CA-Top Secret®

Implementation: BATCH, STC and APPC Guide

Release 3.0

VSE

SP1

Second Edition, September 2000

Contents

About This Guide

Purpose

This guide describes the security features provided by CA-Top Secret for the BATCH and STC facilities and APPC. The user is provided with the necessary information for defining BATCH, STC and the APPCU interface to CA-Top Secret, the control options necessary for implementation, and available customization features.

The intended audience for this guide is security administrators responsible for job production submitted through the BATCH facility. Familiarity with the following CA-Top Secret user documentation is a prerequisite:

- *User Guide*
- *Control Options Guide*
- *Planning Guide*

Organization

Chapter	Description
1	Explains how to define the BATCH facility to CA-Top Secret.
2	Describes the implementation, reporting, and auditing-related control options for BATCH facilities.
3	Describes the security related control options for BATCH facilities.
4	Describes the security related control options for STC facilities.
5	Explains how to protect APPC conversation processing and resources.
6	Explains the requirements when CA-Top Secret is used to secure VSE libraries.
7	Explains how CA-Top Secret processes APF-authorized libraries.
Index	Provides an efficient way to locate specific material.

CA-Top Secret Publications

The following publications are supplied with CA-Top Secret:

Title	Contents
Installation Guide	CA-Top Secret installation for VSE, including: installation and maintenance steps, startup and shutdown considerations, backup and recovery procedures, and Security File conversion.
Command Functions Guide	Comprehensive reference to the TSS command and CA-Top Secret resources.
Control Options Guide	Comprehensive reference of CA-Top Secret control options.
Implementation: BATCH, STC and APPC Guide	Provides for setting up security in Batch, STC and APPC environments.
Implementation: CICS Guide	Provides for setting up security in a CICS environment.
Messages and Codes Guide	Provides all CA-Top Secret messages and codes.
Planning Guide	General guide for planning a successful CA-Top Secret security implementation and describing the latest enhancements to CA-Top Secret.
Report and Tracking Guide	Details the use of TSSUTIL, TSSTRACK, TSSAUDIT, TSSCFILE and TSSCPR and provides sample reports.
Troubleshooting Guide	Includes CA-Top Secret debugging options and facilities, information to be prepared prior to contacting Technical Support, and an explanation of customer support.
User Guide	Conceptual overview of CA-Top Secret architecture and capablilities. Also includes implementation instructions that span all facilities, including: implementation how to's, customization, and the CA-Top Secret utilities.

All guides are updated as required. Instructions accompany each update package.

Related Publications

The common component services formerly known as CA90s have been enhanced and renamed CA Common Infrastructure Services, CA-CIS. All the documentation for the services exists in the following publications supplied by Computer Associates:

Name	Contents
CA-CIS Administrator Guide	A guide for system administrators.
CA-CIS CAICCI User Guide	Describes how to use the communication protocols needed for cross-system communication.
CA-CIS Installation Guide	Tells how to install the CA-CIS.
CA-CIS Message Guide	Lists messages and codes for CA-CIS.
CA-CIS Reference Guide	Describes how to access and use the VSE common components.
CA-CIS Systems Programmer Guide	Details the programming requirements for command, security, and signon exits.

The following publications relate to CA-Top Secret and are available from Computer Associates:

Title	Operating System
CA-EARL Reference Guide	MVS/VM/VSE
CA-EARL User Guide	MVS/VM/VSE
CA-EARL Systems Programmer Guide	VSE
CA-EARL Examples Guide	MVS/VM/VSE

Command Notation

Enter the following exactly as they appear in command descriptions:

UPPERCASE	identifies commands, keywords, and keyword values which must be coded exactly as shown.
MIXEDcase	identifies acceptable command abbreviations. The UPPER-CASE letters represent the minimum abbreviation possible. The lowercase letters of the command may be entered for further clarification **Note:** In some cases, the command processor may require a more detailed abbreviation due to user-defined resource being the same as a command abbreviation. In these cases, either enter the complete command or code a national or numeric character in one of the first four positions of the user-defined resource.
underlining	identifies default operands. These operands do not need to be entered to take effect.
Symbols	"" / * # () , must be coded exactly as shown.

The following items clarify command syntax; do not type when they appear:

lowercase	indicates keyword values which you must supply.
[ ]	identifies optional keywords.
|	separates alternative keywords or values; choose one.
{ }	indicates that you must enter one of the keywords.
...	means the preceding value may be repeated more than once.

The following table indicates the appropriate command format.

Sample Command	Explanation
TSS PER(acid) DSN(dsname)	You must supply a value for the ACID and for the data set name.
MODE(DORM|IMPL|WARN|FAIL)	You must choose **only** one of the keywords.
TSS {ADDto } (acid) MCSAUTH {(INFO) } {REMove } {(MASTER)} {REPlace} {(SYS) } {(IO) } {(CONS) } {(ALL) }	You must select a command function, enter the name of an ACID, enter the MCSAUTH keyword, and select an operand from the list.

Chapter 1. Defining Batch to CA-Top Secret

CA-Top Secret interfaces directly with the BATCH facility using the VSE Standard Security Interface. There are no requirements or modifications necessary to implement CA-Top Secret Security support, except those listed by IBM for initial SEC=YES IPL processing (See, *Installation Guide*).

1.1 Defining the BATCH Facility

The following sections explain how to define the BATCH facility to CA-Top Secret.

1.1.1 Facilities Matrix

The BATCH facility is automatically recognized by CA-Top Secret as defined in the Facilities Matrix. The default attributes for the BATCH facility are:

```
INITPGM=$JOBACC      ID=B       TYPE=01

ATTRIBUTES=IN-USE,ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF

ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT

ATTRIBUTES=NOPROMPT,NOAUDIT,RES,NOWARNPW,NOTSOC,LCFCMD

ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR

MODE=FAIL

LOGGING=INIT,SEC9,MSG,SMF

UIDACID=8   LOCKTIME =000   DEFACID=*NONE*   KEY=8
```

For further information on default attributes, refer to the *Control Options Guide.*

1.1.2 Granting Access to the BATCH Facility

To CA-Top Secret, BATCH is just another facility that must be protected. Any user requiring access to BATCH must be explicitly authorized to access this facility. Users are given access to BATCH through a TSS CREATE entry when initially defining a new ACID to CA-Top Secret, or through a TSS ADD entry for an ACID already defined to CA-Top Secret. Both methods are illustrated in the following example.

```
Method 1:

     TSS CREATE(USER01) TYPE(USER) NAME(USER01) DEPT(TECHDPT)

               PASSWORD(WHISPER) FACILITY(BATCH)

Method 2:

     TSS ADD(USER01) FACILITY(BATCH)
```

1.1.3 Defining Batch Jobs

To define a batch job to CA-Top Secret, each job submitted for execution must be identified by an ACID and a password.

