resident routines are:

- Data set/program prevalidation exit
- Data set/program violation exit
- Data set/program postvalidation exit
- Account prevalidation exit
- Account postvalidation exit

- Unresolved logical device creation exit
- Logon authentication exit.

Service Machine Resident Routines

You can add certain installation subroutines to CA ACF2 for z/VM at provided exits. You must specify these service machine exits in the EXITS VMO record. Each routine must be a self-contained CMS module.

The service machine resident routines are:

- NEWPXIT (New Password Exit)
- PENCRYP (User Password Encryption Routine).

For more information on these exits, see the *Systems Programmer Guide.*

Administering CA ACF2 for z/VM

You can use the ACF command and its subcommands to process CA ACF2 for z/VM rule sets and records. Issue the ACF command from CMS, or use the full-screen panels that we provide to process CA ACF2 for z/VM rules and records. This chapter explains the basic operation of the ACF command and its subcommands, the CA ACF2 for z/VM full-screen feature, and the ACFSERVE command.

Issuing the ACF Command

When the CMS READY prompt appears, you can issue the ACF command:

```
READY; T=1.23/2.34 14:36:00
```

```
acf
```

After the system responds with ACF, you can process CA ACF2 for z/VM rule sets and records.

```
READY;T=1.23/2.34 14:36:27
acf
 ACF
```

ACF Command Settings

After issuing the ACF command, you must establish the ACF command setting. This setting determines the type of CA ACF2 for z/VM record you can process. The ACF command has the following settings:

Setting	Type of CA ACF2 for z/VM Record Processed
ACF	Logonid records
CMDLIM	Command limiting records
CONTROL(VMO)	VM system options
CONTROL(ACFSERVE)	ACFSERVE command privileges
DIAGLIM	Diagnose limiting records
ENTRY	Entry records
LID	Logonid records
MODE|NOMODE	Session setting
PROFILE	Profile records
MODEL	Syntax models
RESOURCE	Resource records
RULE	Access rules
SCOPE	Scope records
SHIFT	Shift or zone records

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

For some settings, you must be more specific about the type of CA ACF2 for z/VM rule set or record you want to process. With entry records, for example, specify the setting and a three-character type code:

```
ACF
set entry(src)
```

The screen above illustrates that you would be processing source entry records.

ACF Subcommands

After you have entered the ACF command and have established the appropriate ACF command setting, you can issue ACF subcommands. These subcommands do the actual processing of CA ACF2 for z/VM rule sets and records.

The ACF subcommands include the following:

ACFSERVE	DELETE	QUIT
CHANGE	END	SET
CMS	EXEC	SHOW
COMPILE	HELP	STORE
CP	INSERT	TEST
DECOMP	LIST	XEDIT

For a given command setting, a subcommand processes a certain type of CA ACF2 for z/VM record or rule set. Not all subcommands are valid in all settings. See the *Administrator Guide* for more details.

Using the CA ACF2 for z/VM FullScreen Feature

The CA ACF2 for z/VM full-screen feature is a panel-driven feature that lets you select the panel for functions you want to perform. If you have the necessary authority in your logonid record, you can create logonids for new users, maintain access rules and resource rules, delete users, and change user privileges. Fill in the appropriate blanks, execute the screen, and the job is performed. Help is available for each screen and for each field on the screen.

To use the full-screen feature, ACFFS from CMS. When you press Enter, CA ACF2 for z/VM displays the Primary Option Menu. From this menu, you can select the option you want to use, such as the one to write access rules (2), or the one to generate reports (6).

As you use the full-screen feature, you will see a series of screens. These screens let you select the administrative function you want to perform. However, CA ACF2 for z/VM never lets you perform functions that you do not have the authority to perform. For example, privileges that you cannot grant never appear on the screen and you cannot move your cursor into the field for that privilege.

Getting Help

While using the full-screen feature, place the cursor on any field and press your HELP PF key (the default is PF1, but you can define your own PF keys). CA ACF2 for z/VM provides you with online assistance for that screen. You can also place the cursor on any field and press your HELP PF key and CA ACF2 for z/VM displays additional information about that field.

The Primary Option Menu

The screen below illustrates the Primary Option Menu screen you see when you access the full-screen feature:

```
M9PA-0000    CA ACF2 for z/VM Primary Option Menu       CA ACF2 for z/VM
OPTION====>_____________________________________________________
                                                          TIME 10:22


     0 Change options or PF keys

     1 User Identification

     2 Data Access Control

     3 Resource Control

     4 Source, Shift, Scope, and Zone Maintenance (not available)

     5 Display System Options (not available)

     6 Audit Reports


PF1=Help       2=Print       3=Quit       4=Return       5=       6=
PF7=           8=            9=           10=            11=      12=Retrieve
```

OPTION ===>

Enter one of the following digits for the type of function you want to perform.

0 Change options or PF keys

Displays the current values of the PF keys and lets you change them.

1 User Identification

Lets you maintain logonids (define new users, change an existing logonid, and so on).

2 Data Access Control

Lets you process access rules.

3 Resource Control

Lets you process resource rules.

4 Source, Shift, Scope, and Zone Maintenance

Not available in this release for future use.

5 Display System Options

Not available in this release for future use.

6 Audit Reports

Lets you run CA ACF2 for z/VM reports.

When you select an option from this menu, CA ACF2 for z/VM displays the panel you need to maintain logonids, access rules, resource rules, and so on. These panels are fully described in the *Administrator Guide,* with the exception of the report panels, which are explained in the *Reports and Utilities Guide.*

Using the ACFSERVE Command

With the ACFSERVE commands, authorized users can look at and move files and other data maintained through the ACF2 service machine. Only users who have the proper CA ACF2 for z/VM privileges in their logonid record can use the ACFSERVE commands. These privileges are AUDIT, ACCOUNT, SECURITY, or a combination of the three.

You can use the ACFSERVE commands to:

- Archive, display, and switch SMF files
- Backup the CA ACF2 for z/VM databases
- Force checkpoints
- Remove a resource type
- Update the databases while CA ACF2 for z/VM is not active
- Terminate database updates while CA ACF2 for z/VM is not active
- Display information on batch machines, databases, group user IDs, source, and infostorage records
- Reload tables, command and diagnose limiting rule sets, infostorage records, the ACFFDR, VMO records, ACFSERVE privilege records, resource types, access rule sets, and shift information
- Display CA ACF2 for z/VM release number, level, and operating status
- Lower password violation counts

- Restart the ACF2 service machine
- Set the SYSID.

See the *Administrator Guide* for complete information on all the ACFSERVE commands, syntax, and return codes.

Centralized and Decentralized Security Administration

CA ACF2 for z/VM enables your site to select centralized or decentralized security administration. You can combine several CA ACF2 for z/VM features to create a centralized or decentralized environment.

In a centralized environment, one security administrator may be responsible for creating and maintaining all the CA ACF2 for z/VM rules and records. To decentralize this environment, the security administrator can delegate administrative duties in the following ways:

- He can create several scoped account managers who have the ACCOUNT field specified in their logonid records. He then creates scope records for each of these account managers so that they can create or modify only logonids for their groups.
- He can grant the scoped security administrator the ability to delete, update, and create rules. By specifying the %CHANGE parameter and a UID mask in the rule sets that control access to the group's data, the scoped security administrator can change the rule sets.
- He can further decentralize the environment by specifying the %RCHANGE parameter and a UID mask for the users that need to add, delete, or change specific rule entries in a rule set. The %RCHANGE privilege enables those users to change the rule entries.
- Also, CA ACF2 for z/VM can force these scoped security administrators to perform changes to CA ACF2 for z/VM records only during a specific shift or from a particular entry source.

How Does CA ACF2 for z/VM Monitor Activity?

CA ACF2 for z/VM uses SMF-type files to record all CA ACF2 for z/VM loggings. These files are maintained and managed by the ACF2 service machine. This SMF data is used as input for all CA ACF2 for z/VM reporting and several utilities. You can monitor the status of and manage SMF files with the ACFSERVE commands.

When Does CA ACF2 for z/VM Create SMF Records?

CA ACF2 for z/VM creates SMF records under the following circumstances:

- Each time you try to log on, but are denied access for any reason
- Each time you are allowed to log on outside of your specified shift through the LOGSHIFT privilege
- Each time you try to access data and CA ACF2 for z/VM aborts your request
- Each time you try to execute a CP command or diagnose and your request is aborted by CA ACF2 for z/VM
- Each time you attempt to access a data set or protected resource (successfully or unsuccessfully) while the TRACE attribute is turned on in your logonid record
- Each time you add, modify, or delete a logonid record
- Each time you add, replace, or delete an access rule set (the new record or deleted rule set is indicated on the logging)
- Each time you add, replace, or delete a record on the Infostorage database
- Each time a record is processed by the recovery utility
- Each time you issue an ACFSERVE command
- Each time you issue a DIRMAINT command (if this support is installed)
- Each time any rule with the LOG option requests a recording.

Report Generators

You should carefully scan the CA ACF2 for z/VM reports to ensure that your computer system remains secure. (Look for a mistake in the specification of an access rule, or an attempt to guess the password of an authorized user, and so on.) These reports produce information that highlights these events, but someone must be looking at the reports to take appropriate action.

Also, you can limit the output of a particular report according to the privileges and restrictions of the specific logonid that is running the report. See the *Reports and* Utilities *Guide* for a more detailed explanation of this option.

CA ACF2 for z/VM reports format and edit information from SMF records and CA ACF2 for z/VM databases to perform these tasks:

- Report recovery of the CA ACF2 for z/VM databases.
- Provide loggings of each ACFSERVE command that was issued. The ACFSERVE Command Tracking Log (ACFRPTCT) report identifies each type of ACFSERVE command and the logonid of the user who issued it.

- Format the data set violation, logging, trace records, program violations, and logging records.
- Report modifications made to resource rule sets and CA ACF2 for z/VM records on the Infostorage database.
- Report and edit information about high-level indexes and their rules, including changes over an historical period.
- Report modifications made to logonid records, including before and after values for changed fields.
- Select the CA ACF2 for z/VM records from the SMF data sets and put them into separate files for further processing by other report generators.
- List password violations and LOGSHIFT system accesses.
- Report modifications made to access rule sets.
- Format the resource violation, logging, and trace records that apply to resources.
- Report and edit information on users (logonid records) according to criteria you specify.
- Provide cross-reference information that ties users to access rules and resource rules, such as listing which logonids have access to which data sets.

You can read more about the CA ACF2 for z/VM reports and sample report output in the *Reports and Utilities Guide.* Besides these reports, CA Earl™ is shipped with CA ACF2 for z/VM. You can find complete details for using CA Earl™ in the *Reporting with CA Earl™ Guide.*

Utility Programs

CA ACF2 for z/VM provides utilities to perform the following functions:

- Erase a tape volume
- Recover the CA ACF2 for z/VM databases by merging SMF update records with the backup CA ACF2 for z/VM databases
- Create logonids from the VM directory
- Merge the CA ACF2 for z/VM databases
- Restore the CA ACF2 for z/VM databases from backup files
- Copy CA ACF2 for z/VM CMS databases
- Convert CA ACF2 for z/VM CMS databases to VSAM.

For more detailed information on these and any of the other utilities, see the *Reports and Utilities Guide.*

Chapter 3: Planning the Implementation

This chapter contains the preliminary steps you should consider when planning for total CA ACF2 for z/VM security. Information in this chapter includes:

- Organizing for security
- Appointing a Security Administrator (SA) to be coordinator of security
- Appointing an Implementation Team (IT) to implement security
- Preparing an implementation schedule
- Distributing CA ACF2 for z/VM documentation
- Identifying security policies and goals
- Identifying the operating environment at your site
- Selecting options for CA ACF2 for z/VM to fit your needs.

This section contains the following topics:

Organizing for Security

The most commonly used approach in organizing sites for CA ACF2 for z/VM is to:

- Appoint someone as the Security Administrator (SA). The SA manages and coordinates the information security program for the site.
- Establish an Implementation Team (IT) to help the SA plan, install, and implement CA ACF2 for z/VM and any related security practices and procedures.
- If your site's size and activity warrants, the SA should have the option to assign more people to the information security function (full- or part-time) to work for or with the SA. These assignments can be on a centralized or decentralized basis, according to your requirements.

Appointing a Security Administrator

The SA serves as the central coordinator for information security and represents a permanent staff position. The SA's responsibility encompasses all phases of implementing CA ACF2 for z/VM (that is, initial planning, migrating to full security, and ongoing administrative activities). Most of the CA ACF2 for z/VM effort occurs early, during the planning and implementation phases (usually the first few months). After CA ACF2 for z/VM controls have been integrated into production systems, the SA must make an ongoing effort to properly enforce security measures. Ongoing administrative functions for CA ACF2 for z/VM include:

- Updating CA ACF2 for z/VM databases
- Reviewing CA ACF2 for z/VM reports
- Following up on questionable acts or possible violations (based on these reports).

