RAKF Version 1 Release 2 Modification 0

A Security System for MVS 3.8j

User's Guide

April 2011

```
*********************************************************************
*                                                                   *
*    © Craig J. Yasuna                                              *
*      See Appendix for detailed Copyright Information              *
*                                                                   *
*********************************************************************
*                                                                   *
*    RAKF is based on the ESG Security System                       *
*    written by Craig J. Yasuna               (Mar 1991)            *
*    adapted to MVS 3.8J: A. Philip Dickinson (Aug 2005)            *
*                         Phil Roberts        (Apr 2011)            *
*                         Jürgen Winkelmann   (Apr 2011)            *
*                                                                   *
*********************************************************************
*                                                                   *
* This document applies to RAKF Version 1 Release 2 Modification 0  *
* with PTFs RRKF001, RRKF002, RRKF003 and RRKF004 installed.        *
*                                                                   *
*********************************************************************
*                                                                   *
* The information in this User's Guide is adapted from the original *
* ESG Security System's documentation as prepared by Sam Golob in   *
* 1991. For historical reasons the original documentation has been  *
* retained and can be found as member $$$$$DOC of HLQ.SAMPLIB (HLQ = *
* High Level Qualifier of RAKF libraries as chosen at installation  *
* time) after RAKF installation.                                    *
*                                                                   *
*********************************************************************
```

Table of Contents

RAKF User's Guide

Introduction

The RAKF Security System is a RACF™-like MVS System Authorization Facility (SAF). RACF™ Version 1.7 facilities are emulated, except for the RACF™ database. Two tables, the users and the profiles table, are kept in storage. The actual security verifications are made by ICHSFR00, using these two tables. The formats of these users and profiles tables are compatible with RACF™ database entry data.

Protection is achieved by routing all operating system or vendor product security calls (including RACDEF, RACINIT, and RACHECK) through the ICHSFR00 RACROUTE interface. ICHSFR00 contains the real verification code. The RAKF Security System is designed to force "one point of handling" for all security calls. ICHSFR00 processes the various kinds of security calls in a standard way that is (mostly) documented by IBM. ICHSFR00 refers to the installation-coded user and profile in-core tables, to make its judgments. These in-core user and resource tables are each reloadable at any time by the execution of their special started tasks.

Installation

Installation is performed through the standard SMP RECEIVE, APPLY and ACCEPT procedure. The process has been tested with SMP level 04.48 which is the level provided by the Turnkey 3 MVS system. This presumably is equivalent to having PTFs UR13349, UR15994, UR17644 and UR19590 applied. Tests have shown that APPLY processing doesn't work reliably at lower levels: For example at level 04.44 APPLY doesn't assemble and link all modules. So before starting please update your SMP as close to level 04.48 as possible.

1. Edit jobs A@PREP, C@APPLY and D@ACCPT:
    - change all occurrences of HLQ to the desired high level qualifier for the RAKF libraries, e.g. RAKF, RAKF.V1R2M0, etc. Please note that HLQs where HLQ.SAMPLIB, HLQ.ASAMPLIB, HLQ.APROCLIB, HLQ.APARMLIB HLQ.MACLIB, HLQ.AMACLIB, HLQ.SRCLIB or HLQ.ASRCLIB already exist are not recommended.
    - change all occurrences of dddddd to the desired volume for the RAKF distribution libraries.
    - change all occurrences of ssssss to the desired volume for the RAKF target libraries.
    - change all occurrences of tttt to the device type and all occurrences of rrrrrr to the volume of the sysres device where SYS1.LPALIB and SYS1.LINKLIB of the system reside on which RAKF is to be installed (If you are running TK3 MVS and install on your current system the device type is 3350 and the volume is MVSRES).
    - Verify that in jobs C@APPLY and D@ACCPT the sequence of the SYSLIB and the target and distribution library DD statements matches your SMPAPP and SMPACC procedures. The jobs have been tested

with the SMP procedures as defined in Volker's MVS TK3 system. If the sequence of DD statements in the SMP procedures in your system doesn't match the sequence of the overriding DD statements in C@APPLY and D@ACCPT RAKF elements may erroneously be placed in SYS1.ASAMPLIB, SYS1.SAMPLIB, SYS1.APROCLIB, SYS1.APARMLIB, SYS1.AMACLIB or SYS1.MACLIB instead of their HLQ.xxxxLIB counterparts.