- The ACID is identified through information on each job card, or through a derived ACID. See Chapter 3 for the various methods of identifying ACIDs associated with batch jobs.
- Password specification is via the PWD= parameter on the ID card, or SEC=(xxxx,yyyy) on the $ JOB JECL card. CA-Top Secret automatically inserts a password on the job card for jobs submitted from other users, based upon the ACID of the submitting job, CICS user or VM user. The mechanism for automatically inserting this password is the TSS SUBACID control option. For jobs submitted from physical readers, RJE and NJE terminals, the PWD= parameter should be hardcoded on the ID card unless the ACID is created without a designated password (NOPW).

Chapter 2. Control Option Requirements

CA-Top Secret provides comprehensive implementation, reporting, and auditing controls. This chapter contains a description of control options, their functions and use as they apply to the BATCH facility. The CA-Top Secret control options are documented in the *Control Options Guide.*

2.1 Modes of Operation

CA-Top Secret supports four separate modes of operation: DORMANT, WARN, IMPLEMENT, and FAIL. Modes are assigned at several different levels:

Global The default for the entire CA-Top Secret community.
Example: **MODE(WARN)**

Facility Affects a particular facility within the community.
Example: **FAC(IMS=MODE=IMPL)**

Profile Affects a particular group of users attached to the profile.
Example: **TSS PER(PROF01) MODE(IMPL)**

User Affects a particular user within the community.
Example: **TSS PER(USER01) MODE(FAIL)**

Subsystem Affects a particular facility associated with CICS, using the CICSTEST control option.
Example: **FAC(CICSTEST=MODE=WARN)**

Resource Forces a particular resource authorization to be processed in FAIL mode. Example:
TSS PER(USER01) TERMINAL(L048T29) ACTION(FAIL)

Note: The global level is implemented via the MODE control option, or on a facility level via the MODE= suboption of the FACILITY control option. The profile, user, and resource levels are implemented via the PERMIT function of the TSS command.

The purpose and operation of each mode as applied to BATCH facilities are defined below.

DORMANT CA-Top Secret does not perform any security checking for normal ACIDs. Password specification is required only for control ACIDs or defined ACIDs with DORMPW in effect.

WARN CA-Top Secret performs full security checking for all access attempts, but does not actually fail the user. Users who cause security violations receive a warning message but are not denied access to the resource unless exceptions have been specified. Password specification is required only for control ACIDs or defined ACIDs with WARNPW in effect.

Note: If WARNPW is specified and the password entered is invalid, then the batch job or started task is failed immediately. See the chapter on implementing security for details.

IMPLEMENT CA-Top Secret provides full security for defined users and resources. Defined resources are protected and violations result in denied access. This mode allows unrestricted access to undefined users and resources (unless the undefined users access protected resources). This mode is considered a gradual implementation mode, because security can be applied selectively with little or no impact.

FAIL Requires that all users be defined to CA-Top Secret and denies all unauthorized facility or resource access unconditionally.

Note: ACTION (FAIL) fails any unauthorized access to the resource, regardless of the ACID's mode.

2.2 Control Options for BATCH

The control options used when running batch jobs are:

- FACILITY
- HPBPW
- JOBACID
- SUBACID

2.2.1 FACILITY Control Option

In general, the FACILITY control option regulates the processing of each system facility. Specifically, in regard to BATCH, it is used to authorize an ACID's access to BATCH through a TSS CREATE/ADD command function.

The FACILITY control option includes numerous operands to further restrict or control facility processing. Consult the *Control Options Guide* for a detailed explanation of these operands.

DEFACID Suboption: This suboption assigns a default ACID for a facility. If CA-Top Secret cannot derive a legitimate ACID via the JOBACID control option, then a check is made to see whether a default value can be used as the source of the ACID for the job. The DEFACID suboption is most frequently used with the BATCH facility. Default ACID options are discussed in Chapter 3, *Implementing Security for Batch*.

2.2.2 HPBPW Control Option

The Honor Previous BATCH Password (HPBPW) control option selects a number of days that CA-Top Secret honors an immediately previous or expired password for batch jobs. This allows batch jobs that execute the next day, or any selected number of days after they are submitted, to execute without abending due to expired or incorrect passwords. By default, this option is inactive.

It must be taken into consideration that using the HPBPW control option results in a security exposure, because normal expired password control or current password entry requirements are not in effect for batch jobs for the length of time specified by this option. Also, if a user suspects that the password is known and changes it, the previous password remains valid for batch jobs for the length of time specified for this option.

You may wish to use this option, particularly if delayed batch processing is normal in your organization. However, if the HPBPW control option is used, it is recommended that the length of time specified for this option be set as short as possible.

2.2.2.1 Omitting Passwords on Jobs

To avoid security exposures, you can submit jobs without having a // ID card in the JECL. When such a job is submitted, CA-Top Secret attempts to assign the job a valid ACID, based upon the SUBACID and JOBACID control options. If NJE jobs are being sent to a system also running CA-Top Secret/VSE, the derived ACID will be passed along to that system.

For an NJE job that fails identifying the passed ACID, normal password checking is performed instead. This procedure has several implications:

- If USER= is specified without PWD= on the ID card, the ACID must have the no password (NOPW) attribute.

- If USER= is specificied with PWD=, all normal password processing is enforced. However, this re-introduces the security exposure of having a password in the JOB.
- If JOBACID processing is used, the derived ACID must have the NOPW attribute.

2.2.3 JOBACID Control Option

Each BATCH job submitted for execution must be associated with an ACID so that CA-Top Secret can tell which resources it can access. There are several different methods CA-Top Secret employs to assign an ACID to a job that is submitted from an unknown source. One method for specifying this ACID to CA-Top Secret is the JOBACID control option. This control option tells CA-Top Secret which information on the JOB statement is used to define the ACID associated with the job. The various methods for deriving this ACID are detailed in Chapter 3, *Implementing Security for Batch*.