These administrative functions and responsibilities can be centralized or decentralized. One way to decentralize the administrative responsibilities is to authorize people throughout the organization to fill out forms that request changes to logonid records, access rules, or other CA ACF2 for z/VM security controls. You can centralize the validation and updating function by requiring that requests be forwarded to the SA, who performs the updates.

Whether CA ACF2 for z/VM administration is centralized or decentralized (and to what degree) depends on your size, structure, and unique needs. CA ACF2 for z/VM lets you decide how administration is handled, both initially and on a continuing basis.

Planning also requires that you determine the scope of authority for various users (whether they should be allowed to update logonid records and access rules). In a decentralized environment, multiple SAs can have jurisdiction over limited groups of users, data, or resources. These limitations are imposed through CA ACF2 for z/VM scoping features. See the Administrator *Guide* for details.

The Security Administrator's Workload after Implementation

The total workload of the SA after CA ACF2 for z/VM implementation depends on:

- The volume of work on the system.
- The degree of centralization or decentralization of CA ACF2 for z/VM administrative duties (for example, how many users have the SECURITY or ACCOUNT privilege).
- Centralization or decentralization of responsibilities and authorization functions.
- Site commitment to data security. This commitment is reflected in areas such as the level of CA ACF2 for z/VM rules, follow-up actions taken on violation attempts, and whether you are using special interfaces.

Activity levels and the factors mentioned above determine the person best suited for this job. Although an SA does not have to be a programmer or computer operator, a SA should have some data processing expertise. The SA normally chairs the Implementation Team and:

- Schedules meetings with other members of the team
- Performs the majority of the coordination functions
- Is instrumental in selecting CA ACF2 for z/VM system options.

After CA ACF2 for z/VM is installed, the SA coordinates administration of the system. The SA should have a relatively autonomous position in the organization. This allows for independent and unbiased decisions and enforces site policies and rules. A low-level position inside the data processing department is not independent. The SA's position should not be in the EDP audit area because EDP auditors must independently audit the security system and its implementation and administration. Commitment to data security is vital to achieve a standard level of protection and security enforcement.

Appointing the Implementation Team (IT)

The primary function of the Implementation Team (IT) is to properly implement CA ACF2 for z/VM and related information security systems and procedures. This is a limited function. Most of the work occurs during the planning and implementation phases. Often, the team even disbands after CA ACF2 for z/VM has successfully been implemented and is functioning in ABORT mode. However, many sites choose to retain the team and have meetings periodically to review the system, reconsider options, and evaluate overall security measures. After CA ACF2 for z/VM is active, you will be in a better position to determine whether the IT should remain intact.

The activities of the Implementation Team include:

- Establishing an implementation schedule
- Assigning responsibilities for each activity
- Reviewing and selecting the CA ACF2 for z/VM options to tailor the system according to local policies and requirements
- Monitoring the progress of the implementation.

This team can also be useful as an ongoing Information Security Committee to assist the SA in identifying security policies and enforcing or eliminating these policies at different levels. A similar security committee might already exist at your site. It can even serve as the basis for an CA ACF2 for z/VM Implementation Team.

The Implementation Team normally consists of the SA (as chairperson) and three to eight other persons. They usually represent areas such as:

Systems Maintenance

Usually the IT representative is the systems programmer responsible for installing and maintaining CA ACF2 for z/VM. This person should be familiar with VM, the VM Control Program (CP), the Conversational Monitoring System (CMS), SYSGENS, and related areas.

Data Center Operations

Someone from operations who is familiar with current naming conventions, production schedules, and normal operations maintenance.

User Support Services

A representative to present the user's point of view and provide communication between the technical and nontechnical personnel.

User Groups

Representatives from these groups (for example, the accounting department) might be included on the IT where user support services people are not available to represent the user's point of view.

Data Security Officers

If database administrators (DBAs), physical security personnel, or other personnel are already active in the data security area, they can often provide valuable input on current use and future needs of data security.

EDP Audit

A representative from the EDP audit group can help define audit concerns for internal controls and their auditability. An auditor can often suggest why you should use certain options for acceptable levels of control, accountability, and auditability.

Upper Management

Rather than have upper management attend all the committee meetings, the SA can represent upper management. If this arrangement is inadequate, the committee can prepare recommendations to forward for action by a higher group. This may be required if corporate security policies must be clarified or established. Either arrangement can work smoothly, as long as communication channels are open.

Preparing the Implementation Schedule

An early function of the IT is to establish a preliminary CA ACF2 for z/VM implementation schedule. This may only include the major points, such as the test IPL (Initial Program Load), the production installation and IPL, and migration to ABORT mode. As the team reviews the CA ACF2 for z/VM documentation and identifies additional tasks it needs to perform, add them to the schedule. A good implementation schedule includes a detailed list of tasks, target completion dates for each, and the name of the person responsible for completing that task. The IT should monitor the progress in each area, resolve conflicts, and periodically update the schedule as necessary.

We have provided an example of a good implementation schedule below. We have also provided general time frames for chronological reference but you should use specific completion dates wherever possible. However, we have not included responsibility assignments in this example because they vary from site to site, depending on the type and number of persons available for the project and the size of the task. Add more detailed tasks as you define them.

Important! This is an CA ACF2 for z/VM implementation schedule, not an installation schedule. The installation of CA ACF2 for z/VM is merely the activation of the code on your computer. The implementation of a security system includes much more, whether the tool to help automate the security process is CA ACF2 for z/VM, or some other product. That is why the planning process is so important and why you should use a team of knowledgeable people rather than the one or two persons needed to just install the package.

Sample CA ACF2 for z/VM Implementation Schedule

The following is an example of an implementation schedule.

Week	Task
Week 0	Order CA ACF2 for z/VM after preliminary requirements and product review.
Week 1	Appoint a Security Administrator.
	Establish the Implementation Team.
	Hold an organizational meeting.
	Distribute CA ACF2 for z/VM documentation to team members.
	Suggest that the SA and systems programmers attend an CA ACF2 for z/VM training class. (This is recommended, but not mandatory.)
Week 2	Establish a general timetable and responsibility assignments.

Week	Task
	Identify existing government, corporate, industry, and local security policies, goals, and objectives.
	Identify local conditions or operating environment, such as naming conventions.
	Identify system users and user groups.
Week 3	Make preliminary option selections.
	Refine the schedule and responsibility assignments.
	Perform a test install of CA ACF2 for z/VM on a second level VM test system.
Week 4	Finalize schedule and responsibilities.
	Refine option selections.
	Perform initial training, if needed(administrators or computer operators).
	Prepare for a test IPL of target system write or tailor any local exits, check option selections, select sample users, jobs, rules, or prepare test procedures.
Week 5	Test IPL (in LOG mode or RULE mode).
	Review results, including report outputs.
	Modify option selections as necessary.
	Modify procedures and operator instructions as necessary.
	Test backup and recovery procedures. Prepare to install CA ACF2 for z/VM on production systems.
	Handle additional training and announcements.
Week 6	Install CA ACF2 for z/VM on first level VM production system (LOG mode or ABORT mode on selective rule sets using RULE mode).
	Establish initial users.
	Write, compile, and store initial rules.
	Closely monitor reports.
	Retest backup and recovery procedures.
Weeks 7-12	Readjust system options and exits if necessary.
	Complete user definitions.
	Complete rules.
	Continue to closely monitor reports.

Week	Task
	Test maintenance procedures.
	Migrate remaining portions of the system to WARN mode and finally to ABORT mode.

You can phase in the CA ACF2 for z/VM modes (QUIET, LOG, WARN, RULE, and ABORT) on a rule set basis with the RULE setting in the MODE operand of the OPTS VMO record. See the “Converting to Full Security,” chapter for details. The above schedule provides guidelines for the timing of these implementation activities. This timing can vary significantly, depending on the circumstances at your site. Many of these activities can occur concurrently or can proceed at different rates independently. For example, the technical installation and IPL of the target system can occur before or after the less technical activities listed in Weeks 4 and 5. Detailed information on some of these topics, such as what should be done to perform the install or during the first IPL, is presented in later chapters of this guide.

Distributing CA ACF2 for z/VM Documentation

Distribute CA ACF2 for z/VM documentation to the Implementation Team and other selected personnel as early as possible. Not all team members need all the guides. Determine the applicable guides for each group and distribute copies accordingly. Use the table below as a guide to determine which groups should receive selected CA ACF2 for z/VM documentation.

Team Members	Guide Name
All IT members and EDP Audit IT representatives	Administrator Guide Command and Diagnose Limiting Guide General Information Guide Installation Guide Implementation Planning Guide Report and Utilities Guide
Systems Programming IT representatives	Messages Guide System Programmer’s Guide
Other personnel, such as upper management or special users	Selected guides, as appropriate.

During later phases of implementation, additional documentation can be appropriate. For example, provide operations with the *Messages Guide* (or those portions applicable to your site) before the first IPL. Users also need copies of the *Administrator Guide* and the *General Information Guide*.

You should also provide users with copies of all new and revised documentation. That way, all necessary personnel have the latest updates and outdated copies of documentation are not left in use.

Identifying Security Policies, Goals, and Objectives

The IT should have some idea of your company data security goals and objectives before implementing CA ACF2 for z/VM. CA ACF2 for z/VM is a tool to implement security policies, automate policy enforcement, and help the company achieve its goals. If you do not define these policies, it is very difficult for the IT to choose appropriate CA ACF2 for z/VM options and proceed successfully through implementation.

Various factors influence your policies and objectives. These include, but are not limited to, the following areas. Review them for applicability to your site:

Government Regulations

The U.S. federal government has a number of regulations that can impact data security requirements. These include the Privacy Act, Foreign Corrupt Practices Act, Securities and Exchange Commission and other agency regulations, and various other accounting and reporting requirements. There are also similar regulations in other countries, such as national privacy acts, transborder data flow regulations, and accounting and taxing regulations. Additionally, take into account any state, provincial, or other local regulations. Government regulations for government agencies are often even more encompassing.

Legal Requirements

Many legal requirements are tied to government regulations, such as requirements about controls over Electronic Fund Transfers. Others may be contractual, such as union agreements (unauthorized employee record accessibility). You may be subject to other requirements if you operate as a service bureau and have contractual agreements with customers about the confidentiality and protection of their data and programs.

Industry Practices and Agreements

Some industries share certain data, while many highly competitive industries tightly guard much of their data. In some areas, the possibility of industrial espionage can be a factor to contend with. Using access control software and personal passwords for individual identities are examples of data security.

Sabotage, White Collar Crime, and Computer Frauds

Weigh threats from external forces (such as activist groups and competition) and internal forces (such as disgruntled employees and opportunists). Other factors that can affect these areas include: How easy is it to convert to cash your available assets using computer fraud? How many people (in collusion) would need to be involved? What are your personnel practices? For example, how are dismissals handled?

Good Business Practices

Use normal business practices (the same practices that your company uses in non-computerized areas) in computerized environments. These include separation of function, a clear line of responsibility and authority, individual accountability, knowing what the control procedures are and that they are in place, knowing who has access to data and records and controlling this access, and various auditability information.

Existing Corporate Policies

Almost every company or agency has some written policies already in place. Many of them do not relate to data or system security. Others are due to the factors mentioned above and can be reviewed as part of these other areas. Identify and consider all existing policies relating to data security, access control, and computer control auditability when you select CA ACF2 for z/VM options and build an overall security plan. Do not overlook future policies because they will probably be easier to implement and enforce if you consider them when designing the initial overall plan.

Organizational Impact

Sites usually want to implement data security that is transparent to users. While CA ACF2 for z/VM provides options to assist in its implementation and transition phases, these alone do not make security transparent. Normally, a site is changing from little or no data access control to significant controls. This is a big difference that cannot be totally transparent. With the proper planning, education, and phased implementation, CA ACF2 for z/VM can alleviate most problems and even create a positive, progressive attitude among users. The IT can also make organizational decisions about how to implement CA ACF2 for z/VM. These decisions include separation of function and centralization or decentralization in the administration of CA ACF2 for z/VM and related controls. The outcome requires organizational and job responsibility changes or minimize these changes if the selected approach is similar to the procedures that already exist at your site.

Procedure Enforcement Need

CA ACF2 for z/VM can also be a valuable aid in enforcing various corporate policies not directly related to data protection. These include items such as naming conventions for user identification. It can help enforce consistent policies throughout the corporation or agency, including different physical locations. The IT should consider these needs when selecting CA ACF2 for z/VM options and implementing its controls.

Identifying Your Local Operating Environment

To select the most appropriate options and to effectively use CA ACF2 for z/VM controls, you must identify local conditions.