2. Submit job A@PREP. This job performs the following actions to prepare your system for RAKF 1.2.0 installation:
   - Delete MODs and LMODs that are defined in MVS 3.8j as placeholders for RACF modules from SMP's target and distribution zones.
   - Delete the placeholder modules as well as potentially installed pre RAKF 1.2.0 modules from SYS1.LINKLIB and SYS1.LPALIB.
   - Allocate RAKF 1.2.0 target and distribution libraries.
3. Receive RAKF by submitting job B@RECV. Be prepared to mount the RAKF distribution tape rakf12.aws after submitting the job.
4. Apply RAKF by submitting job C@APPLY. This job places all RAKF components in their designated target libraries and assembles and link-edits the SVCs (ICHRIN00) and the SAF router (ICHSFR00) to SYS1.LPALIB and the support modules (ICHSEC00, RAKFUSER, RAKFPROF and RAKFPWUP) to SYS1.LINKLIB. These six load modules constitute the complete code provided by RAKF, making up an extremely small footprint as compared with RACF™.
5. Accept RAKF by submitting job D@ACCPT. This step can be skipped if you don't want to accept the RAKF sysmod now. It is, however, strongly recommended to accept it as only then it will be possible to "test drive" updates that might be published later and to return to the previous state in case of problems.
6. Perform an IPL CLPA of your system. The system will come up with all RAKF elements installed but without activating RAKF. Successful installation can be verified from the RAKF banner being displayed at the beginning of the IPL and by checking the system's SVC table for example using menu option T in IMON and scroll forward to the RACF SVCs (130 - 133) which should show entries like this:

   ```
   130 82 00EAEB30 3/4        RACHECK    -RACF IGC00130 RAKF  mm/dd/yy
   131 83 00EAEB52 3/4 YES    RACINIT    -RACF IGC0013A RAKF  mm/dd/yy
   132 84 00EAEB74 3/4 YES    RACLIST    -RACF ICHRIN00 DUMMY mm/dd/yy
   133 85 00EAEB9A 3/4 YES    RACDEF     -RACF IGC0013C RAKF  mm/dd/yy
   ```

   where mm/dd/yy is RAKF's installation date.
7. Check for any applicable RAKF PTFs and install these following the steps in chapter “PTF Installation” on page 6. PTFs are available in folder files/RAKF/PTFs of the H390-MVS Yahoo group and in CBT file 850.
8. Consult chapter “Customization” on page 7 for information on how to customize and activate RAKF.

PTF Installation

To install a RAKF PTF pretty much the same procedure as for RAKF installation is used:

- Upload the PTF to a sequential file or PDS member (LRECL=80,RECFM=FB) on your MVS system.
- Customize job B@RECV from your HLQ.SAMPLIB (HLQ = highlevel qualifier for RAKF libraries as chosen at RAKF installation time): Have the //SMPPTFIN DD statement point to the dataset or PDS member where you placed the PTF and change the RECEIVE command to read:

  RECEIVE S(PTF-ID) .

  PTF-ID is the ID of the PTF, for example RRKF001. Submit the job. Return code 0 is expected.
- Customize job C@APPLY as described in chapter “Installation” on page 4 (if not already done) and change the APPLY command to read:

  APPLY S(PTF-ID) DIS(WRITE) .

  Check the comment section at the beginning of the PTF (the “Cover Letter”) for any special installation instructions for APPLYing the PTF and perform these as appropriate. Then submit the APPLY job. Return code 0 is expected.
- Test RAKF operation with the PTF in place to ensure that no function you use got broken.
  - When RAKF with the applied PTF operates as it is supposed to accept the PTF: Customize job D@ACCEPT as described in chapter “Installation” on page 4 (if not already done) and change the ACCEPT command to read:

    ACCEPT S(PTF-ID) DIS(WRITE) .

    Check the Cover Letter for any special installation instructions for ACCEPTing the PTF and perform these as appropriate. Then submit the ACCEPT job. Return code 0 is expected.
  - When RAKF with the applied PTF exhibits problems, restore the PTF: Customize job D@ACCEPT as described in chapter “Installation” on page 4 (if not already done) and change the ACCEPT command to read:

    RESTORE S(PTF-ID) DIS(WRITE) .

    Submit the job. Return code 0 is expected. After running the RESTORE command your RAKF system is restored to the state before the PTF was applied.

Customization

This chapter describes the steps to customize and activate RAKF 1.2.0 after the basic SMP4 installation has been completed.

1. a) Allocate a PDS named SYS1.SECURE.CNTL (LRECL=80). This PDS will contain 2 members:

   - USERS: The RAKF users table
   - PROFILES: The RAKF profiles table

   If you've already allocated this library in an earlier version of RAKF you can continue to use it. The name of the library has to be SYS1.SECURE.CNTL. Although it could be changed this has to be done in several locations and will for the sake of simplicity not be outlined in these instructions.

1. b) Allocate a sequential dataset named SYS1.SECURE.PWUP with LRECL=18 and RECFM=F (fixed records of 18 bytes, no blocking). This dataset serves as a queue: User initiated password changes are saved here until the next run of RAKFUSER integrates them into the RAKF users table.