2.2.4 NJEUSR Control Option

When no other userid can be identified for NJE store-and-forward nodes, you need to define a default ACID to be used. This ACID is only used for the owner of the job or SYSOUT data on the store-and-forward node, and has no effect on the userid on the execution node.

Note: This user does not need any specific permissions.

Then, use the control option NJEUSR to specify that this user will be available for use with NJE default tokens using the format shown next.

```
TSS MODIFY NJEUSER(acidname)
```

Note: You can use NJEUSR for SYSOUT data without a defined owning ACID.

You can include the NJEUSR control option in your start-up parameters for CA-Top Secret. If you do, the control option needs to be set on the intermediate node where the job or output is being lost, and should be a valid ACID for that node. However, no checking is done at the time the control option is set to verify that the specified ACID is valid.

Note: You must restart CA-Top Secret if the NJEUSR control option is included in your start-up parameters.

2.2.5 SUBACID Control Option

The SUBACID control option is used to assign an ACID to a batch job that is being submitted from a VM, another batch job or an online user.

The SUBACID control option offers two methods of deriving the ACID assigned to a batch job submitted online:

1. The jobname
2. The submitter's ACID

Note: Any other method of deriving the ACID, requires coding through the Installation Exit which is documented in the *User Guide*.

How the SUBACID control option is set determines which source is used, as well as the number of characters from the jobname or user's sign-on ACID that is used to derive the batch job's ACID.

SUBACID(J,n) Tells CA-Top Secret to use the first *n* characters of the submitted job's jobname as the ACID.

SUBACID(U,n) Indicates that the first *n* characters of the submitter's user ID or ACID becomes the ACID for the job.

The default attribute for the SUBACID control option is **(U,7)**. In other words, if CA-Top Secret is unable to derive a valid ACID from the USER= parameter in the JCL, it attempts to run the job under the submitter's ACID by default.

Note: If the USER= parameter on the ID statement or SEC=(xxxx,yyyy) on the * $ $ JOB JECL statement has already been coded, then that value will be the ACID under which the job executes, overriding whatever the SUBACID control option indicates.

Also, if the USER= parameter on the JOB statement has already been coded, then that value will be the ACID under which the job executes, overriding whatever the SUBACID control option indicates.

2.2.6 Automatic Password Insertion

A potential security exposure is introduced if the job submitter has to specify a password via the PWD= parameter. Therefore, if the batch job's ACID that was derived via the SUBACID control option requires a password, then the lack of such a password would cause a security violation.

To avoid this, CA-Top Secret **automatically** supplies the required password for any SUBACID derived ACID. Essentially, this password is visible only to CA-Top Secret.

No password will be inserted if the POWER(VERIFY) control option is in effect. For additional security, passwords will not be kept on spool files either.

Note: The user can override the CA-Top Secret automatic password insertion (via the SUBACID control option) by coding the USER= parameter on the // ID statement.

The user can even submit a batch job under a different user ID by changing the value of the USER= parameter on the // ID statement. To do so, the user submitting the job must have been permitted use of the ACID.

Chapter 3. Implementing Security for Batch

This chapter describes the options provided by CA-Top Secret for implementing BATCH security. The applicable control options, ACID specification, and required authorizations are discussed.

CA-Top Secret views BATCH as just another facility that must be protected and authorized for use. To provide this protection, each batch job must be associated with an ACID and a password so that CA-Top Secret can tell which resources it can access and how they can be accessed. To CA-Top Secret, a batch job's ACID is exactly like a user ACID, in that it has an associated user record with a set of specific access authorizations. In fact, it is usually assigned a TYPE(USER) parameter specification when created. All system entry restrictions that can be designated for a user ACID are available for a batch job ACID, such as facility, source of origin, and CPU restriction.

CA-Top Secret provides many different methods for identifying the ACID associated with a batch job. This chapter describes additional selection options for the basic identification methods.

3.1 Online Versus Non-online Submission

A batch job can enter the system through either an internal or external reader. When the job is submitted through an internal reader it is described as being submitted **online**, even when the job is being submitted by an unattended batch job. All jobs submitted through an internal reader are processed by CA-Top Secret at both job submission and job execution (initiation). However, if a job is submitted on a node that is not running CA-Top Secret and is routed for execution to a node that **is** running CA-Top Secret, none of the submission time features of CA-Top Secret will be available for this job. Therefore, in this instance, CA-Top Secret would not insert USER= or PASSWORD= on the job card.

Therefore in this instance, CA-Top Secret would not inherit security information. External readers are devices such as an RJE station or card reader. For an external reader, there is not available time for submission processing and no ACID is pre-associated with the job unless USER= was hardcoded on the // ID card. Jobs sumitted from VM to an external virtual reader will inherit the ACID of the submitting userid, based on the SUBACID control option. For options and processing of jobs submitted through an external reader, see 3.3, “Assigning ACIDs to Batch Jobs Not Submitted Online” later in this chapter.

3.2 Assigning ACIDs to Batch Jobs Submitted Online

There are two methods for assigning ACIDs to batch jobs submitted online. Both methods apply to all jobs submitted from any online facility or by other batch jobs.

The methods for deriving and ACID are:

1. Through the SUBACID control option.

   The SUBACID control option automatically inserts a derived ACID into the batch job's Power control blocks. The value inserted (which is only visible to CA-Top Secret) is the ACID under which the batch job will run. See 2.2.5, “SUBACID Control Option” for instructions on specifying this control option.

2. Directly through the USER= parameter on the // ID statement.

   If the USER= parameter on the // ID statement has a value submitted with the job, this value becomes the job's designated ACID and CA-Top Secret does not check the SUBACID control option. In fact, when you specify an ACID in the USER= parameter of the // ID statement, it overrides the SUBACID control option.

3.3 Assigning ACIDs to Batch Jobs Not Submitted Online

The information in this section applies to all jobs that are submitted in a BYPASS environment, from a non-CA-Top Secret node, or through an external reader. In all of these cases, an ACID must be assigned to the batch job.

There are several different methods for assigning an ACID to a batch job being run from a physical reader, RJE, or non-CA-Top Secret secured NJE node. The method you select applies to all batch jobs. The processing order for a given batch job follows the sequence in which the methods are listed below.

Note: Once an ACID is assigned, the rest of the methods are skipped.