Local Naming Conventions

Determine existing naming conventions for:

- User identification for the CMS online system and any other software products used locally
- MVS and VSE data set names, VM minidisks, CMS file IDs
- Volume serial names (for example, DASD and tape volumes)
- Physical device names for terminals
- Program names
- Procedure names for CMS EXEC procedures.

The significance of naming conventions depends on which CA ACF2 for z/VM options you choose. Conversely, the options you select can depend on your naming conventions. CA ACF2 for z/VM provides methods of controlling resource access, based on all of the fields listed above. It also lets you write global rules that reference name patterns for each of these fields. Rules are much easier to write if you use consistent naming conventions for minidisks, CMS file IDs, and MVS and VSE data sets. After you write access rules, CA ACF2 for z/VM forces you to comply with your naming conventions.

Standard Security Mechanisms

Identify current security mechanisms and decide which ones to replace and which to use. Before CA ACF2 for z/VM is in ABORT mode, you may want to keep all current security mechanisms active because it does not deny data or resource access while in QUIET, LOG, or WARN mode. If you implement the MODE=RULE option, you can phase in ABORT mode protection on a rule set basis.

Uniqueness of User Identifications

Identify whether each system user is uniquely defined to the system. Identify all users and any existing individual or group IDs. Establish plans to positively identify each system user with a unique logonid and unexpired password. Another significant consideration in planning is the selection of a User Identification string (UID) format, based on your individual ID patterns and organizational groupings.

Dependencies on Job Names and Account Numbers

Determine whether your site is currently using batch job names, account numbers, or similar fields for any controls. Decide whether you should replace these functions with CA ACF2 for z/VM features, discontinue them, or they should coexist. You should also ensure that these controls will not interfere with CA ACF2 for z/VM.

Other Security Controls

Identify other automated or manual security procedures that exist or are required at your site. Consider controls on:

- Physical devices (terminals)
- Test versus production work
- Systems programming activity.

Operating System Configuration

Identify other subsystems and software packages used (or that you plan to use). Review them to determine whether there will be any impact on CA ACF2 for z/VM or the other systems. Particularly important are:

- Disk management systems
- Archiving systems
- Library maintenance systems
- Interactive or online systems
- EXEC file procedures
- VM directory maintenance systems.

Backup and Recovery Procedures

Because the three CA ACF2 for z/VM databases (Rule, Logonid, and Infostorage) are critical to the smooth operation and security of the system, you should plan for their backup and recovery early in the implementation process. We provide an automatic database backup facility. The BACKUP VMO record lets you take up to 16 daily backups of the databases. See the *Administrator Guide* for detailed information about this record.

Selecting CA ACF2 for z/VM Options

Select CA ACF2 for z/VM options to customize implementation to your needs in phases. You must use certain options for the first IPL. You can select temporary values for other options to phase CA ACF2 for z/VM protection into your system. Indicate these choices by setting up various CA ACF2 for z/VM parameters. The Implementation Team should review all the options and select appropriate values for the first IPL and later use (with target dates for changing the significant ones). A few special areas are highlighted below.

UID Format

Carefully plan the structure of the User Identification string (UID). If the current VM directory user IDs have a high information content, such as information about division, department, or job responsibility, it may be adequate to carry over such conventions directly to the UID. But if this is not the case, you can add fields to the logonid record that contains this information. You can instruct CA ACF2 for z/VM to build its UID from a concatenation of these new fields and the logonid record. A UID constructed with a high information content lets the SA manage groups of individuals very easily because data access rules can specify UID patterns for access. In this way, you can give all users in a department or location access to protected resources (such as minidisks, CMS files, and OS data sets) by specifying one rule (you do not have to specify lists of the individual logonids of that group).

Boundaries of CA ACF2 for z/VM Controls

CA ACF2 for z/VM options are specified in the CA ACF2 for z/VM Field Definition Record (ACFFDR) macros and the VMO records. The ACFFDR macros and VMO records affect the overall bounds of controls on your system. These include options such as:

- The mode (QUIET, LOG, WARN, ABORT, or RULE) that CA ACF2 for z/VM is running in
- How centralized or decentralized CA ACF2 for z/VM administration is handled
- Which Control Program (CP) commands and diagnose codes are validated
- Whether to control IUCV or VMCF
- How to handle directory minidisk passwords.
- Because the values chosen for these options have a significant effect on the scope and completeness of CA ACF2 for z/VM controls, the IT should review each of the possible options carefully. This should include an item-by-item review of each individual option in the Field Definition Record and VMO records. See the *Installation Guide* for details about the ACFFDR. For more information about VMO records, consult the *Administrator Guide*.

Using the VM Directory and DirMaint

CA ACF2 for z/VM does not replace the VM directory information for each user. Virtually all information contained in the VM directory is valid, except for the password associated with your logonid. For example, you must enter your CA ACF2 for z/VM password when you log on instead of the VM user ID password. The directory minidisk password can be ignored or enforced. If you install the optional CA ACF2 for z/VM interface for DirMaint (the IBM Directory Maintenance Program Product), you must enter the CA ACF2 for z/VM password whenever DirMaint requires a password revalidation. The interface cannot replace the standard DirMaint password revalidation message and prompt. You should be aware that DirMaint prompts you with its standard password revalidation message, although it is the CA ACF2 for z/VM password that you must enter. CA ACF2 for z/VM also provides optional command limiting validation and logging for the DIRM command. When activated, CA ACF2 for z/VM can validate both privileged and general user DIRM commands and log all DIRM command activities. You can use this DIRM command support to detect and control minidisk overlaps, minidisk deletions, and minidisk transfers. It also provides an effective way to control directory maintenance functions. See the *Command and Diagnose Limiting Guide* for complete details.

Data Ownership

In the VM environment, MDISK entries in the VM Directory determine the data ownership of a minidisk. This is particularly important in decentralized security environments, where you are responsible for writing access rules for your own data. To share a minidisk with other users, the owner of the minidisk (or a security administrator) must compile and store an access rule set. This rule set specifies who can access the minidisk and what conditions must be met before the access is allowed. You can store a rule entry that allows access to other users for each CMS file that resides on the minidisk. See the “Writing Access Rules” chapter for more information.

Attachable Devices

CA ACF2 for z/VM validates all ATTACH commands issued for DASD devices. This validation is similar to minidisk and CMS file validation (you must write an access rule set for each DASD device that could be attached through a CP command). See the “Writing Access Rules” chapter for more information.

CA ACF2 for z/VM Exits

You can use CA ACF2 for z/VM exits to resolve unique user dependencies or provide specialized transition paths. An example is using the validation exit during migration to full security. You can use this exit to allow accesses, even though CA ACF2 for z/VM considers the access a violation. We supply sample source code for some exits with the distribution tape.

Sharing CA ACF2 for z/VM Databases between VM Systems

You can let several different VM systems that run CA ACF2 for z/VM share the same CA ACF2 for z/VM databases.

Synchronizing CA ACF2 for z/VM Databases

You can also synchronize CA ACF2 for z/VM databases between VM and CA ACF2 for z/VM for z/OS and OS/390 with the Database Synchronization Component.

General Planning Information

Before you install and activate CA ACF2 for z/VM, you should review the following:

- The System Management Facility macro (@SMF), see the *Installation Guide*
- ACF2PSMF utility, to process SMF data and run reports, see the *Reports and Utilities Guide*
- Planning for backup and recovery, see the *Administrator Guide*
- CA Earl™ reporting:
  - Reporting with CA Earl™
  - CA Earl™ Reference Guide
  - ACFRPTPP, the SMF Preprocessor Utility, see the *Reports and Utilities Guide*

You should also review the full-screen panels in the *Reports and Utilities Guide* and the *Administrator Guide* because you can modify these panels to meet your specific needs.

Chapter 4: Getting Started

This chapter contains the information you need to perform the first IPL and test. The SA and personnel responsible for the installation should carefully review all the material in this chapter.

This section contains the following topics:

The First IPL

Do the first IPL with CA ACF2 for z/VM active in a controlled environment. Although the entire Implementation Team does not have to be present, the SA and the systems programmer responsible for the installation should be there. You should perform system-testing functions and establish logonid records. To create valid logonids for all authorized users, run the ACFLIDGN utility to create a logonid record for each user defined in the VM directory. You must do this before the first IPL. The ACFLIDGN utility is explained in the “Using the ACFLIDGN Utility” chapter. You must also write some general rules and test the ACF subcommands. There is additional installation information to consider during the first IPL. You can find that information in the *Installation Guide*.

Testing CA ACF2 for z/VM

There are a number of items that you should test shortly after the first IPL (in addition to establishing logonid records and rules). The minimum items are noted below, but additional tests may be necessary for checking any local modifications or special (critical) processes.

Successful IPL

Do the first IPL with CA ACF2 for z/VM on a second-level test system.

CA-ACF2 and Other Service Machine Startup Procedures

Ensure that the CA-ACF2 service machine autologs successfully. Ensure that other important service machines function normally (such as MAINT, MVS, and DirMaint).

User Logons

Do the initial logon using the ACFUSER or MAINT logonids until you establish all user logonids.

Local Exit Testing

Test all local exits and modifications.

ACF SHOW subcommands

Issue ACF SHOW subcommands to verify that the correct VMO records are active (SHOW ALL, SHOW STATE, SHOW SYSTEMS, SHOW ACTIVE).

CA ACF2 for z/VM Reports

Run CA ACF2 for z/VM reports to produce reports that you can use to check other processing.

Backup and Recovery

Test the CA ACF2 for z/VM database backup and recovery procedures.

CA ACF2 for z/VM Commands

Test the various CA ACF2 for z/VM commands and subcommands on CMS to ensure that they all are working as expected (for example, display logonids and decompile rules).

ACFSERVE Commands

Test the operands of the ACFSERVE command.

Product Interfaces

Test interfaces with other products (DirMaint, VMBACKUP, and VMTAPE).

Chapter 5: Establishing Initial Logonids

The starter CA ACF2 for z/VM Logonid database contains predefined logonid records. See the *Installation Guide* for more information about these logonids and their associated passwords and privileges. Only these logonids can access the system until you add others to the Logonid database. This is true regardless of what MODE value (QUIET, LOG, WARN, ABORT, or RULE) you define in the OPTS VMO record.

To create logonids for all CA ACF2 for z/VM users, run the ACFLIDGN utility (described in the next section) to create a logonid record for each user in the VM directory. You must do this before the first IPL. You must also make sure that you change the MAINT password and the other sample IDs during the ACFLIDGN process or during the first logon. Define a valid VM directory entry for all logonids.

Note: The logonids presented in this chapter are only models. You must immediately change the passwords for all these logonids to maintain CA ACF2 for z/VM security. You should cancel, suspend, or delete them as soon as you have established all logonid records. This prevents anyone from using them without authorization. See the *Administrator Guide* for instructions on how to process logonid records.

This section contains the following topics:

Using the ACFLIDGN Utility

The ACFLIDGN utility creates logonids from entries in the VM directory. ACFLIDGN reads the directory and creates a logonid record for each user ID. The ACFLIDGN utility sets the PSWD-EXP field in each logonid record it creates, forcing users to change their passwords when they log on the first time. This engages the CA ACF2 for z/VM protection-by-default philosophy. The following circumstances warrant that the PSWD-EXP bit field in the logonid record be turned off:

- If you set the PSWDALT field (password alter option) of the PSWD VMO record to NO, users with logonid records that include the PSWD-EXP setting cannot change their expired passwords at logon time (they cannot log on).
- If you want a more gradual implementation of CA ACF2 for z/VM logon validation (passwords do not have to be changed when users log on).

To restrict which VM systems a user can log onto:

- Select an attribute to set the VMCHK option of the OPTS VMO record for each system. You can also define your own attribute.
- Give each logonid record the appropriate system attribute.
- Implement the restriction by setting the VMCHK option to the designated value for each system.

If appropriate, turn off the PSWD-EXP bit field and turn on the bit field specified by the VMCHK option of the OPTS VMO record in all logonid records by using the CMS ACF command as follows:

```
acf
ACF
set lid
LID
CHANGE LIKE(-) NOPSWD-EXP VM
```

As soon as ACFLIDGN completes processing, list all logonids using the Selected Logonid Report (ACFRPTSL) to verify that all the necessary logonids are present and correct. For example, ensure that special CA ACF2 for z/VM privileges such as SECURITY, ACCOUNT, AUDIT, LEADER, CONSULT, READALL, and NON-CNCL are only given to users that need them. If you are not sure what these special privileges are, see the *Administrator Guide*.

At this point, it might be necessary to use the ACF command to modify logonids. See the *Administrator Guide* for complete instructions on using the ACF command.

Maintaining the Logonid Database

CA ACF2 for z/VM has a full-screen feature to help you create logonids and maintain the Logonid database. You can also use the ACF command to modify logonids. With the full-screen feature, first create all the logonids for your site. Select the appropriate full-screen panel and fill in the fields. For more detailed information about using the full-screen feature and ACF command to modify logonids, see the *Administrator* Guide.