   If you've already allocated this dataset in an earlier version of RAKF you can continue to use it. The name of the dataset has to be SYS1.SECURE.PWUP. Although it could be changed this has to be done in several locations and will for the sake of simplicity not be outlined in these instructions.

Note: During the process of adapting the profiles and users tables to meet your security requirements (step 6) special attention to the protection of the datasets created in step 1 should be paid:

- SYS1.SECURE.CNTL: started tasks and the user(s) and/or group(s) responsible for RAKF administration need UPDATE access to this dataset.
- SYS1.SECURE.PWUP: started tasks need UPDATE access to this dataset.

As both datasets contain clear text passwords normal users shouldn't be allowed any access to them. An easy way to protect these datasets is to define a dataset profile SYS1.SECURE.* with universal access NONE and selectively allow the RAKF administrator user(s)/group(s) UPDATE access to this profile. If the standard setup is used started tasks have operations authority and thus don't need to be explicitly allowed.

2. If you're a first time RAKF user copy the following members from HLQ.SAMPLIB to SYS1.SECURE.CNTL:

   - MINUSR using a new name of USERS
   - MINPRF using a new name of PROFILES

This establishes a minimal configuration equivalent to an unprotected system with IBMUSER being defined as the only user. IBMUSER has operations privilege allowing all accesses.

If you are running Volker's TK3 (Turnkey MVS) system you can use TK3USR instead of MINUSR which defines besides IBMUSER also the users HERC01, HERC02, HERC03 and HERC04 with equivalent attributes as they are defined in UADS. Note, however, that RAKF doesn't allow users having no password. For this reason IBMUSER, HERC01 and HERC03 have a password of NONE defined instead of no password as in TK3.

Note that each TSO user still needs an UADS entry to define the TSO attributes and authorizations. These are not covered by RAKF's users table.

3. Issue the command

   ```
   S RAKF
   ```

   at the MVS console and reply YES to message RAKF002A. Verify that the profiles and users tables initialize correctly and that you can logon using the usernames and passwords from SYS1.SECURE.CNTL(USERS).

   Note: Although the RAKF procedure loads the in-core users and profiles tables it is not primarily intended to be used for that purpose during normal operations. To routinely update the profiles or user table to activate changes the procedures RAKFPROF or RAKFUSER, respectively, should be used. The main purpose of the RAKF procedure is to provide a means to activate RAKF if initialization didn't take place automatically at IPL time, for example during installation and customization. If the RAKF procedure is run when RAKF had been activated already it will skip the initialization and refresh the in-core users and profile table, which is equivalent to running RAKFPROF and RAKFUSER.

4. To enable automatic initialization of RAKF at system IPL time (highly recommended!) MSTRJCL needs to be modified to contain DD statements pointing to the users and profiles tables and the password changes queue:

   ```
   //RAKFPROF DD DSN=SYS1.SECURE.CNTL(PROFILES),
   //            DISP=SHR
   //RAKFUSER DD DSN=SYS1.SECURE.CNTL(USERS),
   //            DISP=SHR
   //RAKFPWUP DD DSN=SYS1.SECURE.PWUP,
   //            DISP=SHR
   ```

   Member ZJW0003 of HLQ.SAMPLIB provides a sample USERMOD accomplishing this. If ZJW0003 fits your system just submit it. If not use any other method of your choice to add the required DD cards.

5. Edit SYS1.PARMLIB(RAKFINIT) and change the value NO in line 1 to ASK. Then reIPL the system. Message RAKF002A will now be issued immediately after master scheduler initialization and give you the choice to start RAKF or not. Reply YES to initialize RAKF and continue the IPL.
6. Start now modifying your USERS and PROFILES tables until the desired level of protection is reached. Please read chapter “Setting Up and Enabling the

Users and Profiles Tables" on page 9 for information on setting up these tables.

7. Once you're satisfied with your configuration change line 1 in SYS1.PARMLIB(RAKFINIT) from ASK to YES which will cause RAKF to be activated unconditionally during IPL.

Setting Up and Enabling the Users and Profiles Tables

Please note that when MVS is informed that "security" is present on the system, all previously defined passwords, including VSAM passwords, are ignored. Any password protection must be reinstated by the security system. Evidently, the MVS designers wanted any passwords designated by security, not to be interfered with by any other password mechanisms in MVS.

Additionally it should be noted that the ESG Security System, which RAKF is based on, has been developed in 1991 for MVS versions released 5 to 10 years later than RAKF's target MVS 3.8j. Consequently RAKF supports security features that are used by MVS 3.8j differently than by later MVS versions or are not used at all and thus will not work as expected. Please read chapter "Security Authorization Facility (SAF) Support in MVS 3.8j" on page 12 for considerations concerning this situation.