1. An ACID is assigned by hardcoding it on the // ID card, using the USER= parameter.
2. An ACID is assigned by propagating the submitting userid, if SEC=(xxxx,yyyy) was specified on the Power * $$ JOB JECL statement.
3. An ACID is assigned using the JOBACID control option.
4. The global default is used as the ACID.

3.3.1 Using the NODES Resource Class

When a batch job is submitted on a secure VM system and routed via RSCS to MVS using an RSCS-to-JES NJE link, CA-Top Secret can assign the submitting VM userid to the batch job. To enable this, use the NODES resource class with the PERMIT command function shown below.

```
TSS PER(ALL) NODES(nodename.USERJ.) ACCESS(CONTROL)
```

Note: The NODES resource class replaces the VMJESLNK control option. For example, VMJESLNK(ABCD) is replaced by the syntax shown below.

```
TSS PER(ALL) NODES(ABCD.USERJ.) ACCESS(CONTROL)
```

3.3.2 Using the JOBACID Control Option

In general, the JOBACID control option is used to tell CA-Top Secret how to obtain the ACID. This option provides a standard source for deriving the ACID without having to change the existing JCL. JOBACID is used when no user parameter is provided, or when the provided user is not a defined ACID.

Note: If the JOBACID options listed below are not suitable for your site, then consider using the Installation Exit.

Specifying the JOBACID Control Option: The security administrator can use the JOBACID control option to indicate which field on the job card must be used as the ACID. There are several different JOB statement parameters from which CA-Top Secret can derive the ACID for the job:

1. A specified field (1-8) of the accounting parameter.

   To illustrate, specifying **JOBACID(A,1)** indicates that the first parameter of the **accounting** field must be used as the ACID. For example, **ADM100** is the ACID when specified as:

   ```
   // JOB ADMIN ADM100
   ```

   The default for this control option extracts the ACID from the first field of the DOS JECL accounting parameters. For example JOBACID(A,1) indicates the first accounting field is to be used—generating an ACID of ADM100. JOBACID(A,1,3) still indicates that the first accounting field is to be used for an ACID name, but the ACID name should be taken from the third position in the field—generating an ACID of M100.

For sites that use sub-accounting, CA-Top Secret treats the slash (/) and the dash (-) as delimiters of an accounting number used as an ACID. For example, JOBACID(A,1) indicates that in the following account specifications, the ACID is ADM200:

```
ADM200-SMYTHE
ADM200/JUN82
```

2. A specified number of characters (1-8) starting from the left of the jobname.

   Here, specifying JOBACID(J,5) indicates that the first five characters of the jobname be used as the ACID. For example, BUG24 is the ACID when specified as:

   ```
   // JOB BUG24MAY ADM100
   ```

3. A specified number of characters (1-8) of the source reader's name (other than INTRDR).

   For example, specifying JOBACID(R,8) indicates that the entire reader name be used as the ACID.

If the ACID is not defined to CA-Top Secret, it is failed immediately unless a default ACID option (DEFACID) was implemented. How to define a default ACID is covered later in this chapter.

To deactivate JOBACID processing entirely, specify:

```
JOBACID(U,x)
```

Replace *x* with any numeric value; 7 is recommended.

Overriding the JOBACID Designation: When the USER= parameter is coded on the // ID statement or if the submitting user is eligible to be propagated, it overrides any ACID derived by the JOBACID specification. If the USER= parameter is coded, its value is used as the ACID assigned to the job, even if the JOBACID specifies another source for the ACID. For example, when // ID USER=ADA103 is coded on the // ID statement and JOBACID(A,1) is specified, the job's ACID becomes ADA103, **not** the account number.

If the USER= parameter or propagated user is an undefined ACID, CA-Top Secret tries to derive an ACID using JOBACID before attempting to use the BATCH facility DEFACID.

3.3.3 Assigning the Default ACID

If CA-Top Secret is unable to derive an ACID from the USER= parameter or the JOBACID control option, it next checks to see if a default ACID has been assigned. An established default for the ACID allows jobs that otherwise trigger a security violation to run under minimum access authorizations.

Note: If the submitting ACID was propagated from the sending node it will override the JOBACID.

CA-Top Secret provides two methods of establishing default ACIDs for batch jobs using the DEFACID suboption of FACILITY. These methods are:

1. Establishing a global default ACID.
2. Using the job source as the default ACID.

3.3.3.1 Establishing a Global Default ACID

The DEFACID suboption of FACILITY can be used to establish a global default ACID to assign to batch jobs. The MSCA or security administrator enters the following control option into the Parameter File:

```
FAC(BATCH=DEFACID(BATDEF))
```

The BATDEF ACID must be defined with the BATCH facility, have no password specification, and be permitted the appropriate access authorizations. An example showing the creation of a BATDEF ACID is shown below.

```
TSS CREATE(BATDEF) TYPE(USER) NAME(BATDEF) DEPT(SOFTDEV)
          FACILITY(BATCH) PASSWORD(NOPW,0)
```

3.3.3.2 Using the Job Source as the Default ACID

The security administrator can assign a default ACID derived from the name of the physical reader, RJE, or NJE node from which the job is being submitted instead of assigning a single global default. Using the job source name as the default ACID allows you to control remote job submission without changing your JCL. To implement this option, the security administrator must enter the following control option into the Parameter File:

```
FAC(BATCH=DEFACID(RDR*TERM))
```

Once the previous entry is made in the Parameter File, the reader's default ACID can be created. For example, to define the default ACID for RJE TERMINAL R10.RD1, the following command is entered with the appropriate access authorizations:

```
TSS CREATE(R10@RD1) TYPE(USER) NAME('DEFAULT-LOC-10')
     DEPT(DEPTX) PASSWORD(NOPW,0) FAC(BATCH)
     SOURCE(R10.RD1)
```

- The appropriate access authorizations can be tailored to fit the requirements of the types of jobs typically submitted from that particular device. The ACID name generated would be R10@RD1. This convention provides more control over remote job entry security. To use RDR*TERM processing, ACIDs must be created for each remote reader.
- Using the SOURCE parameter provides an extra measure of security protection, since this ACID is only valid for jobs submitted from R10.RD1. Also note that if the R10 terminal is defined, then TERMINAL(R10.RD1) access must be given to the ACID through the PERMIT command function.