Selecting Special Logonids

Sometimes you will need to use special logonids and you must prepare for protection against their unauthorized use. The CA ACF2 for z/VM service machine requires only one logonid. All other service machines on that system, such as OPERATOR and AUTOLOG1, also require a logonid. AUTOLOG1 should have the AUTOALL privilege in the logonid record.

Another special logonid that may be needed is the service machine that normally performs disk backups, such as VMBACKUP. To avoid any problems with the backup process, you should define this service machine logonid with the NON-CNCL attribute. Include a MAINT record entry for this ID in the MAINT VMO record. This MAINT entry allows the backup ID to bypass CA ACF2 for z/VM rule validation when the backup ID backs up your system and does not create an SMF record. See the *Administrator* Guide for more details about the MAINT VMO record.

Consider any other VM service machines that might require a special logonid. All MVS and VSE service machines must also have a logonid record and a VM directory entry.

Defining User IDs for Emergencies

There are two attributes that can be defined to a logonid for emergency situations, FORCEID and NOAUTOU. The following describes each of these parameters.

What are FORCEIDs?

During implementation, you should also determine which logonids you should define with the FORCEID attribute. These IDs are very important. If CA ACF2 for z/VM ever becomes inoperable, only these FORCEIDs can access VM and make repairs to the system.

When a FORCEID user accesses CA ACF2 for z/VM in NOAUTO mode, CA ACF2 for z/VM does not validate the access other than to match that logonid to an internal list of FORCEID users. Normal CA ACF2 for z/VM password validation is not in effect. It is, therefore, **very important** to choose people with the FORCEID attribute very carefully. You define FORCEIDs in the FORCEID operand of the VMXAOPTS macros. The NOAUTO OPERAND of the VMXAOPTS macro controls whether the VM directory password is validated when a FORCEID logs on in NOAUTO mode. For more information about how to define FORCEIDs to your system, see the *Installation Guide*.

What are Noauto Update Users?

In addition to FORCEIDs, CA ACF2 for z/VM also provides a user ID that enables authorized users to log onto CA ACF2 for z/VM while it is inactive and update the databases. These users are called noauto update users. You must identify these user IDs with the NOAUTOU parameter in the VMXAOPTS macros. Only these special users can start up the CA ACF2 for z/VM service machine in NOAUTO mode and issue the appropriate ACF commands to update the CA ACF2 for z/VM databases. For a complete explanation of noauto update users and the ACFSERVE commands used to start the service machine in NOAUTO mode, see the *Administrator Guide*.

Chapter 6: Writing Access Rules

Access rules define the conditions under which access to data can or cannot occur. Only a security administrator or data owner can write access rules. This chapter focuses on how to write access rules and the CA ACF2 for z/VM features that can help simplify this important security component.

This chapter explains how to:

- Use the full-screen feature
- Set the CA ACF2 for z/VM mode
- Write rules for the first CA ACF2 for z/VM startup
- Create access rules
- Write temporary rules.

The SA and other personnel who are authorized to write rules should read this chapter.

This section contains the following topics:

Using the Full-Screen Feature

As with maintaining logonids, you can use the CA ACF2 for z/VM full-screen feature to write, change, and delete access rules. To use this feature, enter ACFFS from the CMS Ready prompt. You see a series of menus to select from. By selecting the screen and entering the appropriate data, writing rules becomes a very simple process. You do not need a high degree of technical expertise to create and maintain access rules. (You can also use the ACF command to maintain access rules. See the *Administrator Guide* for details.)

Setting the CA ACF2 for z/VM Mode

In most cases, sites decide to start CA ACF2 for z/VM in LOG mode. It is not absolutely necessary that you write access rules before the initial CA ACF2 for z/VM start-up. However, in any mode other than QUIET, CA ACF2 for z/VM validates all data access requests. If no rules exist, CA ACF2 for z/VM writes a logging record after validation. It does not prevent data access unless one of the following values are in effect:

- MODE(ABORT)
- MODE(RULE,...) with an applicable ABORT parameter.

CA ACF2 for z/VM generates a large number of logging records if no rules exist. This can lower the performance of your operating system. In addition, CA ACF2 for z/VM reports produced from these logging records are probably of little help in writing production access rules because of the large amount of data. We suggest that you write a few general rules for commonly accessed data. Carefully review these rules and refine them later.

If you select RULE mode, individual access rules can contain the mode for that particular rule set. Access rules without this mode control statement are governed by the system-wide mode specified. See the “Converting to Full Security” chapter for more information.

Writing Rules for the First Start-up

The value you supply for the $KEY control statement depends on the type of data you want to allow others to access. The $KEY statement must begin in the first column. For each data access request, CA ACF2 for z/VM validates the $KEY value that you specified and a data set name value. The $KEY controls which rule set is checked during validation. The dsn (data set name) identifies the full name of the file being accessed. In the VM environment, CA ACF2 for z/VM validates access requests to minidisks, CMS file IDs, MVS data sets, VSE files accessed under CMS, and attachable devices. The following sections explain how CA ACF2 for z/VM determines who can access data by validating a dsn for each of these data types and how the dsn is constructed.

Protecting LINK Requests

CA ACF2 for z/VM validates all minidisk LINK requests. This includes links predefined in a user's VM directory entry and all LINK commands issued through CP. You are automatically allowed access to any minidisks you own. You are denied access to any minidisks that you do not own unless the owner has specifically written a rule that allows you access. CA ACF2 for z/VM validates all access attempts to minidisks not owned by that user. CA ACF2 for z/VM checks:

$KEY

Specifies the logonid of the user who owns the minidisk.

dsn

Specifies the data set name in the form Vaddr.VOLUME

Vaddr

Specifies the virtual device address of the minidisk

VOLUME

Specifies the name of the minidisk.

Protecting CMS Files

CA ACF2 for z/VM validates all requests to access CMS files. CA ACF2 for z/VM checks:

$KEY

Specifies the logonid of the user who owns the minidisk that the CMS file resides on.

dsn

Specifies the data set name in the form Vaddr.filename.filetype

Vaddr

Specifies the virtual device address of the minidisk that the CMS file resides on (for example, V0191.WORK.DATA)

filename

Specifies the name of the CMS file

filetype

Specifies the type of the CMS file.

Protecting MVS Data Sets

CA ACF2 for z/VM validates requests to access MVS data sets through VM CMS. CA ACF2 for z/VM checks:

$KEY

Specifies the high-level index of the data set name. For example, SYS1 is the high-level index of SYS1.SHRPROC.

dsn

Specifies the data set name in the form index.index. ... .index. The rule entry begins with the second-level index and can extend up to 22 levels, with a maximum of eight characters per level.

Protecting VSE Data Sets

CA ACF2 for z/VM validates requests to access VSE data sets through VM CMS. CA ACF2 for z/VM checks:

$KEY

Specifies the high-level index of the file ID. For example, TLCAMS is the $KEY value of a rule set for TLCAMS.WORK.TEXT and TLCAMS.TEST.

dsn

Specifies the VSE data set name in the form index.index. ... .index. The rule entry requires only the second through the last qualifiers. For CA ACF2 for z/VM, a DOS data set name can contain a maximum of eight characters per level (maximum of 44 total characters).

Protecting ATTACH Commands

CA ACF2 for z/VM validates all CP ATTACH commands for DASD devices. CA ACF2 for z/VM checks:

$KEY

Specifies the $KEY value for the dedicated and attached DASD volumes (defined in the ATTKEY option of the OPTS VMO record). The default is SYSTEM.

dsn

Specifies the data set name in the form Rccuu.

cc

Specifies the channel number of the device

uu

Specifies the unit number assigned to the device. This is the real address of the device.

Writing the First Access Rules

The most general rule set that you can write includes only one rule entry for each possible $KEY value. This type of rule allows read, write, and execute access by anyone under any conditions. To write such a rule set for a minidisk and the CMS file IDs it contains requires just two lines. You can write these lines to a CMS file and use them as input to the CA ACF2 for z/VM rule compiler or enter them directly into the compiler using the ACF COMPILE command. The following example shows how this rule set appears:

```
$KEY(TLCAMS)
- R(A) W(A) E(A)
```

Instead of entering a dsn entry, a dash (-) represents all the possible minidisk identifiers; CMS file IDs, and MVS and VSE data set names. The dash (-) is a valid CA ACF2 for z/VM masking character and means the rule entry matches any value in this $KEY. $KEY control statements must begin in the first column. All rule input entries should begin in position two.

Because no other conditions are specified in this rule entry, the permissions apply to all users accessing any of TLCAMS's information. The permissions indicated by R(A) W(A) E(A) specify that reading, writing (updating), and executing (running execs and modules from TLCAMS minidisks) are all allowed.

Early rule sets like this one greatly reduce the number of loggings and the volume of reports. However, from a security point of view, this rule is too general to be a good permanent rule. You should only use this technique selectively and only as a transition aid to reduce report volume. When you are ready to write a more specific TLCAMS rule (while still in LOG mode), remove this temporary rule. This logs all accesses to TLCAMS files again (no rule applies, so they look like potential violations). Or, you could change access permissions in the rule from A (allow) to L (allow but log). Now, CA ACF2 for z/VM reports of TLCAMS file accesses can help write specific TLCAMS rules.

Writing Temporary Rules

Another CA ACF2 for z/VM rule feature that you could use is to specify that a rule entry is only temporary. This is done by putting an end date on it. For example, you could have created the previous TLCAMS rule like this:

```
$KEY(TLCAMS)
* TEMPORARY RULE TO REMOVE TLCAMS FROM REPORTS
- UNTIL(12/31/97) R(A) W(A) E(A)
```

When this rule is compiled and stored by CA ACF2 for z/VM, it automatically expires on December 31, 1997. With no rule left allowing anyone access after that date, accesses to TLCAMS files would again appear on the logging reports. This way, the security administrator would not forget to refine the initial rule. This example also contains an optional comment statement (any line with an asterisk in column one).

Based on the CA ACF2 for z/VM reports, you can write additional rules and redefine existing rules until you feel most rules are complete. You can use WARN mode as an additional period to refine rules before you implement full ABORT mode. See the “Converting to Full Security,” chapter for more information on these modes.

For additional information on rule set syntax and hints on rule writing, see the Administrator *Guide*.

Chapter 7: Writing Command Limiting Rules

With CA ACF2 for z/VM command limiting rules, you can control which CP (Control Program) commands and command operands a user can execute. That is, you can restrict even privileged users from executing a particular CP command without having to modify CP. This chapter explains the steps that you must follow to implement command limiting at your site.

This section contains the following topics:

Implementing Command Limiting

You can implement command limiting when CA ACF2 for z/VM is first installed. However, we recommend that you do not activate command limiting until you store the command limiting rules.

To implement command limiting:

- Select CP commands for CA ACF2 for z/VM validation
- Ensure that the command models are activated and loaded in the Infostorage database
- Write initial command limiting rule sets
- Compile and store command limiting rule sets
- Set up command limiting validation
- Move the command limiting rule sets toward full security.

The following sections describe the first five tasks. The sixth task is described in the "Converting to Full Security" chapter.

Selecting CP Commands for CA ACF2 for z/VM Validation

Before writing any command limiting rules, select the CP commands to include or exclude from CA ACF2 for z/VM validation.

CP Commands You Can Exempt from Validation

You may not need to validate the execution of certain CP commands because VM Class G users may commonly use them. Such commands include:

BEGIN

Continues or resumes execution in the virtual machine at the specified location

CLOSE

Ends the spooling activity

RESET

Clears all pending interrupts from the specified virtual devic

STORE

Stores data in specific registers or locations

SYSTEM

Clears virtual storage and associated keys, or simulate the RESET or RESTART commands

TERMINAL

Sets virtual console options.

CP Commands Recommended for Validation

You may need to tightly control the execution of other CP commands. This can include Class A commands such as:

FORCE

Logs the specified user off the system

SHUTDOWN

Terminates all VM/SP functions

STCP

Changes the contents of real storage.

You can also allow the use of certain commands, but log their execution. When determining which CP commands to include or exclude from validation, you can use the previous lists of commands as a guide. You should carefully review the security exposures imposed by each CP command and place controls where necessary.

Setting up Command Limiting Validation

If you write command-limiting rules for ACFSERVE commands, you must grant all users authority to use the ACFSERVE QUERY STATUS command. You can do this by adding the following rule entry to the ACFSERVE rule set:

```
$KEY(ACFSERVE)
 QUERY STATUS UID(-) ALLOW
```

Ensuring Models are Activated and Loaded

Command limiting models are activated through the MDLTYPE operand of the CMDLIM VMO record. There is one command model for each standard CP command. Each model describes the valid operands and combinations for a command. Change the MDLTYPE operand of the CMDLIM VMO record and compile the models. We provide models for standard CP commands. If you have developed unique CP commands, you can use CA ACF2 for z/VM to validate them. You must write a model to describe the command and its operands. See the *Command and* Diagnose *Limiting Guide* for complete details about writing models.