The way that protection will work on your system is entirely up to your control. Protection depends completely on the way you code the users and profiles tables. RAKF utilizes these tables for its protection decisions, instead of using a RACF™ database. Minimal tables to create a state equivalent to MVS without an active security product have been provided. Starting from these you should gradually move forward to tighten security as needed on your system.

In order for both the users and profiles tables to be valid, they must be in sort order. Under RPF Edit, the command "SORT" is adequate to sort the records correctly (if they were coded properly in the first place). Sort errors will inhibit initialization of the tables and will generate nasty error messages.

Note that the tables may contain comment lines starting with an asterisk (*). If you have comment lines in your tables using a "SORT" command in RPF Edit will move these lines away from their desired location. Consequently it is advisable to keep your users and profiles tables in sort order manually if comment lines are used.

In the profiles table, DEFAULT, or "UNIVERSAL" access for any facility or dataset must be coded before any specific access is coded. You code a universal access entry in the profiles table by leaving the user group field blank. Then you code other entries for the same facility, specifying different settings for each user group that will have special access (or denial of access) to that facility.

Generally, only those features included in "IBM RACF™ Version 1.7" have been emulated.

RAKF uses the decision logic of the ESG Security System without modifications. The author of the ESG security system has referred to two RACF™ manuals during his

planning. It is important for all users of RAKF to obtain these manuals also. These are needed in the security administration, which will be ongoing.

The two manuals are:

SPL RACF (SC28-1343-2)	-	Referred to for information how to write macros and return codes.
RACF Administrator's Guide (SC28-1340)	-	Has an overview of profiles that should be used, and their structure.

You should try to find versions of these manuals being as close as possible to RACF™ Version 1.7 because later manuals describe features not available with RAKF and/or MVS 3.8j.

An example for a minimal users table may be found in the member MINUSR in HLQ.SAMPLIB. An example for a minimal profiles table may be found in the member MINPRF. Users of Volker's Turnkey 3 system can find in member TK3USR a users table defining all users present in the system after initial TK3 installation to RAKF. Note, however, that these tables only define users and passwords. All profiles are set to grant every request, so the system is in a state equivalent to running without an active security product after activating these minimal tables.

Members CJYUDATA and CJYPDATA of the original ESG Security System distribution as available in CBT file 165 are sample tables showing a realistic setup and might be used as an example for defining the RAKF tables for MVS 3.8j. It is hoped that enough data is provided in these examples to give a working knowledge for further coding and setting up but it should be noted that these tables use resource classes available only in MVS versions 5 to 10 years later than MVS 3.8j and thus cannot be used as they are. Major differences in security handling between MVS 3.8j and later systems are detailed in chapter "Security Authorization Facility (SAF) Support in MVS 3.8j" on page 12.

Please be informed that in core, these tables will be read from the bottom upwards which is the reason why the source tables need to be in ascending sort order: Reading them in reverse order ensures to find the most significant hit for a resource or user search before any less significant hits, so tables search can (and will!) always be stopped upon the first hit.

The users table is coded in SYS1.SECURE.CNTL(USERS) as follows:

Columns	Contents
1 - 8	USERID (TSO, CICS, or whatever application)
10 - 17	User Group (Installation defined)
18	Asterisk '*' to denote that multiple user groups exist for this userid. Otherwise blank.
19 - 26	Password
28	Operations Authority (Y or N). If "Y", then access is always granted to this user unless it is denied explicitly.
31 - 50	Comment field (ignored by ESG Security but used by "IBM RACF™").
51 - 80	Ignored

Userids in the users table that were set up with multiple group entries will get the highest authority for all protected objects in all the groups. As a practical example, multiple groups are used for managers who oversee the work of several programming groups. The multiple group arrangement gives these managers access to everything done by all the groups under them.

The users table is activated by issuing the command

```
S RAKFUSER
```

at the MVS console.

The profiles table is coded in SYS1.SECURE.CNTL(PROFILES) as follows:

Columns	Contents
1 - 8	Facility title: (DASDVOL, DATASET, FACILITY, etc.) See the RACF™ Administrator's Guide. You need to know the different facility types used by the operating system, CICS, TP products, and vendor products.
9 - 52	Dataset Name, or Generic Name, or Name to be protected. (Generics are achieved using the asterisk '*'. See the examples.)
53 - 60	User Group Id (Installation defined). Blanks in this field denote universal access rules for this resource.
61 - 66	Permission Level (NONE, READ, UPDATE, ALTER)
67 - 72	Blank

To protect products other than MVS 3.8j, they must have an interface to the security system, typically through the use of specific profiles in the FACILITY class.

The profiles table is activated by issuing the command

```
S RAKFPROF
```

at the MVS console.

It has to be reemphasized that your security protection is completely dependent on how you code these tables. Please get most of your knowledge from the RACF™ Administrator's Guide.