3.3.4 Protecting Card and Remote Readers

For jobs being submitted from a physical reader, RJE, or NJE terminals, the submitter's password must be manually coded in the PWD= parameter on the ID statement unless the associated ACID does not require a password. Since there is no way for CA-Top Secret to insert the password on remote or physical readers, the following suggestions are offered for implementing controls in this situation.

- Use a TSS CREATE/ADD entry with the batch job's ACID to specify the PASSWORD(NOPW) attribute. In the following example, the JOB1 ACID does not require a password, so CA-Top Secret does not check for password authorization. JOB1 executes without coding a password in the ID statement, thus avoiding security violation.

  ```
  TSS CREATE(JOB1) TYPE(USER) NAME(JOB01) DEPT(OPERAT)
        FACILITY(BATCH) PASSWORD(NOPW,0) SOURCE(R10)
  ```

  Note: The disadvantage of specifying the PASSWORD(NOPW) attribute is that no CA-Top Secret security is enforced for the person who placed the JOB in the reader.

- To allow some control of job submission from remote readers, use the TSS CREATE/ADD entry to attach the TERMINAL attribute to the ACID associated with the batch job. This restricts the job's execution to that device.

  Adding the SOURCE parameter to the ACID also restricts the job's initiation to a specific terminal or reader as shown in the example below.

  ```
  TSS ADD(JOB1) SOURCE(R10)
  ```

- Another option is to monitor job submission through the AUDIT attribute. A security administrator, with the correct administrative authorities, can add the AUDIT attribute to the ACID associated with the batch job. This provides a trail of all activity related to this ACID in the Audit/Tracking File. Refer to the *Report and Tracking Guide* for further information.

3.4 Explicit Password Specification

If explicitly specifying the password on the JOB statement is acceptable from a security standpoint, this can be done using the PWD= keyword as shown in the following example.

```
// JOB REPORT ACC158
// ID USER=TCH707,PWD=G3T1Q
```

3.5 Changing a Password on a Job Card

To change a password through BATCH, enter:

```
// JOB REPORT ACC158
// ID USER=TCH707,PWD=G3T1Q,NEWPW=FOR2D
```

3.5.1 ACID Propagation

If USER= and PWD= are omitted from the ID card, Power will propagate the submitting ACID to the job. That is, CA-Top Secret will let Power associate the submitting ACID with the job and will not require an ID card.

Propagation will take place over various Power NJE nodes. Therefore is it possible to have a job submitted on one node and executed with the submitting ACID on a different node without having USER= or PWD= appear in the job.

By default, the ACID will propagate; however, by using the PROPCNTL resource class you can suppress ACID propagation, whether or not there are any permits for those resources. This is useful when jobs are submitted from a multi-user online system and your site intends to prevent the region ACID from being associated with batch jobs submitted from the online system.

The following example suppresses propagation for any ACID starting with CICS.

```
TSS ADD(anyacid) PROPCNTL(CICS)
```

To allow propagation to occur after issuing the above command, issue the following command:

```
TSS REM(anyacid) PROPCNTL(CICS)
```

To suppress all propagation, use:

```
TSS REPL(RDT) RESCLASS(PROPCNTL) ATTR(DEFPROT)
```

Note: This does not affect the normal CA-Top Secret SUBACID processing.

Chapter 4. Implementing Security for STC

Whenever a job initiates in a VSE partition, a period of time exists where it is impossible to determine what ACID will be logged on. During this time, it is possible to execute security related items like LIBDEF processing and programs. An example of this is the process which takes place during dynamic partition startups when the partition is created.

Since this is a potential security exposure, CA-Top Secret secures this processing time using the STC facility (Started Task Processing).

The actual time a program or job executes under the STC facility ends whenever CA-Top Secret has identified and logged on the correct ACID as specified or inherited in the executing JCL.

STC processing in CA-Top Secret is linked to a concept of procedure names (PROCs). The name of the PROC being invoked in VSE is linked to the name of the partition where the VSE JOB executes.

For example, a job executing in partition BG will have an STC PROC named FORSECBG associated with it, a job in Dynam Partition C1 would have FORSECC1 assigned to it.

The dynamically-created PROC names should be added to the CA-Top Secret Started Task Table (STC) and when added, the security administrator can choose to assign what CA-Top Secret ACID should be logged on while the STC task is running. For example,

```
TSS ADD(STC) PROC(FORSECBG) ACID(DUMMY)
```

When a job enters the BG partition, CA-Top Secret dynamically assigns the PROC name FORSECBG to it. It then finds the name in the STC table and in the above case attempts to log on an acid named DUMMY under the STC facility.

It is important that in this example, DUMMY is granted access to the STC facility and does not have any logon password.

Naturally, VSE has many different partition names, so it is impossible to define each FORSECxx name in the STC table. A more generic method of attaching the PROC names to an ACID can be done like this:

```
TSS ADD(STC) PROC(FORSEC**) ACID(DUMMY)
```

In this example any job starting in any VSE partition has the ACID DUMMY logged on during STC processing.

STC logon processing does not produce any logon messages on the console unless CA Top-Secret is unable to locate a match in the STC table, in which case the default BYPASS acid will be logged on.

Use the TSS LIST(STC) DATA(ALL) command to list the contents of the STC table.

4.1 Prompting for an Assigned ACID

This option provides an additional form of protection by forcing the operator to supply the correct ACID for the STC before it is allowed to execute.

Note: If the ACID assigned to the STC is defined with a password, then the user receives a second prompt for the password.

To allow operators, systems programmers, and production control personnel to start STCs that prompt for the ACID under which they are to run, use this model:

```
TSS ADD(STC) PROC(stcname) ACID(PROMPT) [STCACT]
```

Now, when an operator starts this VSE JOB the operator is prompted for the ACID supplied on the CREATE command and under which the STC is to run. If the ACID has a password, the operator is also prompted to provide it. If the ACID/password combination is valid, STC runs under the authorizations associated with this ACID. If an undefined ACID or an incorrect password is used, then the STC fails immediately.

When the STCACT attribute is also designated, as shown in the previous TSS ADD entry, CA-Top Secret also prompts the operator to supply a user ACID and password in addition to the STC's ACID and password (if required).

4.2 Bypassing Security Checking

To define a specific STC on which no CA-Top Secret security checking is performed, a single TSS ADD entry is needed:

```
TSS ADD(STC) PROC(FORSECBG) ACID(BYPASS)
```

The BYPASS keyword specified in the acidname field means that started task PROC FORSECBG is executed without any security checking. To make this type of entry, MISC9(STC) authority is required.