Writing Initial Command Limiting Rule Sets

Write the command limiting rules before you implement CP command limiting for selected commands. Write one rule set for each CP command subject to command limiting validation. Each rule set is identified by the CP command name.

The initial command limiting rule sets can be general, like this one:

```
$KEY(SPOOL) MDLTYPE(440)
- ALLOW
```

The following rule logs each command execution. These loggings provide a more specific view of who uses the command and how it is used. With this information, you can write a more specific rule set.

```
$KEY(SPOOL) MDLTYPE(440)
- LOG
```

The MDLTYPE parameter lets you have different command limiting rule sets on the database. It lets you have separate rules for new and changed CP commands and also provides for separate rule sets for multiple systems sharing the same database. The MDLTYPE parameter must correspond to the MDLTYPE of the CMDLIM VMO record. The following rule set lets you use the CP SPOOL command on the test system.

```
$KEY(SPOOL) MDLTYPE(TST)
- ALLOW
```

Compiling and Storing Command Limiting Rule Sets

You can create initial command limiting rule sets by first entering the text of the rule set into a CMS file. Compile the rule set using the ACF COMPILE subcommand under the CMDLIM setting. For detailed instructions on compiling command limiting rule sets, see the *Command and Diagnose Limiting Guide*.

Chapter 8: Writing Diagnose Limiting Rules

CA ACF2 for z/VM diagnose limiting rules tell CA ACF2 for z/VM to validate a user's authority to issue a diagnose instruction with a particular diagnose code. These rules let you protect specific diagnose codes. Each rule entry has the following elements:

- A description of the user that the rule applies to
- A description of the circumstances under which the rule is valid
- The type of access allowed (ALLOW, LOG, or PREVENT).

This chapter explains the process for implementing diagnose limiting and writing rules to protect diagnose codes.

This section contains the following topics:

Implementing Diagnose Limiting

You can implement diagnose limiting when CA ACF2 for z/VM is first installed. However, we recommend that you do not activate diagnose limiting until you store the diagnose limiting rules.

The procedure for implementing diagnose limiting is as follows:

- Select diagnose codes that CA ACF2 for z/VM validates
- Write the initial diagnose limiting rule sets
- Compile and store the diagnose limiting rule sets
- Set up diagnose limiting validation
- Move the diagnose limiting rule sets toward full security.

The first four tasks are explained in the sections that follow. The fifth task is explained in the "Converting to Full Security" chapter.

Selecting Diagnose Codes for CA ACF2 for z/VM Validation

Before you write the diagnose limiting rules, select the diagnose codes that CA ACF2 for z/VM includes or excludes from validation.

Diagnose Codes You Can Exempt from Validation

You can determine that executing certain diagnose codes requires no validation because they are commonly used by VM Class G users. Such diagnose codes include:

x'0C'

Stores the VM time information in the user's virtual storage.

x'18'

Performs input or output operations to a direct access device. System performance can be greatly affected if x'18' executions are logged.

x'20'

Specifies channel command word (CCW) chain to execute on a tape, disk, or unit record device. System performance can be greatly affected if x'20' executions are logged.

x'54'

Controls the function of the PA2 function key.

x'58'

Communicates with IBM 3270 display stations.

Diagnose Codes Recommended for Validation

You should restrict execution of the following diagnose codes:

x'30'

Reads one page of the system error recording area

x'34'

Reads the system dump spool file

x'3C'

Updates the VM directory

x’84’

Replaces the specified data in any VM directory entry.

These are only a few of the diagnose codes you can choose to validate. You should review all diagnose codes for security concerns and implement diagnose instruction control as necessary.

Setting up Diagnose Limiting Validation

You can include or exclude diagnose instructions from CA ACF2 for z/VM validation through the DIAGLIM VMO record. By using the DIAGLIM record to specify diagnose codes that CA ACF2 for z/VM should not validate, you can avoid validating selected diagnoses altogether and achieve better system performance.

Writing Initial Diagnose Limiting Rule Sets

Before you implement diagnose limiting you must write diagnose limiting rules. Write one rule set for each diagnose code CA ACF2 for z/VM validates. The $KEY value for the diagnose code is DIAGnnnn, where *nnnn* is the hexadecimal diagnose code. You can use the following rule set to validate a diagnose instruction with code x’0004’:

```
$KEY(DIAG0004) MDLTYPE(440)
 UID(TLCAMS) LOG
 UID(TLCPJM) LOG
```

The MDLTYPE operand lets you have separate sets of rules in a shared database complex. This operand is also shared with command limiting. For more information on the MDLTYPE operand of the DIAGLIM VMO record, see the *Administrator Guide*.

Diagnose instruction loggings provide you with a more specific view of who uses each diagnose code and how it is used. With this information, you can write more specific rule sets. If you include all diagnoses, be sure to write a rule for the 0ACF and ACF diagnoses that CA ACF2 for z/VM uses. Minimally, the MAINT user should be able to use the 0ACF and ACF2 diagnose. This lets the MAINT user IPL an CA ACF2 for z/VM-secured system with an IPL command and also use the ACF command.

Compiling and Storing Diagnose Limiting Rule Sets

You can create the initial diagnose limiting rule sets by entering the text of the rule sets into CMS files. Use the ACF COMPILE subcommand under the DIAGLIM setting to compile the rule set. For detailed instructions on compiling diagnose limiting rule sets, see the *Command* and *Diagnose Limiting Guide*.

Chapter 9: Writing Resource Rules

Resource control protects system resources from unauthorized access. Resources are valuable system services or objects that can affect the integrity of your system, such as group user IDs and certain commands, for example, the AUTOLOG command. CA ACF2 for z/VM lets you control access to resources by writing resource rules that name the resource, type of resource, and the users who can access that resource. As part of your initial implementation of CA ACF2 for z/VM, you should write resource rules to protect:

- ACCOUNT use
- The target of an AUTOLOG command
- The target of a DIAL command
- Access to group machines
- Access to the Inter-User Communication Vehicle (IUCV)
- Access to the Virtual Machine Communication Facility (VMCF).

This section contains the following topics:

Protecting ACCOUNT Use

You can assign an account number to a virtual machine whose costs are charged in a VM environment. This requires an ACCOUNT control statement in the CP directory entry for the virtual machine. The ACCOUNT statement includes a primary account number and up to seven alternative account numbers.

When a machine is logged on or autologged, it is assigned a number from its associated directory ACCOUNT statement. By default, the machine inherits its primary account number. An alternate number (if one exists) is assigned if it is specified in the LOGON or AUTOLOG command line as a value of the ACCOUNT operand. CA ACF2 for z/VM optionally uses resource rules instead of the directory for account validation. The primary (default) account is stored in the Logonid database.

To implement ACCOUNT support, the following five steps are necessary:

1. Determine your account mode setting. The ACCVLD operand of the OPTS VMO record establishes the account mode. There are three possible settings: YES, FULL, and LID.
2. Assign the VLDVMACT logonid privilege. This step is necessary if you use the ACCTVLD(LID) setting. If you want to use full account security (ACCTVLD(FULL)), skip this step.
3. Assign the VMACCT logonid value. Virtual machines that have account validation must have their default accounts assigned to them in this field.
4. Establish account resource rule sets by executing the ACFCVACT utility or issuing the COMPILE subcommand of the RESOURCE setting.
5. Set the account mode by setting the ACCTVLD operand of the OPTS VMO record to FULL or LID.

For additional information on ACCOUNT validation support, see the *Systems* Programmer *Guide*.

Protecting the AUTOLOG and XAUTOLOG Commands

This section provides some guidelines to follow when you implement AUTOLOG and XAUTOLOG command control. The following terms are used in this section:

Initiator Machine

Indicates the virtual machine (or user of that machine) who issues the AUTOLOG or XAUTOLOG command.

Target Machine

Indicates the virtual machine that is autologged using the AUTOLOG or XAUTOLOG command.

Password Suppression

Specifies an option that applies to the AUTOLOG, XAUTOLOG, and LOGON (not LINK) command. When you must supply a password with AUTOLOG or XAUTOLOG, this option requires that you enter the password following a prompt. The password is automatically suppressed by CA ACF2 for z/VM. If you enter the password on the same line as the AUTOLOG command, CA ACF2 for z/VM rejects the command when password suppression is in effect. CA ACF2 for z/VM cannot recognize a password that you enter on the same line as the AUTOLOG command.

Special CA ACF2 for z/VM logonid attributes for the AUTOLOG and XAUTOLOG command include:

AUTONOPW

Indicates a virtual machine with this privilege can be autologged without specifying a password. AUTONOPW has no effect on the logon process. The user of a machine with this privilege can log on using normal logon procedures.

AUTOALL

Indicates a virtual machine with this privilege can autolog any virtual machine without specifying a password, no matter what privileges the machines being autologged might have. Like AUTONOPW, AUTOALL has no effect on the logon process. The user of a machine with this privilege can log on using normal logon procedures.

Complete the following steps to implement the AUTOLOG command control:

1. Establish the appropriate command limiting controls to let users execute the AUTOLOG and XAUTOLOG commands. Implementation for these controls is at your discretion. The initiator machine must have the appropriate CP privileges for the AUTOLOG command.
2. Implement password suppression if required. Use HCPAC0 VMXAOPTS PSWDSUP=YES.
3. Establish AUTOLOG resource rule sets that let target virtual machines be autologged.
4. Determine which users can autolog or be autologged without passwords. Assign AUTOALL or AUTONOPW logonid privileges accordingly.

Protecting the DIAL Command

This section provides some guidelines to follow when implementing DIAL command control. The following terms are used in this section:

Dialer

Specifies the user who is requesting access to a target virtual machine. When a dialer requests access to a secured machine, CA ACF2 for z/VM prompts him to enter his logonid and current password.

Target Machine

Specifies the virtual machine to which the dialer is requesting access.

Complete the following steps to implement DIAL command control using CA ACF2 for z/VM:

1. Identify the virtual machines to be secured for DIAL access. By default, all machines are secured.
2. Establish command limiting controls that let users execute the DIAL command (optional).
3. Write DIAL resource rule sets that let target machines be accessed.

When the DIAL succeeds, CA ACF2 for z/VM creates a $DIAL SMF record. When the DIAL is severed, CA ACF2 for z/VM creates a $DROP SMF record. If a DIAL command does not pass each validation step, CA ACF2 for z/VM denies access and creates an SMF record. Details and examples for completing each of the above steps are provided in the following sections.

Establishing DIAL Command Limiting Controls

When you restrict DIAL using command limiting, you prevent unauthorized attempts to execute this command. The following controls determine the implementation for DIAL command limiting:

- The CMDLIM VMO record indicates the commands that are protected by command limiting. Use a LIMIT list to specify whether DIAL command limiting is performed. The default CMDLIM specification is ALL, which specifies that CA ACF2 for z/VM validates all commands.

  Display the commands currently validated for command limiting using the SHOW CMDLIM subcommand of the ACF command.

- If you decide to validate the DIAL command, create an appropriate DIAL command limiting rule set. For example:

  ```
  acf
  ACF
  set cmdlim
  CMDLIM
  compile dial
  $KEY(dial)
   - uid(-) allow
  COMPILER ENTERED .....
  store
  RULE DIAL STORED .....
  end
  ```

  This rule set lets all users issue the DIAL command to any virtual machine. Display the DIAL command limiting rule with the DECOMP subcommand of the CMDLIM setting.

  You also have to create a resource rule to let users DIAL to the target machine. Writing DIAL resource rules is explained in the next section.

Writing DIAL Resource Rules

To implement DIAL support for virtual machines, you must write DIAL resource rule sets and store them on the Infostorage database. The $KEY value for a resource rule is the target machine (the machine being dialed). For example:

```
acf
ACF
set resource(dia)
RESOURCE
compile
ACFpgm510I ACF COMPILER ENTERED
$KEY(dirm) type(dia)
 uid(maint) allow
 uid(tlc) allow
end
ACFpgm551I TOTAL RECORD LENGTH=NN BYTES NN PERCENT UTILIZED
store
ACFpgm769I RULE DIRM STORED
```

The DIAL operand of the RESCLASS VMO record defines the type-code required for DIAL resource rule validation. The default is DIA. You can modify this value.

This rule set lets DIRM be dialed by MAINT and any user with the TLC- UID mask. You should create additional DIAL resource rules that are appropriate for your site using the COMPILE subcommand of the RESOURCE setting. You can display resource rules with the DECOMP subcommand of the RESOURCE setting. If you do not want CA ACF2 for z/VM to validate a dialer to a virtual machine, set the DIALBYP (DIAL resource validation BYPASS) attribute on the target machine's logonid record. See the *Administrator Guide* for more information about DIALBYP.

Protecting Group Machines

Many sites allow multiple users access to a single virtual machine. It is sometimes necessary to have a virtual machine that several users can access. For example, several database administrators from different departments maintain a centralized database. They could perform the necessary maintenance from one virtual machine. To access the machine, all of the administrators enter the same logonid and password of the machine. Only one person can have access to the machine at a time.

Although using a virtual machine this way serves a valuable purpose, it imposes a possible security breach. From an auditing standpoint, there is no way to determine who is using the machine (other than someone logged on). Individual accountability is lost and passwords are shared.

You can solve this problem by defining a virtual machine with the special GRPLOGON privilege in its logonid record. A machine with this privilege is called a group virtual machine.

Many people can use this machine, but only one person can have access at a time. They all can be identified through auditing. The following terms are introduced in this section:

Group virtual machine

Specifies a virtual machine defined with the GRPLOGON privilege in the Logonid database. Many people can use it and still be individually identified by the system.

Group user

Specifies someone who uses a group virtual machine.

This section describes the user interface and requirements for each method of gaining access to a group virtual machine. It also includes a brief account of the mechanisms that are involved with each method, with an explanation of how individual accountability is enforced. A group virtual machine can also be autologged, just like any other virtual machine. For complete details on autologging virtual machines, see the Protecting the AUTOLOG and XAUTOLOG Commands in this chapter. After entering LOGON MAINT to use the MAINT group machine, group users supply their own logonids and passwords, not the MAINT password.

Logging on as a Group User

```
logon maint
ACFpgm263R Enter your ACF2 logonid
TLCAMS
ACFpgm244R Enter ACF2 password
PSWD
```

After a user has gained access, CA ACF2 for z/VM performs all validations as if the group virtual machine were logged on as a regular user. CA ACF2 for z/VM validates normal data and resource accesses against the group machine, not the group user. The SMF records that CA ACF2 for z/VM creates contain information on the group machine and also identify the group user. The user who logged onto the group virtual machine is identified in the CA ACF2 for z/VM reports. This enforces individual accountability.

LOGON-BY

In earlier releases of CA ACF2 for z/VM, you could access a user ID in two ways:

1. Enter the LOGON command and your password
2. Use the group logon facility.

With group logon, a logonid is designated as a group ID if the GRPLOGON attribute is present in a logonid record. A logonid record is designated as a mandatory group ID by the GRPLOGON attribute. Access to the ID requires a user to enter his own CA ACF2 for z/VM logonid and password and eliminates the need to share passwords. CA ACF2 for z/VM verifies the logonid and password against the group logon resource rules to determine if access is allowed. LOGON-BY lets you log onto another VM user ID and lets you optionally share IDs when a group logon resource rule exists that allows access. LOGON-BY:

- Allows authorized users access to user IDs without password sharing
- Improves logon sequence for group logon
- Allows user IDs to designate other users to access their IDs
- Identifies user IDs and accessing logonids in all CA ACF2 for z/VM reports.

Optional group IDs allow users to share user IDs. A logonid record is designated as an optional group ID when the GRP-OPT attribute is present in the logonid record. GRP-OPT designates an ID as an optional group ID. A logonid with this attribute can be logged onto as the primary ID, or a group ID. To access a virtual machine with this attribute as a group ID, this privilege must be present and a group logon resource rule must exist. If GRP-OPT and the GRPLOGON field are both present, then GRPLOGON takes precedence. To access an optional group ID, the group user enters the LOGON-BY syntax. When you use LOGON-BY to access another user ID, that ID becomes a group ID for the duration of the session.

Using the LOGON-BY Syntax

There are three syntax variations for LOGON-BY. Following are examples of the syntaxes. We recommend that you use number one to access another user ID because it ensures that CA ACF2 for z/VM does not display your password on the screen.

```
LOGON lid1 BY lid2
```

```
LOGON lid1 BY lid2 password
```

The password variable represents the user's own CA ACF2 for z/VM password.

```
LOGON lid1
```

CA ACF2 for z/VM issues the password prompt message and the user must reply:

```
BY/USER01/password
```

Enter the LOGON-BY syntax (including slashes) exactly as shown above. For complete details and instructions on how to use the LOGON-BY syntax, see the *Administrator Guide*.

Implementation Considerations

If your site currently allows users to log onto group IDs as themselves, you must change those IDs from mandatory group IDs to optional group IDs by removing the GRPLOGON attribute and specifying GRP-OPT. For example, you cannot use the syntax shown in the following example:

```
logon maint
ACFAB8263R Enter your CA ACF2 for z/VM logonid
maint
```

For complete information about how to change mandatory group IDs to optional group IDs, see the *Administrator Guide*.

Writing Group Machine Resource Rules

When you log onto a group virtual machine, a resource rule validates whether you can use the machine. This occurs regardless of the CA ACF2 for z/VM data access mode setting and any special privileges you might have (for example, SECURITY).

CA ACF2 for z/VM checks the resource rule this way:

- The user ID of the group virtual machine is validated as a rule set $KEY value.
- The UID of the user that you entered after the prompt:

  ```
  ACFpgm263R Enter your CA-ACF2 logonid
  ```

  This validates system entry. For this value, standard UID masking applies.
- The rule set type-code is GRP by default.

To implement CA ACF2 for z/VM group logon support for virtual machines, you must write group logon resource rules and store them on the Infostorage database. The $KEY value for a resource rule is the group logon machine.

For example:

```
$KEY(MAINT) TYPE(GRP)
 UID(TLC) ALLOW
 UID(MAINT) PREVENT
```

The RESCLASS VMO record defines the type-code required for resource rule validation of group machines. The default specification is GRP, implementing type-code GRP. This rule set lets any user with the UID mask TLC- log on as a MAINT group user. To ensure that no password sharing takes place, we recommend that you prevent users from logging on as the group machine itself. The PREVENT rule entry specifies this.

Use the COMPILE subcommand of the RESOURCE setting to create resource rules for the group machines at your site. Below is an example of how to create the resource rule set previously displayed.

```
acf
ACF
set resource(grp)
RESOURCE
compile
ACFpgm510I ACF COMPILER ENTERED
$KEY(maint) type(grp)
 uid(tlc) allow
 uid(maint) prevent
end
ACFpgm551I TOTAL RECORD LENGTH=NN BYTES NN PERCENT UTILIZED
store
ACFpgm769I RULE MAINT STORED
end
```

Invalid access attempts are reported in the Resource Event Log (ACFRPTRV). For more information about the ACFRPTRV report, see the *Reports and Utilities Guide*.

Important Group Machine Information

Important details apply to the system entry process for logging onto group virtual machines.

- A user cannot log onto a machine with the AUTOONLY privilege, regardless of whether that machine is a group machine. Invalid logon attempts are reported in the Invalid Password/Authority Log (ACFRPTPW).
- Password suppression is an IBM security feature that is implemented in the PSUPRS operand of the SYSJRL macro. It forces you to enter your password on a separate line from your logonid when you log on. Your password is not displayed when you enter it.

  If this feature is turned off, CA ACF2 for z/VM does not accept your password on the same line as your logonid. For example, if you try to log onto the MAINT group machine as a group user (by supplying your own password) or as the machine itself (by supplying the MAINT password), CA ACF2 for z/VM denies access and sends the following error message:

  ```
  logon maint password
  ACFpgm278I Logon for groupid -- password will be ignored
  ```

  When you log onto a group machine as a group user, you do not need to be defined as a user ID in the VM directory. The only prerequisite for a group user is that the user is defined with a logonid record in the Logonid database. You must always define the group machine with a logonid record and a user ID in the VM directory.
- Every time someone attempts to gain access to the system on a machine with the GRPLOGON privilege, CA ACF2 for z/VM automatically looks for a group logon resource rule. This occurs regardless of the CA ACF2 for z/VM data access mode setting and any special privileges the user has.

Protecting IUCV, APPC/VM, and VMCF

The Inter-User Communication Vehicle (IUCV), Advanced Program-to-Program Communications/VM, and the Virtual Machine Communication Facility (VMCF) define communication paths for the transfer of data between two processes (for example, virtual machines and CP system services). With CA ACF2 for z/VM, you can specify a fine degree of audit and control when establishing and terminating IUCV, APPC/VM, or VMCF communication paths.

You must supply the VMSAF LID attribute for the service machine to perform an APPC/VM path connection with password validation. This includes APPC/VM VTAM support (AVS) service machines.

The following terms are used in this section:

Initiator@Machine

Invokes a request to perform a data or message transfer.

Target Machine

Specifies the recipient of a request to perform a data or message transfer. This recipient is not necessarily a virtual machine, but can also be a CP system service or an application-defined resource ID.

When you issue a request to establish or terminate an IUCV, APPC/VM, or VMCF path connection, CA ACF2 for z/VM can allow, prevent, or log the action with a resource rule. CA ACF2 for z/VM checks the resource rule this way. (Each rule entry defines an IUCV, APPC/VM, or VMCF path connection from an initiator virtual machine to a target machine.)

- CA ACF2 for z/VM validates the user ID of the target machine as a rule set $KEY value.
- CA ACF2 for z/VM validates the user ID, CP system service, or resource ID of the initiator machine as a UID value of an entry in the rule set. For this value, standard UID masking applies.
- The rule set type codes are IUC (to validate IUCV and APPC/VM) or VMC (to validate VMCF) by default. You can modify these values.

Invalid path attempts or loggings using IUCV, APPC/VM, or VMCF resource rule validation are reported in the Resource Event Log (ACFRPTRV).

Writing IUCV Resource Rules

You can initiate IUCV communication using IUCV CONNECT and terminate it with IUCV SEVER. The following rule allows a path connection. TLCPJM can issue an IUCV CONNECT to communicate with TLCAMS. You must specify the initiator, TLCAMS, in the UID of the rule entry. You must specify the target (TLCPJM) as the $KEY value:

```
$KEY(TLCPJM) TYPE(IUC)
 UID(TLCAMS) ALLOW
```

Establishing and terminating the IUCV communication path is logged only if you specify the LOG access permission in the rule entry. If this is the case, CA ACF2 for z/VM creates an SMF record during the IUCV CONNECT function (indicating IUCV path initiation) and during the IUCV SEVER function (indicating IUCV path termination).

Writing APPC/VM Resource Rules

You can initiate VMCF communication through VMCF AUTHORIZE and terminate it with VMCF UNAUTHORIZE. The path is controlled in exactly the same way as IUCV communication. The following rule allows a path connection. TLCPJM can issue a VMCF AUTHORIZE to communicate with TLCAMS. You must specify the initiator (TLCAMS) in the UID of the rule entry. You must specify the target (TLCPJM) as the $KEY value:

```
$KEY(TLCPJM) TYPE(VMC)
 UID(TLCAMS) ALLOW
```

Establishing and terminating the VMCF communication path is logged only if you specify the LOG access permission in the rule entry. If this is the case, CA ACF2 for z/VM creates an SMF record during the VMCF AUTHORIZE function (indicating VMCF path initiation) and during the VMCF UNAUTHORIZE function (indicating VMCF path termination).

Important Information

Important details apply when you create IUCV, APPC/VM, or VMCF resource rules:

- To activate IUCV, APPC/VM, and VMCF support, specify the IUCVVLD and VMCFVLD operands in the OPTS VMO record. See the *Administrator Guide* for additional information.
- The COMSEC operand of the SRVMOPTS or the VMXAOPTS macro indicates the name of each target machine, CP service, and resource ID secured for IUCV, APPC/VM, and VMCF resource rule validation.

  The default COMSEC specification is (INCLUDE,-). It secures all virtual machines, CP services, and resource IDs for IUCV, APPC/VM, and VMCF resource rule validation. You must write resource rules to use IUCV, APPC/VM, and VMCF communications with this default setting.

  To secure a CP system service in the COMSEC list, a percent sign (%) represents an asterisk (*) to distinguish it from the masking character. For example, to exclude the *MSG service in a COMSEC EXCLUDE list, enter (EXCLUDE,%MSG). The specification COMSEC=(INCLUDE,ALL) does not secure all virtual machines, CP services, and resource IDs for IUCV, APPC/VM, and VMCF validation. ALL is a valid target name for VMCF only (specified as a $KEY value for VMCF resource validation). ALL indicates that all virtual machines and CP services for VMCF are validated, not those for IUCV or APPC/VM. INCLUDE, ALL does not perform resource rule validation for IUCV or APPC/VM.