Special attention should be given to the fact that RAKF allows ALTER access to all undefined resources. A user can, for example, delete any dataset on any volume even if access NONE is defined for the dataset as long as there is no DASDVOL profile defined: ALTER access to a DASDVOL allows scratching of any file on it regardless of the file's protection and exactly that's what a user gets if no DASDVOL profile is defined.

Batch Jobs and Started Task Considerations

Batch jobs and STCs are controlled as follows: All batch jobs default to a userid of PROD and a user group of PRDGROUP. Started tasks are forced a userid of STC and a user group of STCGROUP. This default is imposed by ICHSFR00. Any job that has no userid connected to it is assigned a userid of PROD and a user group of PRDGROUP by ICHSFR00. That situation is true for jobs submitted by RJE or by a local (card or internal) reader. The authorities (i.e. Operations or not) of the PROD and STC userids are hardcoded in ICHSFR00. Upon initial RAKF installation, the PROD user has no Operations authority while the STC user has Operations authority defined. The specific authorities of the PRDGROUP and STCGROUP groups are controlled by the profiles table.

Under MVS 3.8j no userid propagation takes place. Without further measure all jobs entering the system have no userid and thus get userid PROD and group PRDGROUP assigned by RAKF, be it jobs submitted by already authenticated users or jobs, or by started tasks. That means that all jobs that shall run under a specific userid need to have the USER and PASSWORD parameters coded on their JOB card. Coding these parameters constitutes a security hole if jobs are saved in publicly readable JCL libraries. But even with read protected JCL libraries having to code these parameters is for sure not desired.

To avoid having to code USER and PASSWORD parameters on JOB cards it is highly recommended to install an IKFEFF10 user exit that automatically provides these parameters for jobs submitted using the TSO submit command processor and to configure RPF to use TSO submit instead of RPF submit. That way the majority of jobs submitted in day to day system usage don’t need to code these parameters on their JOB cards. There are many such exits available on the CBT tape and it might be a problem to find one that fulfills that (and only that) function and works on MVS 3.8j. File 358 from the “Georgia Departement of Labor” on the old CBT249 archive does exactly that.

Security Authorization Facility (SAF) Support in MVS 3.8j

RAKF supports security features that are used by MVS 3.8j differently than by later MVS versions or are not used at all and thus will not work as expected. The “historical” reason for this is that RAKF's predecessor ESG Security System is dated 1991

which is 5 to 10 years later than most components of "current" MVS 3.8j systems. Some of the issues resulting from this discrepancy are discussed here.

General Resource Classes

There are resource classes supported by RAKF but not used by MVS 3.8j. So, if you are going to define a profile in any resource class in the RAKF profiles table that you never used before make sure to test it by trying an access that should be denied by that profile. Only if it then really gets denied, you can be sure that the corresponding hook to call the SAF is already implemented in the level of MVS you are running.

The probably most relevant of these classes is the PROGRAM class which is meant to be used to protect programs from being executed by unauthorized users: RAKF will accept profiles in the PROGRAM class flawlessly but none of the programs defined there will be protected because MVS 3.8j simply doesn't ask SAF for permission before executing a program.

So, basically, with MVS 3.8j one has to make sure that “critical” programs cannot be executed by unauthorized users through other means. The following solutions might be feasible on a case by case basis:

- Programs with source code available: Introduce a profile in the FACILITY class and have the program verify the caller's authorization against this profile.

  An example for this method can be found within RAKF itself (introduced through PTF RRKF003): The utilities RAKFUSER and RAKFPROF used to update the in-core USERS and PROFILES tables are critical in the sense that anyone having access to them can take over the security administration of the whole system and thus can conduct arbitrary fraudulent activities. To enable control over who is authorized to use these utilities they request READ access to FACILITY RAKFADM and don’t execute if this access isn’t granted. That way the use of these utilities can effectively be restricted by defining profile RAKFADM in the FACILITY class and giving only authorized users or groups READ access to this profile.
- Programs without source code available: Create a separate loadlib protected by a DATASET profile and place the program there. This, of course, is kind of a last resort.

Dataset Protection

The implementation of calls to the security system in MVS 3.8j to protect datasets greatly relies on the concept of indication: Only datasets having their "RACF indicator" set on are reliably protected. The RACF indicator is a bit in the type 1 DSCB of a non-VSAM dataset or in the catalog entry of a VSAM object. Once activated RAKF ensures that all newly created datasets, catalogs and VSAM objects have their RACF indicator turned on. But this is typically not the case for already existing ones. So it is strongly recommended to manually turn the RACF indicator on for all datasets, catalogs and VSAM objects that already existed before RAKF activation. It should also be noted that once the whole system is RAKF protected (i.e. the RACF indicator is on for

all datasets, catalogs and VSAM objects) it is no longer feasible to run it without RAKF being active as most accesses will then be denied.

First time users of RAKF might have a problem instantly RACF indicating all datasets in the system. Also there might be users needing to protect only non-VSAM datasets (i.e. don't need catalog and/or VSAM object protection). For those users a set of ZAPs to the MVS modules handling non-VSAM datasets is provided that enforce calls to RAKF for all non-VSAM datasets, not only for those with RACF indicator turned on. So the easiest way to get some basic protection on the system is to simply install RAKF together with the MVS ZAPs and you're all set.

It should be noted, however, that this constitutes a massive change of logic flow in the modules ZAPed. There exist complex interactions between VSAM catalog management and the DADSM functions modified by the ZAPs. So, while intensive testing has shown that these ZAPs together with correctly defined RAKF DATASET and DASDVOL profiles provide reliable protection for non-VSAM datasets, they partly break VSAM catalog management. An artifact of this situation is that with applied ZAPs an IDCAMS "DELETE CLUSTER" command of a VSAM cluster that hadn't been allocated with the SUBAL option ends with CC=0 but leaves the data and index dataspaces of this cluster orphaned on disk.

Phil Roberts has found a workaround to delete these orphaned VSAM components from disk which is cited here to help in case this problem occurred. He wrote:

```
"I have tinkered some and have a work around for the situation where a
VSAM dataset may get orphaned due to a mixed environment. It doesn't
take any outside utilities.

Basically if a VSAM cluster is deleted while running with the MVSZAPs
(not recommended operation but perhaps necessary for some) one can:

- DELETE hlq.vsam.name NOSCR
- CDSCB hlq.vsam.name.data DSORG(PS) VOL(xxxxxx) RACF
- CDSCB hlq.vsam.name.index DSORG(PS) VOL(xxxxxx) RACF
- RPF 3.4 with hlq and VOL xxxxxx to C CATALOG then D to delete the
  orphaned components from pack xxxxxx"
```

This example illustrates why it is strongly recommended to use the MVS ZAPs only temporary, during a phase while working to RACF indicate the whole system, and to remove them as soon as possible.

The following table gives an overview on the influence of the ZAPs in a few scenarios and should help to decide whether to use them or not:

		All datasets VSAM objects and catalogs are indicated		System has mixed indication status			
				VSAM		NVSAM	
		VSAM	NVSAM	indicated	not	indicated	not
Z	Reliable protection	Y	Y	Y	N	Y	Y
A	Reliable catalog management	N	Y	N	N	Y	Y
P	Catalog protection	Y	Y	Y	N	Y	N
Z	Reliable protection	Y	Y	Y	N	N	N
No A	Reliable catalog management	Y	Y	Y	Y	Y	Y
P	Catalog protection	Y	Y	Y	N	Y	N

It should be noted that in a mixed environment without ZAPs even indicated non-VSAM datasets aren't reliably protected, that's not a typo! The table shows clearly that one should by all means try to reach the "not ZAPed and fully indicated" configuration because this is the only one to provide full protection without introducing risky changes to basic system functionality.

The following steps describe the installation and removal of the MVS ZAPs:

1. Run job LPABACK: In step 2 you'll ZAP MVS modules IFG0194A, IGC0002I, IGC00030 and IGG0553A. These modules are typically located in SYS1.LPALIB but they might also be elsewhere in LPALIST or LINKLIST. Job LPABACK copies these modules and their aliases to a backup library... just in case you want to revert to the non-ZAPed versions later. Find out where these modules reside on your system and change SYS1.LPALIB in job LPABACK to the name of that library. If you want to use another name for the backup library than RAKF.LPALIB.BACKUP then change that name too at the location indicated by a comment in the JCL. Then submit the job and check that the library has been created correctly. It should contain 4 members and 17 aliases.
2. Run job ZAPMVS38: This job applies the MVS ZAPs. If necessary change the //SYSLIB DD statement to the name of the library where IFG0194A, IGC0002I, IGC00030 and IGG0553A reside. If module IFG0194A had already been ZAPed as recommended in Phil D's original rakf.pdf installation procedure then comment out the 6 ZAP statements indicated by the respective comment in the job. Run the job and verify that the ZAPs where successful. If any of the ZAPs failed your system isn't at a service level compatible with the ZAPs. The ZAPs were tested with the following PTF levels of the ZAPed modules:
    - IFG0194A PTF UZ74083
    - IGC0002I PTF UZ68267
    - IGC00030 PTF UZ63439
    - IGG0553A PTF UZ63439

This is the level of Volker's Turnkey 3 system. If you're using other PTF levels most probably the ZAPs need to be reworked to reflect at least the correct addresses of the ZAPed instructions. After having run job ZAPMVS38 successfully your system is ready to use with RAKF 1.2.0 for basic non-VSAM dataset protection.

3. Restore job LPAREST: If you want to remove the MVS ZAPs... job LPAREST copies the modules backed up in step 1 by job LPABACK back to their original locations. If you adjusted dataset names in LPABACK make analogous changes in LPAREST before running it. Please note that Phil D's original OPEN processing ZAP is still applied after running LPAREST if it was already applied before running LPABACK. So, if you want to get rid of that ZAP too which is strongly recommended, you need to find another source to restore IFG0194A and its aliases from (hopefully you backed them up before applying Phil D's ZAP).

Deinstallation

This document describes the steps to deinstall RAKF 1.2.0 and reinstate native MVS security behavior.

```
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
Danger!!! Danger!!! Danger!!! Danger!!! Danger!!! Danger!!!
\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
```

DEINSTALLING RAKF IS NOT RECOMMENDED!

After deinstallation RACF indicated files will no longer be accessible. This may lead to a not IPLable or not accessible system. Removing RAKF and reinstating native MVS security behavior is not recommended unless in preparation the RACF indicator of all non VSAM datasets, VSAM catalogs and VSAM objects has been turned off.

The recommended way instead of deinstalling RAKF is to let it be installed and active and replace the profiles table by the minimum table MINPRF from HLQ.SAMPLIB. This enables ALTER access by everyone to everything and thus is access wise equivalent to a vanilla MVS system without RAKF.

DEINSTALLING RAKF IS NOT RECOMMENDED!

```
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
Danger!!! Danger!!! Danger!!! Danger!!! Danger!!! Danger!!!
\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
```

The following instructions assume that you've read the above recommendation and understood the consequences of removing RAKF from your system. Continue at your own risk:

1. If you've applied the MVS ZAPs using job ZAPMVS38 consult chapter “Dataset Protection” on page 13 on how to remove these ZAPs. The ZAPs MUST be

removed prior to deinstalling RAKF otherwise the system will not be accessible or will not even IPL after the deinstallation.

2. Copy members C@APPLY, RAKFRMV and RAKF2MVS from your RAKF SAMPLIB to a private library. Work with these copies throughout the following steps as the RAKF libraries will be deleted! The jobs to be edited and submitted in the course of this procedure have been prepared under the assumption that the libraries to be deleted or modified can be located through the standard catalog search order. If this is not the case look thoroughly through the JCL to be sure to add VOL and/or UNIT parameters to the DD statements to correctly identify the libraries.
3. a) If RAKF SYSMOD TRKF120 has been APPLIed but not ACCEPTed, edit installation job C@APPLY:
   - change all occurances of HLQ to the high level qualifier of your RAKF libraries, e.g. RAKF, RAKF.V1R2M0, etc. and verify the correct sequence of the overiding DD statements against your SMPAPP procedure as described in chapter “Installation” on page 4.
   - change the command "APPLY S(TRKF120) DIS(WRITE)" in line 30 to read "RESTORE S(TRKF120) DIS(WRITE)".

   Submit changed job C@APPLY to remove all RAKF components from your system.

   b) If RAKF SYSMOD TRKF120 has been ACCEPTed, edit job RAKFRMV:

   Change all occurrences of tttt to the device type and all occurrences of rrrrrr to the volume of the sysres device where SYS1.LPALIB and SYS1.LINKLIB of the system reside from which RAKF is to be deleted (if you are running TK3 MVS and delete from your current system the device type is 3350 and the volume is MVSRES).

   Submit changed job RAKFRMV to remove all RAKF components from your system.
4. Review the DD statements pointing to SYS1.LINKLIB and SYS1.LPALIB in job RAKF2MVS and add VOL and/or UNIT parameters if necessary. Change all occurances of HLQ to the high level qualifier of your RAKF libraries. Submit the job which concludes the deinstallation procedure.

The system libraries as well as SMP are now reverted back to native MVS security. Before activating this configuration through an IPL CLPA make sure that no dataset needed for IPL is RACF indicated.

Version History

ESG Security System (March 1991)

The ESG Security System was published by Craig J. Yasuna as an alternative to IBM's RACF™ and similar products. It uses the ICHRTX00 security router exit to communicate its security decisions to MVS. ICHRTX00 is a user exit of the SAF router ICHSFR00.

RAKF "RAcK oF" Security System (August 2005)

RAKF was published by A. Philip Dickinson as an adaption of the ESG Security System to MVS 3.8j. The ESG Security doesn't natively support MVS 3.8j for several reasons. The major ones are:

- At MVS 3.8j times the SAF router ICHSFR00 was part of the RACF™ product and thus not available on MVS systems without RACF™ being installed. So the ICHRTX00 exit as a convenient place for third party security products to hook into the SAF didn't exist.
- incompatible ACEE handling.
- The ESG Security System uses the BAS instruction which is not available in S/370.
- ESG Security's 24/31 bit AMODE handling wasn't compatible with the usage of high order address bytes for flags in MVS 3.8j.
- incompatible parameter lists of RACROUTE, RACDEF, RACHECK and RACINIT.
- handling of in core profiles incompatible with MVS's VSAM catalog management.

Phil D. converted ESG Security's ICHRTX00 router exit into an ICHSFR00 SAF router to overcome the first major incompatibility. He also solved the next two points and in parts the 24/31 bit AMODE issues. The rest remained undetected when he published his work and caused several problems when using RAKF on MVS 3.8j:

- Arbitrary 0C4 abends during RACINIT and RACDEF processing. The most severe of these abends is the "initiator blowout" mentioned in http://tech.groups.yahoo.com/group/H390-MVS/message/10015.
- Arbitrary 0C4 abends and FREEMAIN errors during RACHECK processing of VSAM catalogs and objects. These problems are discussed in http://tech.groups.yahoo.com/group/H390-MVS/message/11811.
- Arbitrary invalid authorization decisions: Access denied if it should have been granted and access granted if it should have been denied. The latter in a way that with some trial and error any user is able to acquire ALTER access to any dataset in the system.

RAKF 1.2.0 (April 2011)

When the above mentioned problems became visible and identified as being RAKF caused in several MVS 3.8j environments it turned out that Phil D's RAKF source wasn't accessible any more (lost in a package crate from moving). Phil Roberts stepped in and reconstructed the source by disassembling Phil D's binary RAKF distribution and comparing it with the original source of the ESG Security System. Based on that reconstructed source the rest of the incompatibilities listed above were identified and corrected by Jürgen Winkelmann.

To avoid another loss of the source to occur an additional effort has been made to clean up the source to a publishable state and to provide an SMP4 compatible source distribution enabling RAKF installation using the standard SMP 4 standard RECEIVE, APPLY, ACCEPT procedure.

RAKF 1.2.0 PTF Summary

The following PTFs are available as of the publishing date of this manual:

RRKF001: Enable comment lines (lines starting with *) to be entered in the source users and profiles tables and minor bug fixes in RAKFUSER utility.

RRKF002: Enhance RACINIT NEWPASS functionality to support permanent password changes to be initiated by an application. This enables standard password change functionality as for example entering `currentpw/newpw` on the “ENTER CURRENT PASSWORD for uuuuuuu” at TSO logon time.

RRKF003: Security enhancement in users and profiles tables processing.

RRKF004: Consolidation of documentation members from RAKF’s SAMPLIB to this manual.

These PTFs are available in folder files/RAKF/PTFs of the H390-MVS Yahoo group and in CBT file 850, future PTFs will be made available at the same locations.

Appendix A: Copyright Information

Please observe the ESG Security System copyright whenever using this product:

```
*
*********************************************************************
*                                                                   *
*    COPYRIGHT (C) 1991 BY CRAIG J. YASUNA.  ALL RIGHTS RESERVED.   *
*                                                                   *
*    THIS SOFTWARE PRODUCT OR ANY OF ITS COMPONENTS MUST NOT BE     *
*    SOLD, GIVEN, OR OTHERWISE DISTRIBUTED TO ANY OTHER COMPANY     *
*    WITHOUT THE PRIOR WRITTEN PERMISSION OF:                       *
*                                                                   *
*                                  CRAIG J. YASUNA, PRESIDENT       *
*                                  ENTERPRISE SYSTEMS GROUP         *
*                                  2 MARC COURT                     *
*                                  EDISON, NEW JERSEY 08820         *
*                                                                   *
*    THIS PRODUCT IS NOT "PUBLIC-DOMAIN", BUT ITS AUTHOR HAS GIVEN  *
*    PERMISSION THAT IT BE DISTRIBUTED ON THE CBT MVS MODS TAPE.    *
*                                                                   *
*********************************************************************
*
* From: Yasuna, Craig
* Sent: Thursday, April 07, 2011 6:47 PM
* To: Winkelmann Juergen
* Subject: RE: Question concerning the ESG Security System
*
* Absolutely ... I am very glad that the code still lives on and that it
* has value.
*
* THANKS!!! - Craig
*
* -----Original Message-----
* From: Winkelmann Juergen
* Sent: Thursday, April 07, 2011 12:46 PM
* To: Yasuna, Craig
* Subject: Question concerning the ESG Security System
*
* . . .
*
* Phil Roberts reconstructed Phil Dickinson's changes to the original
* ESG source through disassembly and compare. Based on that source I
* finalized Phil D's work and now have RAKF fully working on MVS 3.8j.
* I'm still in a final testing phase. After having finished this, I'd
* like to post RAKF to the H390-MVS group and also to submit the changed
* source back to CBT to avoid it getting lost again. Before doing this,
* I'd like to ask for your consent as the original author. As far as I
* understood Phil D asked for the same permission in 2005 but I don't
* want to just quietly take over from him.
*
```