Note: The STCACT attribute cannot be specified for STCs in bypass mode.

4.3 Security Options for Undefined STCs

There are five options CA-Top Secret uses to treat undefined STCs. All are specified through a TSS ADD(STC) PROC(DEFAULT) entry, and the option you select will apply to all undefined STCs.

Default STC options are:

- Assign a default ACID.
- Prompt undefined STCs for user ACID.
- Fail all undefined STCs.
- Bypass security for all undefined STCs.
- Run as an undefined user.

4.3.1 Assigning a Default ACID

To establish a default ACID, the security administrator enters:

```
TSS ADD(STC) PROC(DEFAULT) ACID(acidname)  [STCACT]
```

All undefined STCs then run under the authorities associated with this ACID. If an executing STC makes a resource access request that exceeds the authority of this ACID, a security error results. The advantage of this approach is that it can be used to allow basic security protection with a minimum impact on productivity.

4.3.2 Prompting for the ACID

To force the operator to supply an ACID for an undefined STC, the security administrator enters:

```
TSS ADD(STC) PROC(DEFAULT) ACID(PROMPT)  [STCACT]
```

The STC then executes under the authorities designated for this ACID by the console operator.

4.3.3 Failing All Undefined STCs

To instruct CA-Top Secret to fail all undefined STCs, the security administrator enters:

```
TSS ADD(STC)  PROC(DEFAULT)  ACID(FAIL)
```

Generally, selection of this option is only advisable in environments in which a systematic and comprehensive analysis of started task security requirements has been made. Before using FAIL, ensure that all STCs are explicitly defined.

4.3.4 Bypassing Security Checking

To instruct CA-Top Secret to allow all undefined STCs to bypass security checking, the security administrator enters:

```
TSS ADD(STC) PROC(DEFAULT) ACID(BYPASS)
```

There must be careful consideration of the potential security exposures before this option is selected. It is recommended that this option be used only while running tests.

4.3.5 Running as an Undefined User

To instruct CA-Top Secret to allow undefined STCs to run as an undefined user, the security administrator enters:

```
TSS ADD(STC) PROC(DEFAULT) ACID(UNDEF)
```

Whatever approach has been selected for handling undefined users, also governs how undefined STCs are handled. This approach could also be used to run undefined STCs under a default ACID.

Chapter 5. Securing APPC Sessions

The APPC interface for CICS Transaction Server (TS) 1.1 facilitates the establishment of communication sessions between partner programs on other systems and within the same system. This allows for the sharing of work, data, and services between systems and across networks. Each session has the potential for security exposure, and the need for security controls must be determined.

An APPC conversation consists of both a sending and receiving logical unit, or LU. These LUs represent VTAM nodes or other points of entry into the network. Security at the LU - LU level can be provided through a combination of CA-Top Secret and CICS TS 1.1 options. By using the APPCLU Record Table you can define which LUS can be used for an APPC conversation and what, if any, security informat ion is required for that link to take place.

5.1 Implementing APPC Session Security

There are two basic steps for using CA-Top Secret to secure APPC conversations. The are:

1. Specify which LUS can establish sessions by defining remote and partner LUS to the CA-Top Secret APPCLU record.
2. Activate APPC security options within CICS TS 1.1.

5.1.1 Maintain APPCLU Record Table

After reviewing your security requirements and determining which LUs will be used for APPC conversations and what level of security should be maintained for those conversations, you need to identify authorized LU links to CA-Top Secret. The LINKID keyword is used to identify the authorized LU links.

The APPCLU record maintains a list of LINKIDs (one LINKID per LU-LU connection) in the following format:

```
netid.localLU.partnerLU
```

netid The name of the network on which the local LU resides.

Note: You should supply the same netid specified by the netid= statement in the VTAM ATCSTRxx member.

localLU The VTAM name of the local LU.

partnerLU The VTAM name of the partner LU.

You do not have to specify all three values, and you do not have to provide the full names for each value. However, the CA-Top Secret algorithm is designed to search for the best match. Therefore the most explicit entry (i.e., the longest) which best matches the requested resource name will be used. The following entries begin with the most explicit example and end with the least explicit example.

```
NET1.SYSTEMA.SYSTEMB
NET1.SYS
NET
```

5.1.1.1 LINKID Keywords

The LINKID keyword is used to maintain the appropriate entry to the APPCLU record for APPC conversations. Each LINKID entry can be associated with one or more of the following keywords:

SESSKEY An 16-byte hexadecimal encryption key unique to each LU-LU authorized link. If additional verification is required for a session to be established between the two LUs, the SESSKEY is used to encrypt and decrypt connection messages. If the exchange is satisfactory to both LUs, the session is established.

Note: If you supply a SESSKEY, you must also indicate an INTERVAL.

SESSLOCK Used to single out which LUs cannot establish sessions.

INTERVAL Must be supplied if you indicate a SESSKEY. This keyword determines how frequently SESSKEY must be updated. The value can range from 1 to 32767.

CONVSEC Determines what, if any, additional identification needs to be provided and verified for an LU-LU session to be established. Select one of these values:

NONE No security information is required.

CONV Security information is required. A userid and password need to be verified.

Note: CONV overrides the SECACPT value specified in the VTAM APPL statement.

ALREADYV Indicates that a userid and password have already been verified by the partner LU and that the path is trusted.

PERSISTV Indicates that a userid and password must be verified on the first request; for subsequent requests only the userid is verified.

AVPV Supports both ALREADYV and PERSISTV values; the security type used depends on the incoming request.

Note: CONVSEC security does not apply if you are using a VTAM release prior to 3.4.

To add an entry to the APPCLU record, use the following syntax:

```
TSS ADD(APPCLU) LINKID(netid.localLU.partnerLU)
    [SESSKEY(sesskey) INTERVAL(nnnnn) [CONVSEC(operand)]]
    - or -
    [SESSLOCK]
```

These entries can later be REMOVEd through the TSS REMOVE command and updated using the TSS REPLACE command.

5.1.1.2 Examples

In the following examples LSCA01 is responsible for maintaining the APPCLU record.