- The IUCV and VMCF operands of the RESCLASS VMO record define the type codes required for IUCV, APPC/VM, and VMCF resource rules, respectively. The defaults are IUCV=(IUC) (implementing type code IUC for IUCV and APPC/VM validation) and VMCF=(VMC) (implementing type code VMC for VMCF validation). You can modify these values.
- You must write an IUCV, APPC/VM, or VMCF resource rule set and store it on the Infostorage database. The $KEY value for a resource rule is the target machine. For example:

```
$KEY(DIRM2) TYPE(IUC)
 UID(TLC) LOG
```

This rule set establishes a communication path initiation, allowing users with the TLC- UID mask to issue an IUCV CONNECT to DIRM2. Because the LOG access permission is specified, CA ACF2 for z/VM creates an SMF record when the IUCV path is established and terminated. This rule would also allow an APPC/VM CONNECT to a DIRM2 resource ID.

- Create additional IUCV, APPC/VM, and VMCF resource rules with the COMPILE subcommand of the RESOURCE setting. Here is an example of creating the resource rule set shown above:

```
acf
ACF
set resource(iuc)
RESOURCE
compile
ACFpgm510I ACF COMPILER ENTERED
$KEY(dirm2) type(iuc)
 uid(TLC) log
end
ACFpgm551I TOTAL RECORD LENGTH=NN BYTES NN PERCENT UTILIZED
store
ACFpgm769I RULE DIRM2 STORED
end
```

You can display the resource rules on the Infostorage database with the DECOMP subcommand of the RESOURCE setting. Each rule entry defines a one-way IUCV, APPC/VM, or VMCF path connection from an initiator virtual machine to a target machine. A separate rule is required for each recipient to perform an IUCV, APPC/VM, and VMCF data or message transfer. In each rule, you must specify the logonid of the target machine or CP service as the $KEY value. The UID portion of the rule entry is the initiator machine.

Masking in Resource Rules

You can also mask the $KEY of a resource rule. This greatly reduces the number of rules that you need to write during initial implementation.

Use masking to select a series of items, such as files or programs that have similar characters in their names. Instead of selecting each item individually, masking lets you select all items at once by entering a masking name-that is, a common root name and masking symbols. A masking symbol is like a “wild card” that represents any other character that can be included in the name of an item. CA ACF2 for z/VM supports two masking symbols: the asterisk (*) and the dash (-).

How to Mask Resource Rules

Let’s look at this example of how you can use masking to create resource rules. To control the AUTOLOG command in a later phase of implementation, write a rule like this:

```
acf
ACF
set resource(alg)
RESOURCE
compile
$KEY(********) type(alg)
 uid(-) log
RESOURCE
store
end
```

In this rule, CA ACF2 for z/VM allows and records all virtual machine autologging. With this rule, you can gather the information to write specific rules without impacting any normal operations.

Building the Resident Resource Type List

To use masking, you must first build a resident resource type list. CA ACF2 for z/VM builds this list when you initialize the ACF2 service machine or use an ACFSERVE command. Building these resident lists is important because CA ACF2 for z/VM uses them to determine which resource rule to use during validation.

The RESTYPE VMO record specifies the three-character type code of the resources you want CA ACF2 for z/VM to automatically build a resident list for. For example, to build a list for the standard CA ACF2 for z/VM controlled resources, the RESTYPE VMO record is:

```
RESTYPE TYPES=(ALG,DIA,GRP,IUC,VMC)
```

We recommend that you use the RESTYPE VMO record because it ensures that CA ACF2 for z/VM always builds the lists.

Building a Resident List Manually

The ACFSERVE command lets you manually build a resident list while CA ACF2 for z/VM is running. For example, to build a list for the AUTOLOG command, enter ACFSERVE LOAD RESOURCE ALG. You can specify any three-character type-code.

Chapter 10: Writing Shared File System Rules

The VM Shared File System (SFS) extends the CMS file system by giving CMS users the ability to use CMS files that do not reside on user minidisks, but in hierarchical directories that are very similar to the directories available under MS-DOS.

This section contains the following topics:

SFS File Identifiers

Each SFS file server manages a collection of SFS files called a filepool. There is only one filepool for each SFS server machine. Every user must know the name of the filepool where any files reside that they might need access to. The filepool name is an integral part of the file identifier. Every SFS file in a filepool is associated with its owner by residing in a root directory of the same name as the owner user ID. Like MS-DOS, there may also be hierarchical subdirectories under each root directory that also contain SFS files. SFS supports up to eight levels of subdirectories. Each subdirectory name can be up to 16 characters long.

Let's take a look at how all these components fit together to form the file identifier for an SFS file. As an example, consider a CMS file called MY DATA that is owned by user ID TLCAMS in the filepool APPLDATA. This file resides in TLCAMS' root directory. The complete file identifier for this file would be:

```
APPLDATA:TLCAMS.MY.DATA
```

A colon (:) must always follow the filepool name and the owner ID always follow the colon.

Now, let's consider a more complex example where the file resides three levels deep under subdirectories. The first level is a directory called UNITEDSTATES. The second level is called ILLINOIS. The third level is called CHICAGO. The full file identifier is:

```
APPLDATA:TLCAMS.UNITEDSTATES.ILLINOIS.CHICAGO.MY.DATA
```

It is important to note that the last two elements of the file identifier are always the filename and filetype of the CMS file. You can refer to an SFS file using the full file identifier, however, this is usually awkward. Most users instead prefer to access the directory where the file resides at a CMS filemode exactly as they would for a minidisk.

Let's assume that you want to access the directory in our most recent example as the E disk. You would enter the following:

```
ACCESS APPLDATA:TLCAMS.UNITEDSTATES.ILLINOIS.CHICAGO E
```

After completing this step, all the files in that subdirectory are accessible as if they resided on a minidisk at mode E. In this example, you would refer to MY DATA E. Because you will normally want to access files that reside entirely in a single filepool, there is a way to declare a filepool so that you do not have to enter the filepool name again. An example of this syntax follows:

```
SET FILEPOOL APPLDATA:
```

After you enter the SET FILEPOOL command, use the following command syntax to access the directory:

```
ACCESS TLCAMS.UNITEDSTATES.ILLINOIS.CHICAGO E
```

You can also refer to your own root directory with a period (.) instead of entering its name. For example, TLCAMS could access this directory by entering:

```
ACCESS .UNITEDSTATES.ILLINOIS.CHICAGO E
```

Protecting Shared File Systems

Like CA ACF2 for z/VM minidisk file level protection that is split into minidisk volume rule and separate CMS file rules, CA ACF2 for z/VM SFS protection is split into two different types of rules: Directory access rules and file access rules. While a user cannot access files on a minidisk unless he is authorized to link to the minidisk, he can access files in an SFS directory without having the authority to access the directory itself. This is most easily done through an alias or through callable services library (CSL) routines. Users can also have write access to files in a directory for which they only have read authority. The main restriction is that if a user does not have read access to a directory, he cannot list the contents of that directory. A user must have write authority to a directory to:

- Create a new file
- Create a new subdirectory
- Create an alias
- Delete a file or alias
- Rename a file.

Additional authorization is necessary to manipulate each object in the directory. To provide directory level authorization, the directory owner must write a directory access rule as part of his rule set. The SFS directory access rule is required to issue the CMS ACCESS command for the directory. The next two sections provide information on writing directory and file access rules for shared file systems.

Writing Access Rules for SFS Directories

To access a file in an SFS directory, you need the proper authorization to access the relevant directory. You do this by writing a directory access rule as part of your rule set. The SFS directory access rule is required to access files and to issue the CMS ACCESS command for the directory. For example, if TLCPJM tries to access TLCAMS' directory called UNITEDSTATES.ILLINOIS.CHICAGO that resides in the APPLDATA filepool, he must issue the following command:

```
ACCESS APPLDATA:TLCAMS.UNITEDSTATES.ILLINOIS.CHICAGO E
```

Because TLCPJM does not own this directory, the action defaults to access this directory in read only mode. To allow access, the following read access rule is required:

```
$KEY(TLCAMS)
 / FILEPOOL(APPLDATA) DIR(UNITEDSTATES.ILLINOIS.CHICAGO)UID(TLCPJM) READ(A)-
EXEC(A)
```

Even though some CMS commands (such as RENAME) can directly address SFS files without the directory being first accessed by the CMS ACCESS command, you still need the directory access rule because implied access is in effect for the duration of the file access. You can use standard CA ACF2 for z/VM masking characters for the filepool (abbreviated FP) and the directory values. User ID TLCPJM can explicitly request that this directory be accessed in write mode as follows:

```
ACCESS APPLDATA:TLCAMS.UNITEDSTATES.ILLINOIS.CHICAGO E (FORCERW
```

There are two types of SFS directories: File Control directories and Directory Control (DIRCONTROL) directories. File Control directories are the most common type. The above access rule would work even if the user only has READ authority for the directory. DIRCONTROL directories require both READ and WRITE access permission in the rule to authorize the access, as shown in the rule below:

```
$KEY(TLCAMS)
 / FILEPOOL(APPLDATA) DIR(UNITEDSTATES.ILLINOIS.CHICAGO) UID(TLCPJM) -
 READ(A) WRITE(A) EXEC(A)
```

Because SFS directories accessed in write mode can also be read, you must always specify READ(A) when you specify WRITE(A) for SFS access rules.

Writing Access Rules for SFS Files

After you write a rule allowing a user to access a directory, you must also write a rule allowing him to open the files in that directory. Unlike native SFS security where DIRCONTROL directories provide no individual file-level security, CA ACF2 for z/VM SFS security protects individual files in both File Control and DIRCONTROL directories.

Suppose TLCPJM needs to read TLCAMS' file called MY DATA in the UNITEDSTATES.ILLINOIS.CHICAGO directory. That directory resides in the APPLDATA filepool, so the following rule is required:

```
$KEY(TLCAMS)
 / FILEPOOL(APPLDATA) DIR(UNITEDSTATES.ILLINOIS.CHICAGO) UID(TLCPJM)–
 READ(A) EXEC(A)
 MY.DATA FILEPOOL(APPLDATA) DIR(UNITEDSTATES.ILLINOIS.CHICAGO)–
 UID(TLCPJM)
 READ(A) EXEC(A)
```

In the above example, we have included the directory access rule and the file access rule because both are required. If you wanted to write a single rule that would authorize TLCPJM to read all the files in this directory and also authorize read access for the directory itself, you could write a rule like this one:

```
$KEY(TLCAMS)
 / FILEPOOL(APPLDATA) DIR(UNITEDSTATES.ILLINOIS.CHICAGO) UID(TLCPJM) –
 READ(A) EXEC(A)
```

Chapter 11: Converting to Full Security

The design philosophy of CA ACF2 for z/VM states that all data and resources are protected by default. CA ACF2 for z/VM is shipped with a default mode of ABORT. However, you can phase in data protection gradually, enabling you to implement access control without a security exposure.

There are two methods that you can use to convert from an environment with no security to one with full security under CA ACF2 for z/VM. These are system-wide conversion and selective migration. We explain both methods in this chapter.

This section contains the following topics:

System-Wide Conversion

Your site can move through the various conversion phases by using the standard CA ACF2 for z/VM system-wide option selections and no exits. You can activate individual security modes for access rules, command limiting rules, and diagnose limiting rules. The chart below provides information on where you can define CA ACF2 for z/VM modes for specific functions.

Function	VMO Record
Access Rules	OPTS VMO record
Command limiting rules	CMDLIM VMO record
Diagnose limiting rules	DIAGLIM VMO record

See the *Administrator Guide* for more information on VMO records and how to activate mode values.

CA ACF2 for z/VM Mode Transition

CA ACF2 for z/VM provides for a gradual migration to full security. The following fives modes ease the transation:

QUIET Mode

After you install CA ACF2 for z/VM, you can set the system mode to QUIET. For example, you may still be writing access rules to access data, but you want users to be able to access data without creating SMF records. The purpose of QUIET mode is to let you begin rule writing while CA ACF2 for z/VM is active, although not in full ABORT mode. In QUIET mode, as in all CA ACF2 for z/VM system modes, the system verifies that the user is authorized to access the system and verify the following:

- Logonid and password of a user entering the system are correct
- Access is through a correct input source
- User is accessing during a specific shift and according to other privileges defined by the user’s logonid record.

Note: CA ACF2 for z/VM does not check access rules in QUIET mode.

LOG Mode

After you install CA ACF2 for z/VM and have written enough rules to control access to data, you may want to migrate to LOG mode. LOG mode lets you check to see that the appropriate users can access certain data and that CA ACF2 for z/VM creates SMF records when unauthorized users try to access that data. In LOG mode, CA ACF2 for z/VM checks users at system entry, as it did in QUIET mode. It also checks the Rule database for any access rules that apply to the request. If it determines that an attempted access permission or type is invalid according to a rule, or if CA ACF2 for z/VM cannot find a matching rule set, it allows access, but logs the violation.

From the reports, the security administrator writes, compiles, and stores rules to reduce the number of future violations. The security administrator should consult with data owners to determine whether the accesses are valid. You can also use a decentralized administration with multiple security administrators writing rules (each for his own department or group).

WARN Mode

WARN mode functions the same way as LOG mode, except that CA ACF2 for z/VM issues a warning message to the user when there is a violation to an access rule. With each warning message, CA ACF2 for z/VM also sends a site-supplied warning message. The message instructs the user that in the future this access will be denied if the data owner or security administrator does not change the rule to allow access. WARN mode gives users who do not have permission to use the data a chance to inform the security administrator that they need access. CA ACF2 for z/VM also creates a logging for each of the accesses that are denied. Use WARN mode only for a limited period of time (two weeks is a reasonable guideline) to ensure that you refine rules as necessary. During that time, users have a chance to request any needed changes before migrating to ABORT mode and causing abends. If you run CA ACF2 for z/VM too long in WARN mode, it tends to make users ignore the messages and become impatient with the system.

RULE Mode

RULE mode lets you migrate to full ABORT mode on a rule set basis. There are three action values that you can set for RULE mode (one for access rules, one for command limiting rules, and one for diagnose limiting rules). One value tells CA ACF2 for z/VM what to do if no rule exists on the Rule database. If a rule exists, CA ACF2 for z/VM checks the setting in the $MODE control statement for a mode for that rule set. You can also instruct CA ACF2 for z/VM in what to do if no $MODE control statement exists. For each of the three action values, you can specify QUIET, LOG, WARN, or ABORT. RULE mode gives you maximum flexibility when implementing security. It allows you to target your critical object for abort protection, while leaving less critical objects in QUIET, LOG, or WARN mode.

ABORT Mode

This is the final and normal mode of operation. Under ABORT mode, CA ACF2 for z/VM provides full protection. Invalid accesses are denied. Only authorized users are allowed access to the system. Rules determine and control access to data and resources. CA ACF2 for z/VM denies all invalid access attempts and logs the violations.

Selective Migration

Many sites use an alternative conversion method to place CA ACF2 for z/VM into RULE mode at an early stage, then allow or log specific requests using the $MODE control statement in the rule sets. The mode enforced for a given access is determined by any of the following:

- Migration by rule set
- Migration by user group

- Local criteria
- A combination of the above.

This lets you selectively phase in protection for different groups (departments or geographical locations) or for different data (production versus test).

Migrating by Rule Set

Two standard CA ACF2 for z/VM features enable migration based on information in the applicable rule set. These are RULE mode and NEXTKEY.

Using RULE Mode

To activate RULE mode for access rules, specify the following in the OPTS VMO record:

```
MODE(RULE,no-rule,no-$mode)
```

To activate RULE mode for command limiting rules, specify the following in the CMDLIM VMO record:

```
MODE(RULE,no-rule,no-$mode)
```

To activate RULE mode for diagnose limiting rules, specify the following in the DIAGLIM VMO record:

```
MODE(RULE,no-rule,no-$mode)
```

For more information on these VMO records, see the *Administrator Guide*.

In RULE mode, CA ACF2 for z/VM verifies the access based upon the value specified in the $MODE control statement. The NO-$MODE parameter specifies a default mode for rule sets that do not contain the $MODE control statement. Similarly, the NO-RULE parameter specifies a default mode if CA ACF2 for z/VM cannot find a rule set (no record). The mode can be any of the other four CA ACF2 for z/VM modes: QUIET, LOG, WARN, or ABORT. For example, you can set the mode for data access validation in the VMO OPTS record:

```
MODE(RULE,ABORT,ABORT).
```

Suppose TLCPJM wants write access to the PROD TEST file on TLCAMS's 0191 minidisk. As shown below, the TLCAMS rule controls access to this file when CA ACF2 for z/VM validates it.

```
$KEY(TLCAMS)
$MODE(LOG)
 V0191.VOLUME R(A) W(A)
 V0191.PROD.MASTER UID(TLCPJM) R(A) W(A) E(A)
 V0191.PROD.TEST UID(TLCPJM) R(A) W(P) E(A)
```

Even though the rule indicates that TLCPJM cannot write to the file (W(P)), the $MODE(LOG) control statement tells CA ACF2 for z/VM to allow the access but log it. If you specify $MODE(QUIET), CA ACF2 for z/VM allows the access with no logging. If you specify $MODE(ABORT), CA ACF2 for z/VM prevents access.

Below is a sample transition rule showing how the CP SPOOL command can begin in LOG mode:

```
$KEY(SPOOL)
$MODE(LOG)
 CON PURGE UID(TLCAMS) LOG
 CON - UID(TLCPJM) ALLOW
 PRINT COPY - ALLOW UID(*)
 PRINT RSCS ALLOW UID(*)
 - UID(*) ALLOW DATA(ALLOW ALL OTHER OPERAND COMBINATIONS)
```

After you test the rule, you can change the $MODE statement to test the rule set under WARN, then ABORT mode. After all rule sets are under ABORT mode, you can change the applicable system-wide mode to ABORT. A diagnose limiting rule set can begin in LOG mode as follows:

```
$KEY(DIAG0004)
$MODE(LOG)
 UID(TLCAMS) ALLOW
 UID(TLCPJM) ALLOW
```

After you test the rules under LOG mode, you can change the $MODE statements to test the rule sets under WARN, then ABORT mode. After you complete these tests, you can change the system-wide diagnose limiting mode to ABORT.

Using NEXTKEY for Access Rules

The NEXTKEY feature lets you split an access rule set into several smaller rule sets. Each could specify its own $MODE. For example:

```
$KEY(TLCAMS)
$MODE(LOG)
 V0191.VOLUME R(A) W(A)
 V0191.PROD.MASTER UID(TLCPJM) R(A) W(A) E(A)
 V0191.PROD.TEST UID(*) NEXTKEY(TEST)

$KEY(TEST)
$PREFIX(TLCAMS.V0191)
$MODE(ABORT)
 PROD.TEST UID(TLCPJM) R(A) W(P) E(A)
```

CA ACF2 for z/VM protects the PROD.TEST file on the TLCAMS 0191 minidisk in ABORT mode, while CA ACF2 for z/VM logs any invalid access to the PROD.MASTER file on the same minidisk. For more details on NEXTKEY, see the *Administrator Guide*.

Migrating by User Group

During these phases (such as LOG, WARN, and ABORT modes), other activity is taking place besides rule writing. Review the system-wide options selected in the ACFFDR and VMO records and modify them as you reach various implementation schedule points. You should bring additional areas under CA ACF2 for z/VM control if they were not defined initially, such as the VM directory maintenance program. You should identify all users, assign them unique logonids, and define them to CA ACF2 for z/VM. You must determine and define their privileges. For example, instruct users to issue their logonid and password at logon time. They may need some instruction on recovery procedures and new CA ACF2 for z/VM commands and messages, or both.

In decentralized environments, you may need to provide additional user training in rule writing and using CA ACF2 for z/VM commands. Refer them to the CA ACF2 for z/VM documentation.

Throughout the migration phases and on a continuing basis after you have reached full ABORT mode, you should print and review the CA ACF2 for z/VM reports. These reports are very useful in the early phases to help you write rules and define user privileges. As soon as you establish certain privileges and rules (even in LOG mode), the reports identify violations that you should research and take appropriate action. On an ongoing basis, these reports can be an invaluable aid in detecting security threats.

Sharing CA ACF2 for z/VM Databases

Under CA ACF2 for z/VM, you can set up your CA ACF2 for z/VM VM systems to share the same databases. This section outlines the basic administrative procedures for implementing database sharing. You can implement database sharing when CA ACF2 for z/VM is installed or later. We supply a database merge utility named ACFDBVSM that merges CMS databases into VSAM databases. You must first decide whether it would be an advantage to implement database sharing. Both sides of this option are illustrated on the following page.

Benefits of Database Sharing

Some benefits of database sharing are:

- Security administration workload is reduced. You only need to maintain records on one set of databases instead of several databases.
- Security administration time is reduced. Because you have to maintain fewer records, you need less maintenance time. Administration time normally spent in switching from system to system is also saved.
- Accuracy and organization of records is increased. In most cases, you must only check one set of records for accuracy. Having fewer records to maintain also makes it easier for you to locate information.
- Easier for users to maintain. If you work with several operating systems, you need to be concerned with only one logonid record and one password.

Limitations of Database Sharing

An CA ACF2 for z/VM logonid must have the same attributes under all systems. However, a logonid can have unique attributes under each system. You can circumvent this limitation by creating and maintaining additional logonid records.

If you spend a lot of time processing logonid records and access rules across several operating systems, you will benefit from database sharing.

Preparing for Database Sharing

The following list describes some information for database sharing. Existing CA ACF2 for z/VM sites have additional information to consider if they run the database merge utility (ACFDBVSM). This information is pointed out below, where appropriate.

Passwords

When you log on for the first time under database sharing, you must use your most recently-updated password. The PSWD-TOD field of the logonid record indicates the date that you last changed your password.

Access rules

After ACFDBVSM processing, any rules that existed in both databases are merged.

Entry, scope, shift, and zone records

You should inspect these records before you implement database sharing. After the implementation, check these records for existence and accuracy. Only the records that are not duplicated by their VSAM counterparts should be added during ACFDBVSM processing.

UID

You must use the same UID structure on all systems so that CA ACF2 for z/VM interprets all rule sets the same way for all systems. Print specific logonid records with their associated access rules for later comparison.

Unique system control

If you want to exercise separate control over different VM systems in a shared database complex, consider having unique ACFFDRs for each system. With unique ACFFDRs, you can specify a separate SMFID so you know what system is being logged. You can control what systems users can to log onto, have unique or shared rules sets for command limiting, diagnose limiting, CA ACF2 for z/VM-defined resources, and separate ATTACH rules.

SMF control

To identify which system an SMF logging occurred on, use the SMFID= operand of the @SMF macro in the ACFFDR. The SYSID macro in the ACFFDR is also recorded in the SMF records.

Command limiting and diagnose limiting control

For separate command limiting and diagnose limiting rule sets

- Choose an MDLTYPE value for each unique system.
- Compile the command models for each value. For example, if you have three systems installed, call them MDLTYPE=VMA, MDLTYPE=VMB, and MDLTYPE=VMC. Compile the model set under each MDLTYPE:

```
ACF

set model

MODEL

compile ZVMnnn mdltype(VMA) nolist

 .

 .

 .

compile ZVMnnn mdltype(VMB) nolist

 .

 .

 .

compile ZVMnnn mdltype(VMC) nolist

 .

 .

 .
```

- Implement by setting the MDLTYPE operand of the CMDLIM VMO record to the designated value for each system.

CA ACF2 for z/VM-defined resource control

To have unique system resource rules for CA ACF2 for z/VM-defined resources, modify the TYPES option of the RESTYPE VMO record and write the appropriate resource rules.

DASD ATTACH control

To have unique ATTACH rules for each system, modify the ATTKEY option of the OPTS VMO record and write the appropriate system rules.

Activating Database Sharing

Activating database sharing is mainly a technical process. Read all the prerequisites for this process before proceeding. Detailed information on the technical aspects and the required site or system maintainance steps is explained in the Systems *Programmer Guide*.

Recovering Databases under Database Sharing

If you decide to share databases, test the CA ACF2 for z/VM database recovery procedures thoroughly. Databases that fail and are shared by multiple systems can cause all systems to fail. Keep alternate databases current. All CA ACF2 for z/VM products provide automatic database backup options that you can use to ensure that alternate databases are always available. For more information on how to recover the CA ACF2 for z/VM databases, see the *Administrator Guide*.

Chapter 12: Fundamentals Included in Your Package

This chapter discusses the fundamental items included in your CA ACF2 for z/VM package.

This section contains the following topics:

Genlevel Maintenance Updates

These contain announcements of relatively severe problems (and their fixes) when we feel it is inappropriate to wait until a new release is issued. The genlevel maintenance update numbering system is GENLEVEL: *rryymmpppss*. These variables are explained below:

rr

Specifies the product release number (two digits)

yy

Specifies the genlevel year (two digits)

mm

Specifies the genlevel month (two digits)

ppp

Specifies the three-character product code (two letter and one digit)

ss

Specifies the genlevel sequence number (three digits).

GENLEVEL: 800505AM901 is an example of a valid genlevel maintenance update.

Installation Tape and Distribution Tape

This tape contains the basic system and its related utilities, such as report generators and installation procedures. It also contains interfaces to other vendors’ products and many useful execs to help you implement CA ACF2 for z/VM quickly.

Miscellaneous Announcements

Periodically, announcements and information are mailed to CA clients. These announcements include information on upcoming training classes and user conferences, official holiday schedules, documentation rates (for extra copies), and questionnaires. These mailings are sent to the same customers that receive standard CA ACF2 for z/VM documentation.

Index