Example 1: LU01 and LU02 can establish sessions but the SESSKEY, which is ABCDEFGH, must be verified first. That SESSKEY must also be updated every 45 days. To accomplish this, LSCA01 enters:

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.LU02) SESSKEY(ABCDEFGH) CONVSEC(CONV)
    INTERVAL(45)
```

Example 2: LU01 **cannot** establish a session with LU03. To prevent this, LSCA01 enters:

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.LU03) SESSLOCK
```

Example 3: The SESSKEY used to verify that LU02 and LU03 can establish sessions has to be updated every 30 days. Previously, the SESSKEY only had to be changed every 60 days. To institute this change, LSCA01 enters:

```
TSS REP(APPCLU) LINKID(SYS1.LU02.LU03) INTERVAL(30)
```

Example 4: Later, LSCA01 can issue a TSS LIST command to review and verify the changes he has made. The command would look like this:

```
TSS LIST(APPCLU) DATA(SESSKEY)
```

In response, CA-Top Secret would provide the following information:

```
ACCESORID   = *APPCLU*   NAME         = APPC/MVS LU SECURITY
LINKID      = SYS1.LU01.LU02
  TOT VIOS  =     0      MAX VIOS     =     0  LINK STATUS=AVAILABLE
  CONVSEC   = CONV       VIOLATIONS   =     0
  SESS KEY  = ABCDEFGH
    EXPIRES = 92-30-11   INTERVAL     =    45
LINKID      = SYS1.LU01.LU03
  TOT VIOS  =     0      MAX VIOS     =     0  LINK STATUS=UNAVAILABLE
  CONVSEC   = N/A        VIOLATIONS   =     0
  SESS KEY  = N/A
    EXPIRES = N/A
```

To view a single entry in the APPCLU record, LSCA01 could issue the following command:

```
    TSS LIST(APPCLU) LINKID(SYS1.LU02.LU03)
```

When only a single entry is listed, the SESSKEY is not displayed. To view the SESSKEY associated with a particular LINKID, you **must** view the entire APPCLU record using the syntax demonstrated in the previous example.

5.1.1.3 Authority

To maintain the APPCLU record, the security administrator needs to have MISC2(APPCLU) authority. To view SESSKEYS, he must also have DATA(SESSKEY) authority.

5.1.2 Activate APPC Security in CICS TS

APPC session security is activated in CICS TS 1.1 through two security options that are checked at session bind time. XAPPC is a system initialization option that requests CA-Top Secret to return security record information concerning the session partners to the requester, in this case CICS. Set this option to YES to activate the external security process.

BINDSECURITY is a resource definition parameter that can be associated with either a CONNECTION or TERMINAL definition in CICS. The parameter should be set to YES to activate the extraction of the session key from the APPCLU security record.

For additional information, please refer to the appropriate IBM CICS TS manuals concerning the definition of the above options and the activation of APPC bind-time security requirements.

Chapter 6. VSE Library Security

This chapter is designed to help clarify the requirements when CA-Top Secret is used to secure all VSE library accesses, when SEC=YES has been specified in the IPL Procedure.

6.1 What is Library Security?

Library security is IBM's implementation of security in native VSE, which allows a security administrator to control accesses to Libraries, Sub-Libraries and members, using a predefined table called DTSECTAB.

When VSE is IPLed with SEC=YES, the VSE librarian routines, issues Security SVC calls (SVC 108), passing an Access Parameter List (APL) in R1. This APL is then verified against the user logged on, and the pre-defined DTSECTAB.

Various access schemes can be used with DTSECTAB, but for now we will only look at the one called UACC, or Universal Access Method.

It is important to understand that CA-Top Secret does not replace or modify the IBM librarian security logic, it simply front ends the logic, making the DTSECTAB obsolete.

SEC=YES in the IPL procedure, indicates library security is needed. The Systems Programmer or Security Administrator needs to code a sample DTSECTAB (we suggest you use the one supplied by IBM as a sample), which provides sufficient authority to access the libraries needed to bring up VSE, CA-Top Secret, and CA-CIS. A sample of such a DTSECTAB has been described in the CA-Top Secret *Installation Guide*. It is very important that any Library or Sub-Library defined in this DTSECTAB, has UACC=CONN specified, otherwise VSE will not issue Security checks against the library later on.

During the startup of the CA-Top Secret server partition, control is switched from the DTSECTAB to the definitions found in the CA-Top Secret database.

6.2 What Takes Place During a Member Level Check?

Assume we have a VSE library called PAYLIB which contains a member called LIBDEF.PROC, located in the VSE sublibrary PAYLIB.PROC. Access to LIBDEF.PROC needs to be protected against accidental updates. The VSE library was defined to standard labels like this:

```
// DLBL PAYLIB,'VSE.PAYROLL.LIBRARY'
```

An ACID called SYSA, now executes a Batch LIBR program, to Catalog LIBDEF.PROC into PAYLIB.PROC.

```
// EXEC LIBR
ACC S=PAYLIB.PROC
CATALOG LIBDEF.PROC....
* Sample
/*
```

CA-Top Secret now enters a 4 step Security validation logic.

1. First access to the dataset called VSE.PAYROLL.LIBRARY is checked. Since VSE accesses the dataset called VSE.PAYROLL.LIBRARY, the Security Database is first checked for ownership of that Resource. To grant SYSA access to the dataset, the following permit could be issued:

   ```
   TSS PERMIT(SYSA) DSN(VSE.PAYROLL.LIBRARY) ACCESS(UPDATE)
   ```

   An optional PERMIT for access to the VOLSER containing the library could also be performed here.

2. If the DSN security check passes, access to the library named PAYLIB is checked. This ensures that no one can attempt to access the specific library, using an over-riding // DLBL or VSE library name, consisting of the VSE dataset name, with the library name added to the end. In this example, the resource name build will be:

   ```
   VSE.PAYROLL.LIBRARY.PAYLIB
   ```

   First the library must be defined as protected:

```
TSS ADD(VSEDEV) VSELIB(VSE.PAYROLL.LIBRARY)
```

This specifies to CA-Top Secret, that any access to a library in VSE dataset VSE.PAYROLL.LIBRARY should be considered protected, regardless of what library name has been specified in the DTF name. The access to that library could be granted using the following permit command:

```
TSS PERMIT(SYSA) VSELIB(VSE.PAYROLL.LIBRARY.PAYLIB) ACCESS(UPDATE)
```

Note: Generic prefixing is allowed, but not recommended for VSE library rules, it is important that some type of control is kept of the last part of the resource name, which is the actual VSE Librarian Library name used (or DTF name).

3. If the VSELIB security check passes, access to the VSE Sub-Library called PAYLIB.PROC is checked. Access to the Sub-Library can be granted, using the following permit command:

   ```
   TSS PERMIT(SYSA) VSESLIB(PAYLIB.) ACCESS(UPDATE)
   ```

4. And finally, access to the member called LIBDEF.PROC in PAYLIB.PROC will be checked. The access to this library, can be granted using this permit command.

   ```
   TSS PERMIT(SYSA) VSEMEMBR(PAYLIB.PROC.LIBDEF.PROC) ACCESS(READ)
   ```

In this example, the access would FAIL. ACID SYSA had update access to all libraries in the PAYLIB Library, but only READ access to the member called LIBDEF.PROC in PAYLIB.PROC.

To summarize the VSE library security access in CA-Top Secret, all accesses gets validated in this order:

1. DSN(dataset),
2. VSELIB(dataset.dtfname),
3. VSESLIB(VSE Sub-Library)
4. VSEMEMBR(VSE library member)

The ACID must pass all 4 checks before being granted access to the resource. If any of the resources are not owned or known to CA-Top Secret, access is assumed and the ACID will skip to the next security check in the sequence.

Chapter 7. APF Authority Checking

This chapter clarifies the requirements for CA-Top Secret to process RACROUTE security checks. These security checks are authorized functions.

7.1 What is APF Authority?

APF authority is IBM's implementation in OS/390 of security to certain operating system functions, such as entering Key 0 or Subsystem mode.

The OS/390 operating system checks where programs are loaded from before it allows these functions to execute, this prevents function calls from programs loaded from non-APF libraries. RACROUTE was designed to use this built-in feature, as an extra added security against certain authorized calls being issued. The IBM *OS/390 RACROUTE Macro Reference* guide, documents these calls as being required to be in KEY 0 or Supervisor mode, before they are allowed to execute.

7.2 How is APF Implemented in VSE/ESA?

In VSE/ESA 2.3 and below, CA-Top Secret **does not** allow user-defined RACROUTE calls to issue these restricted calls. Users are recommended to use the provided #SECUR macro call instead. In VSE/ESA 2.4 and above, IBM ships RACROUTE as a part of the OS/390 Services Library, so extra security has been added to allow the authorized calls to pass.

In VSE/ESA 2.4, CA-Top-Secret considers the following libraries to be APF-authorized.

IJSYSRS (and any sublibrary)
PRD1 (and any Sublibrary)
PRD2 (and any Sublibrary)
CAILIB (and any sublibrary)

This includes any sublibrary in any library called APFLIB, such as USER.APFLIB and any phases loaded into the System Directory List (SDL) during or after an IPL.

By default, when a partition or task initiates, it is considered APF-authorized in VSE. After the initialization, CA-Top Secret will monitor the task for any program load taking place. If a program is detected being loaded from a library NOT covered by the above rule, the task will be marked as NOT APF-authorized, and the list of Restricted RACROUTE calls will be denied if they are called.

7.3 How Does This Affect My VSE System?

In normal day-to-day processing this does not affect VSE systems at all; since all the products provided by IBM and Computer Associates are covered by the default rule listed above.

If you install any other products that issue RACROUTE calls, all of the programs which are loaded into that partition or task must be installed in a library, sublibrary, or SDL that is APF-authorized.

Note: Due to the nature of APF and the dependencies of various VSE libraries, it is recommended that VSE/ESA 2.4 users IPL with SEC=YES and protect those dependent libraries from being updated. Use CA-Top Secret VSE library and sub-library protection to do this.

Index

User Registration Form

If you are interested in registering as a user of Computer Associates software, please complete the information below. Send it to the following address:

Computer Associates International, Inc.
ATTN: User Registration
One Computer Associates Plaza
Islandia, NY 11749

Registration entitles you to receive such items as:

- newsletters
- product release announcements
- information about User Groups
- information about CA conferences
- client questionnaires

You *do not* have to be the primary contact for CA software within your company or organization. All you have to be is interested in CA software, how it is used, and where it is headed.

Product(s): _____

_____

_____

_____

Site ID: _____
(Enter UNKNOWN if you do not know your Site ID.)

Name: _____

Position: _____

Company or organization: _____

Address: _____

Address: _____

Phone No.: _____

Date: _____

I would like additional information on: _____

Demand Analysis Request Form

Please use this form to make suggestions for product improvement. Send it to the following address.

Computer Associates International, Inc.
ATTN: Demand Analysis Requests
One Computer Associates Plaza
Islandia, NY 11749

Be sure to print your name and address below if you would like a reply.

Name: _____

Position: _____

Company or organization: _____

Address: _____

Address: _____

Phone No.: _____

Date: _____

Is DAR from a User Group? (Yes/No) ___ User Group ID: _____

Priority: _____

Description of Requested Item:

_____

_____

_____

_____

_____

_____

_____

_____

_____

CA Account Manager:	CA Office:
Input by:	Date:

Reader Comment Form

CA-Top Secret Implementation: BATCH, STC and APPC Guide

Release 3.0 VSE

Document Number: R101TS30IAE

Please use this form to tell us how you use this manual and to make suggestions for improvement. Send it to the following address. (To request additional publications, call 1-800-841-8743 or check the CA home page (http://www.cai.com) on the Internet. To ask questions about the functionality of CA products, contact your CA representative.)

Computer Associates International, Inc.
ATTN: Reader Comment Form
One Computer Associates Plaza
Islandia, NY 11749

Be sure to print your name and address below if you would like a reply.

Name: _____

Position: _____

Company or organization: _____

Address: _____

Address: _____

Phone No.: _____

Date: _____

Years of experience with this CA product: _____

Overall Rating

	Good	Fair	Poor	Why?
Accuracy				
Organization				
Clarity				

Reference Aids

	Good	Fair	Poor	Why?
Contents				
Index				
Glossary				
Reference to Other Manuals				

Clarity

	Good	Fair	Poor	Why?
Instructions and Procedures				
Examples				
Graphics				

How Manual Is Used:

How do you use this manual in your job?

How often do you use this manual in a week?

Suggestions:

What do you like about this manual?

What do you dislike about this manual?

What would you like us to change?

Additional Comments: