Security for the corporate platform

Before you begin

Before installing SecureEntry in a given machine, you should read carefully this on-line manual, and be sure that all prerequisites are met. Notice that this is a quite impressive piece of software, but because of its nature, it is very important that you know what you are doing when using it.

Also look for a file named *README.DOC* in the first installation diskette, for last minute changes and detailed driver changes description.

Copyright

What is SecureEntry

SecureEntry is a software product that enhances any OS/2 workstation, in standalone or network configuration by providing it with an extensive set of functions in the area of security and personalization.

With SecureEntry you can define the users and groups of users that are allowed to use the workstations, and at the same time, set up their working environment in such a way that they will have the machines already configured to do the work they are intended to, by also restricting the features and functions they should not be able to access.

By doing all of the above, SecureEntry does effectively convert a single user machine into a workgroup machine, which is effectively then a serially shared multiuser machine. From the end user's perspective, all what changes from his normal way of working is that an identification will be required before starting a work session, being at

that point when SecureEntry will establish what security profiles to activate, thus setting up the environment for the user working in the workstation.

SecureEntry has been designed in such a way that security components can easily be 'plugged in' so that if a given software or hardware piece is to be secure-controlled, a new security component can be written and incorporated easily to the architecture. As a sample, you can easily write and define to SecureEntry an interface to your Lotus Notes user's profiles so that each user logging into any workstation will have his personal profile activated for him. The following security components are included in the base product :

Desktop restrictions component : Allows to restrict the properties of the objects existing in the OS/2 Workplace Shell desktop in a per user/group basis. For instance, you could define some objects to be visible for some users and invisible for others.

The Personal desktop component : Allows the definition of 'personal folders', as well as their contents, so that those folders will 'travel' with the user no matter in which machine does the user log on (supposing SecureEntry is installed in a networked environment). This folders are also known as memory and/or user folders.

The Launchpad component : Allows you to define a given Workplace Shell Launchpad for each user or group of users, so that each one has always the launchpad that better suits their working needs.

The Window list component : This component's purpose is to personalize the switch list behavior, with a wide range of options, that can vary from the inhibited state to a full-function one. You can, among other things, select which entries are allowed to appear in the switch list, and whether you permit it to be used only as a task switcher, or do specific manipulations with the appearing programs. You can also change the aspect of the switch list.

Windows behavior component : Also called restricted window frames, allows you to set up the initial position/size of any application frame window, as well as inhibit any of their system menu entries. For instance, you may be interested in not letting users to 'play' with a given application fonts, by deactivating that application's 'Fonts' menu entry.

SES behavior component : This component allows for customizing general operating parameters for the system, such as the desired Ctl-Alt-Del behavior, systemwide hot keys, or screen saver bitmaps and activation settings, in a per user or group basis.

Floppy restrictions component : This component works at a device driver level in order to avoid or allow floppy drive access depending on the customized profile for the user logged on. It allows also access in encrypted mode.

Treelock component : Working at a file system level, this component allows the administrator to define which part(s) of the existing file system is the user allowed to read, look at, write or execute, in a per process basis. For instance, you could define a given user to only be able to write to certain directories when working with a given word processor program. This component is also referred to as 'Tree_lock'.

The shortcuts component : This component allows for definition of shortcut keyboard/mouse combinations that allow for opening of the desired objects when issued.

The Hooked objects component : This component allows for the configuration of hooks to certain objects, so that a user chosen program is launched at object open or close time. This allows for easily setting up protected folders through one time password mechanisms. One implementation of such a tool is also supplied as part of this component.

The WarpCenter component : By using this component, you can define personalized warpcenters that match your security criteria, and then assign the resulting profiles to different users or user groups.

The Public Applications component : This component handles the administration of LAN Server Public Applications, allowing an administrator to assign a set of LAN Server Public Applications to users and groups of users.

The Processes Auditor component : This component allows you to measure the CPU utilization time by different processes on a user basis.

Other than the above mentioned components, which are programs that in the end you can profile in a per user/group basis, SecureEntry would not be complete without another set of programs which allow you to fully 'lock' the workstation from a security point of view. These are :

Startup process : Substitutes the OS/2 'startup.cmd' default processing by enabling the processing of a command file which can not be interrupted by the user, thus making the complete boot process a secure one, and at the time allowing for startup messages and progress indicators to be shown.

Session control programs : These are really the 'heart' of a SecureEntry system, allowing for users identification and control, plus functions such as logoff, lockup, shutdown,... in a secure way.

Administration subsystem : Allows you to easily define and assign the security profiles for the different components to your users and groups of users, through a common and integrated view, which is independent from the SecureEntry installed environment. This subsystem includes graphical interactive and also batch tools to do the administration.

File integrity check tools : Which you can use to grant critical files integrity at certain intervals.

Other tools : Such as the VDM utilities or the selective application closer, can be used to end up the integration of the SecureEntry software with your own production software.

Programming APIs : SecureEntry provides also with a set of comprehensive programming interfaces and user exits which can be used to Taylor its behavior to the required customer's needs.

Combine these with the broad scope of environments supported, and the provided level of user customization and code exits that allow for perfect customer control, and you will envision the new meaning that SecureEntry gives to the concept of 'Workstation security'.

SecureEntry is a very complex package, and for its nature, you are encouraged to read this document carefully before attempting to install and run this package.

IMPORTANT : Delivery policy, beta components and packaging

Delivery policy

It is important to notice that SecureEntry, as a package, is not being developed following the traditional versioning scheme of other products. Due to its modular nature, we are allowed to follow a more flexible and customer oriented policy. That is :

We are not providing with new or repackaged versions of the product in a timely basis. Instead, we distribute quite frequently new builds which are a complete replacement for the old ones. A new build can be used to install a new machine or service a previous SecureEntry installed one. This new builds incorporate :

Fixes to existing reported problems

New functions

For the first ones, i.e fixes, you are recommended to install the new build only if you have faced the reported fixed problems, which are described within the README.DOC file supplied with that build level, unless otherwise stated.

For the second ones, i.e new functions, these will be installed with the new build, but will be in 'beta' i.e not fully granted state for a while. The functions that are still in beta will be also properly documented within the above mentioned readme file.

As a last fact, note that we will only accept problem reports for production machines on functions that are not in beta. Likewise, you are only allowed to use these new functions appearing after your SecureEntry contract, as long as you have contracted SecureEntry evolutive maintenance.

Packaging : Evaluation vs. Production copies

SecureEntry has two different licensed package types :

An evaluation copy is full functional but only valid for a granted amount of days, being it normally of 180 days after build creation. After this period, you will start receiving messages about the fact that either the product has to be deinstalled, or properly licensed. Note that you are not allowed to run an evaluation copy in a production environment.

A production copy does not expire.

You can service a machine with an evaluation copy using a production copy, and the machine will automatically become licensed. Similarly, you can service a production copy with an evaluation one, and the machine will remain a production copy enabled one.

You can verify the type of package to install by looking into the ASCII file *SENTRY.SIG*, located in all of the distribution diskettes. After installing, this file is placed in the SecureEntry path, INSTALL directory.

UCM code

Note that the distribution policy for UCM code is slightly different :

There is no evaluation version of the UCM-MVS/ESA code. This code is only supplied to contracted customers. If you want to checkout the functionality provided by UCM, please proceed to install the UCM-OS/2 code, which is supplied with SecureEntry/2.

For the UCM-OS/2 code, the code is supplied completely with SecureEntry/2 drivers, either evaluation or production ones. However, the UCM-OS/2 code as provided is locked and can be used only for evaluation purposes for a period no longer than 180 days after package creation date. Once evaluated, and if you choose to contract the solution, you will be able to unlock the code as explained in the UCM administrator guide.

Hardware requirements

The following is necessary to run this version of SecureEntry :

1. A machine with 12 MB of memory (or more) and a 486-level processor (minimum)
2. Disk space free of at least 11.5 Mb (2 Mb additional required if you plan on installing the SecureEntry tutorial)
3. LAN hardware for LAN installations.

Note that when using WorkSpace On-Demand, you may also use Intel(c) architecture Network Station client machines such as the IBM Network Station series 2800 thin client, which has been fully tested as a RIPLable client, SecureEntry enabled of a WorkSpace On-Demand server.

Software requirements

The following (or upper level) software must be installed in the machines were you intend to install SecureEntry 3.0 on :

1. OS/2 WARP with Fixpack 17 or superior. The minimum screen resolution supported for administration tasks is of 800 x 600.
2. OS/2 SES (Security Enabling Services). This is only mandatory if you choose to use it at installation time instead of the SecureEntry provided SES emulator.
3. IBM LAN Server/Requester 4.0 Only for Lan Server type SecureEntry installs. The Server must be installed in the machine intended for SecureEntry Server function.
4. WorkSpace On-Demand, only if you wish to use SecureEntry WorkSpace On-Demand support. For WorkSpace On-Demand 1.0, the OS/2 Warp 4 code that execute the WorkSpace On-Demand requesters must be at FixPak 8 level minimum (FixPack 9 is recommended however to avoid memory leak problems). For WorkSpace On-Demand 2.0, then you won't need any additional FixPak for the OS/2 Warp 4 code that the WorkSpace On-Demand requesters execute.
5. If using the Processes Auditor Component, you are strongly encouraged to have Fixpak 40 for Warp 3.0, or Fixpak 10 for Warp 4.0.
6. If installing the UCM feature, then you also need :
    1. DDCS/2 single user in the UCM administration workstations
    2. The following Host computer software :
        1. MVS/ESA
        2. DB2
        3. RACF V1.9.2 or later (if using RACF validation).

If you had a previous version of SecureEntry installed.

If you had SecureEntry 2.0 or 1.0 installed:

You must deinstall it before upgrading. Use the appropriate tool supplied with your current version of SecureEntry to do so. Note that desktop restriction profiles you may already have in your installation are fully compatible with SecureEntry 3.0, as well as Launchpads. SecureEntry 3.0 can migrate those together with your users definitions to the new database format. To do so, you must run the 'MIGRADB.CMD' program once in the SecureEntry Server machine after reinstalling the new version. Note that this tool expects the environment variable 'SGM_LS' to be set and pointing to the directory where the 'old' SecureEntry users information was stored. (take note of its setting before deinstalling the current version).

If you had an 'alpha' version of SecureEntry 3.0 installed (builds 35..72) :

Do the following :

1. Save, if necessary, the file 'x:\SGMSHELL\NOUSER\EDYREGDB.VLB' (only if it exists and you want to keep on working with it after installing. Note that this file is the database for all the users and groups security components. If you decide so to maintain it, remember to enter as institution name at installation time, the string 'SER' (three characters), so that the master key used to decipher this file is compatible with the one used to encipher your old database.
2. Save the files 'x:\SGMSHELL\DLL\EDYCUST.DLL' and 'x:\SGMSHELL\EXEC\EDYCUST.CMD' if you have used the SecureEntry user exits to do anything special and wish to keep on doing it after installing.
3. If you have a user written file 'x:\SGMSHELL\EDYFILT.DLL', save it also.
4. Save also the file 'EDYSTART.CMD', from the boot drive root directory, for if you have some code there that you wish to keep.
5. If you have machine security profiles (located in the nouser directory), copy those to the first SecureEntry 3.0 installation diskette within a directory named 'NOUSER' which you will have to create. Then at installation time these profiles will be copied were they belong.
6. Execute the command 'UNINSTAL.CMD' located in the first SecureEntry 3.0 diskette. It may give you some warnings, and errors about commands not found, but upon a shutdown and reboot, will be able to deinstall SecureEntry alpha from your machine.
7. After rebooting, notice that SecureEntry alpha is no longer in the machine. Note that this is not exactly the case, since some warnings appeared in the previous steps, but enough is done so that reinstalling will succeed. Now, proceed with the installation.
8. Then install the actual copy of SecureEntry, by following the steps explained under 'Installing SecureEntry'.
9. After installing this version of SecureEntry, and before rebooting, recopy the above saved files :

   Place the user exit code where it belongs, after migrating it manually (note that there are some new entry points and that security components do not any longer require to be activated here). These are the 'EDYCUST.*' files.

   Place your original 'EDYSTART.CMD' file again in the boot drive root directory.

   Place your personal 'EDYFILT.DLL' in the 'DLL' directory.

If you have an 'EDYREGDB.VLB' file saved (security components database), then place it in the 'NOUSER' directory. (Note that you may have to remove the READONLY attribute from this file)

If you had a previous 'beta' or GA version of SecureEntry 3.0 (builds 72..) :

Follow the service procedure to update your software. This is explained afterwards, under Providing service to a SecureEntry 3.0 machine.

If you had a previous version of UCM installed.

If you have UCM 3.0 already installed, and you want to migrate to UCM V4R0 revision, then you must follow the steps explained underneath:

1. Follow the service procedure explained in the UCM administrator guide to set up UCM V4R0 at your host site.
2. Follow the service procedure to upgrade your UCM administrator workstation software. This is explained afterwards under Setting up the User Centralized Management workstation.
3. You can skip this procedure if you do not plan to activate the new UCM Refresh Branch Online procedure, since what follows is a sample guideline of how to migrate your branches and activate the Refresh branch online procedure:

   You have to migrate your branches to SecureEntry build level 191 or higher. You can do this task by applying SERVICE to your machines, as explained under Providing service to a SecureEntry 3.0 machine. Note that the service procedure does not add the policy refresh parameter to the appropriate server's **edystart.cmd** line (EDYSRV load sentence). You can force this change by adding the appropriate rexx code to a service hook (POSTSERV.CMD), as explained under Personalized installations. The EDYSRV refresh policy parameters are explained under EDYSRV and EDYFREE.

   While you are activating the refresh procedure, you can use both methods to update the branches definitions (group and resource definitions): the refresh branch online (EDYSRV.EXE with refresh parameter) and the batch process (EDYUCDIS.EXE, UCMP01, ...) method.

   Both methods can live together while you are changing the load parameter for EDYSRV process. You have to be aware of the fact that the UCM change tables at the host can not be deleted until the migration has been ended and you are still using the batch update process. To grant it, you can follow the next steps:

      When you run the UCMP01 and UCMP02 batch processes to get delta information of your branches, the Force Delete parameter must be specified with 'N' value.

      Do not change the threshold branch number. This parameter of the policy refresh must remain set to '0000' value. This is to prevent the deletion of the UCM change tables through the refresh branch online process.

      Once you have activated the refresh parameter for EDYSRV process at all of your branches, and no longer expect to use the batch update process, you can then modify the threshold branch number value to activate the purge process of the UCM change tables and set it to the

appropriate value (number of installed branches). You can do this by using the UCM users and groups management tool.

Installing SecureEntry

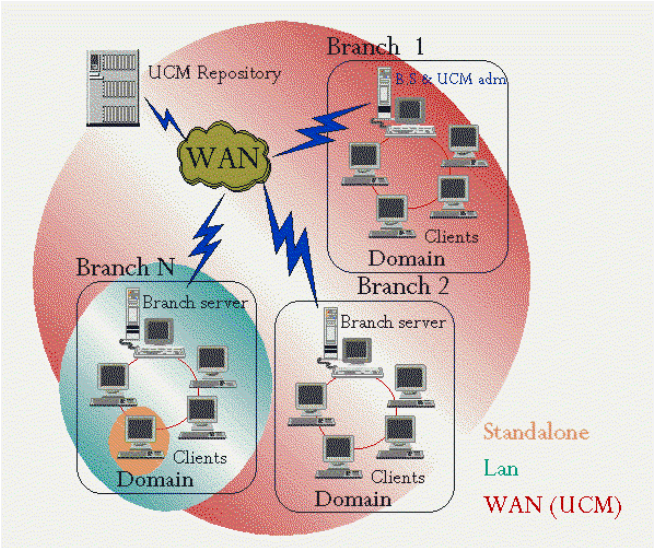
The SecureEntry install process is composed of two different subprocesses :

First, the SecureEntry installation setup, which is always interactive, allows you to define the parameters and characteristics for your installation(s), ending up by writing a response file for the second part. The module that does this work is INSTALL.EXE. This module invokes, if the user chooses to do so, the second part of the installation in order to setup the actual machine.

Then, the SecureEntry workstation installation process, where the SecureEntry files are transferred to the system, and the environment is set up in the machine so that after rebooting the machine, SecureEntry will be installed, enabled and running. This part of the installation can be either interactive or batch (no user intervention). INSTALB.EXE is the module in charge of this part.

Installation environments

One of SecureEntry's strongest points is its flexibility. That is, the ability of this software to run in different environments and to be adapted to fit different security requirements. The following picture shows the different suitable scenarios for installation.



The best part of all this is that administering/working with one, one hundred or ten thousand workstations is exactly done the same way, irrelevant of the environment, with the same tools and easiness, except for few exceptions.

This chapter provides detailed description for the different environments:

- Standalone environment
- IBM Lan server environment
- Network environment.Other networks
- Special (centralized) environments

Standalone environment

Being the simplest one, this working environment is intended for not networked machines, in which administration and runtime are done from the same workstation, which also holds the security database were the different security profiles are stored. This environment is very well suited for home machines, mobility professionals with shared portable workstations, and as a SecureEntry demonstration and training environment.

IBM Lan server environment

Under a Local area network environment, and if the network software running is the IBM Lan Server/requester one, then SecureEntry can be installed as a 'back-pack' to the network software, providing additional local security function. This is achieved by synchronizing SecureEntry and UPM signons, and allowing for Lan server administration through the SecureEntry administration tools.

After installing SecureEntry this way, all Lan Server users and groups will be automatically considered SecureEntry users and groups, with the added possibility of being assigned some or all of the SecureEntry local security component profiles.

It is important to note that under this kind of environment, you are not maintaining a 'couple' of secured and unrelated workstations, but instead creating a 'workgroup' of workstations which share a common security profiles repository. This is why one of the machines in the network plays the role of a 'server' of security profiles, being this one the machine where the SecureEntry database is created. This machine will have to be necessarily powered on and with the Lan Server software running at the time when other machines are to be connected, so that the user's signon information is available. This is the reason why the normal choice for the SecureEntry repository server machine is the domain controller machine, but being this not a mandatory option.

From the administrator's point of view, this environment is not much more different from the standalone environment, being the main differences :

The graphical administrator tool will now show buttons to configure Lan Server specific resources and user's characteristics (resource alias and user's parameters).

The batch administration tool will accept a couple of new keywords to configure Lan Server specific resources.

Group names will have to be named beginning with the letters 'SG', to identify them as a SecureEntry group definition which can accept security component profiles.

Since the underlying users and groups database is now that of the Lan Server (UPM), then its rules are enforced (naming, hierarchy, and allowable value ranges).

Since the security profiles are to be used by all the machines in the network, you should plan for them and provide all the machines in the Secure Entry network with the necessary objects to activate those profiles. For instance, you will achieve the best results by 'cloning' the workstations so that their basic desktop looks the same.

As a last comment, note that you can configure a Lan Server machine as a backup controller, in order to be able to keep on working in the event that the domain controller fails. Refer to Setting up a backup domain controller for Lan Server for a detailed description of how to do so.

Network environment. Other networks

If you have a network of connected workstations which are not running under IBM Lan Server, you can configure SecureEntry also as a 'workgroup' of workstations, i.e, sharing a common repository for security components, but in this case using the network software only as a 'transport' mechanism to allow access to this repository. The idea being that you configure the workstations so that a shared directory path exists and is used to access and establish the logged on user's security profiles, but without the ability (from SecureEntry point of view) of administering the alien network resources.

This configuration is really much similar to that of a standalone installation, being the only exception the fact that SecureEntry users, groups and security profiles are defined within the shared directory.

If you want to further integrate SecureEntry with your network software, you can always write a specific 'LMP' (logon modular procedure) which does the network signon to your own network. Samples are provided within the Secure Entry installation directory, and an explanation of the process is provided later on.

Special environments

There is one more consideration to take into account when describing the working environments : If you have contracted UCM (Users Centralized Management), then you have two more options :

One is to do password validations to RACF or similar software running at the host site. This requires an APPC link to the host running in the server machine, plus the activation of a few additional components in the network. Of course, using this option alone implies that you are responsible for maintaining a user's definition database at the host site which matches that one existing in the network (although password synchronization will be done by SecureEntry itself). Optionally, you can still do user/password validation at a host level if using UCM by means of the UCM provided RACF emulator.

The second one is to activate central administration of users, groups and security profiles, so that all administration changes are stored at the host site, within a DB2 database by using the UCM software, and the local Secure Entry databases existing in the administered networks are used as 'shadows' of this central database. Just by doing this the concept of 'security workgroup' is migrated into that of a 'corporate security policy', allowing any of the defined users to sign on to any of the workstations of any of the corporation networks.

Regular installations

Before proceeding with SecureEntry installation, and if you have contracted the UCM solution, read and perform the UCM installation procedure in order to prepare the prerequisite software in the workstations as well as in the host system.

Right before installing SecureEntry 3.0, verify that you have installed the prerequisite software, and also take into account :

1. Take note of and remove any software that you may be starting from the startup folder of the WorkPlace shell. Since you are installing in a machine to be used by several users, you probably may want to start those applications from a SecureEntry user exit, from the *EDYSTART.CMD* procedure which will substitute the regular *STARTUP.CMD*, or from the SecureEntry *EdyStart folder*, which will be installed by the product as a substitute for this function.
2. If you are installing under IBM Lan Server environment, and wish to install a SecureEntry server (not necessary for requesters), and you have IBM Lan server local security enabled (i.e, the server has HPFS386 installed), then you need to logon to Lan Server as an administrator in order for SecureEntry installation process to be able to to create the 'SGMSHELL' alias, where the SecureEntry database will reside.
3. In the same case, (IBM Lan Server installs), make sure that no groups exist defined that begin by the letters 'SG', so that they will not be confused with SecureEntry groups later on.
4. If you are installing SecureEntry in a WorkSpace On-Demand server, and wish to enable the different WSOD workstations for SecureEntry later on, then take into account the following restrictions :

   WorkSpace On-Demand should be completely installed before proceeding with the SecureEntry install process.

You must use the Security Enabling services SecureEntry emulator in your installation. (Basically, uncheck the *Use OS/2 Security Enabling Services* checkbox, as explained underneath).

You must install SecureEntry in a directory named *SGMSHELL* within the *IBMLAN\RPL\BBxx.yy* directory. This directory is the one proposed by the installation tool if WorkSpace On-Demand is detected.

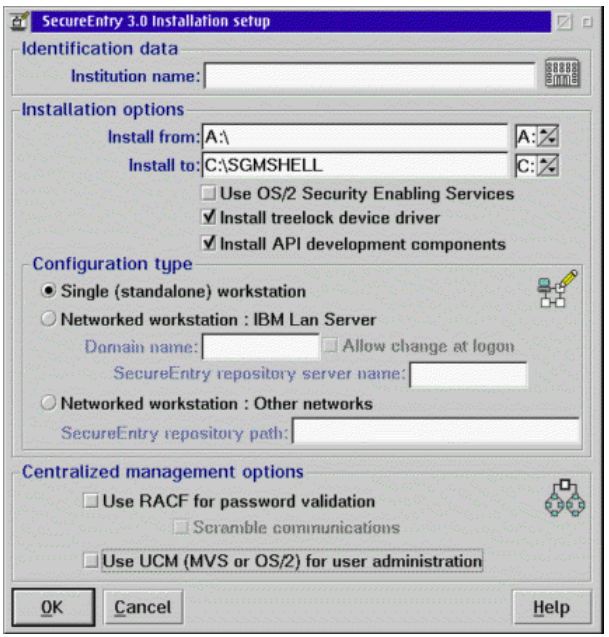
Later on, and after completing the installation process, read WorkSpace On-Demand support to know how to enable the different WSOD workstations for SecureEntry.

Once you are ready to install, just by typing :

```
INSTALL [response file path-name]
```

from the first SecureEntry installation diskette, all the installation processes will be chained, and you will be able to define your installation setup response file, and optionally configure the workstation (by automatically calling the next step : Workstation Installation utility). Note that the default response file if no other is specified, will be called 'SENTRY.CNF' and placed where the INSTALL.EXE resides (disk and directory).

The first (interactive) part of the installation process looks like :



The fields that you can fill up in this panel are :

Institution name: Enter here the name of your institution. This, together with some other information, will be used in order to generate a private master key for ciphering your security profiles data base. This is the reason why this field can not be left blank.

Install from : Enter here the path to the source of the installation package files.

Install to: Enter here the destination directory where you want SecureEntry to be installed. Note that if WorkSpace On-Demand is detected, the default directory presented here will be *IBMLAN\RPL\BBxx.yy\SGMSHELL*, which will allow you to later on enable your SecureEntry client workstations.

Use OS/2 Security Enabling Services: Uncheck this button if you want to install SecureEntry with its Security Enabling Services emulator active, instead of the real thing. Everything will work the same, except for the superuser signal, as will be explained later on. Not using the real Security Enabling Services is specially useful for memory constrained systems, or when installing under OS/2 versions for which these services are not available, such as OS/2 SMP or the different WSOD clients. Note that using the emulator is also a prerequisite for installing in Coexistence mode with other SES clients.

Install treelock device driver: Only if your OS/2 kernel does NOT support the Security Enabling Services hooks, you will need to uncheck this button.

Install API development components: Uncheck this check button if you do not intend to develop with the SecureEntry APIs in the machine(s) to install the software in. This way, you will save some hard drive space and left out from the installing process the SecureEntry source code and API information.

Standalone environment: Check this radio button if you wish to install SecureEntry in standalone mode. (see SecureEntry environments descriptions).

Networked workstation. IBM LAN server: Check this radio button if you wish to install in IBM Lan Server environment. You are then also prompted for the following values :

> *Domain name:* Enter here the name of your Lan Server logon domain.
>
> *Allow change at signon:* Check this button if you wish to allow the user to be prompted for and sign on to other SecureEntry available domains.
>
> *SecureEntry repository server name:* Enter here the name of your Lan Server domain controller computer, that is where the SecureEntry profiles database will be installed.

Networked workstation. Other networks: Check this radio button if you wish to install SecureEntry in Lan environment, non IBM Lan Server. You are also prompted for :

> *SecureEntry repository server path:* Enter here the remote path to access the SecureEntry repository from any requester machine.

Use RACF for password validation: If you have contracted UCM, then you can check this button to allow RACF password validation at signon time, or leave it unchecked to use the provided RACF emulator for user/password validation, if UCM is also installed. Note that RACF is only available under MVS environments, so you can not choose this option when installing the UCM repository server under other environments.

Playing the same role as RACF validation, you can optionally make Network Signon Coordinator/2 act as your main password synchronizer. You do not need to have UCM contracted to do so. In this case, you will have to manually configure it as explained in the provided NSC/2 LMP.

Scramble communications: In case you use RACF for users authentication (previous checkbox), then you can optionally choose to make use of the SecureEntry scrambling facility (EDYURACF) to scramble userids and passwords at transmission time, during signon. You do not need this feature if your APPC network is secure enough, or already using encryption at a lower level.

When using the RACF emulator provided by UCM, data scrambling will always be active by default.

Use UCM (MVS or OS/2) for users administration. If you have contracted UCM services, then you can check this button to allow for the workstation to be administered at signon time from a host site. If this is the case, then you may be interested in setting up the UCM environment variable **SGM_UCM_THIS_BRANCH** through a proper **CONFIG.ADD** file before installing each different branch machine, as explained under Personalized installations.

After you have filled this panel, you will be given the option to continue the installation process by automatically calling the workstation installation utility, or exiting. If you exit now, the SecureEntry response file will have been created, and you can optionally use it in other workstations invoking the workstation installation utility through a command line.

Personalized installations

It may be the case that you desire to distribute your own profiles or certain files with SecureEntry installation process. You can do so by following what is explained herein.

You can place in the first or last installation diskette whatever files you want, within the directory were they will have to reside after installing. i.e, in order to set up automatically a default SecureEntry launchpad profile as the default one, copy it (EDYPAD.INI) into a directory named 'NOUSER' within the first installation diskette.

To distribute other component files which may not reside within the SecureEntryPath directory, copy them into a created :

BOOT Directory, if you wish those files to be copied to the boot drive.

x$ Directory, if you want those copied to the 'x' drive.

If you want SecureEntry to add automatically some *config.sys* settings to the installed workstation, then create an ASCII file named *CONFIG.ADD* within the root directory of the first installation diskette, with the desired definitions. As a sample :

```
Sample CONFIG.ADD
-----------------

SET SGM_ALLOW_CAD=TRUE
SET SGM_PM_WAIT_B4_KILL=0
SET SGM_UCM_THIS_BRANCH=Branch 22, Johnson st. 4561
```

Now, if you want to do more complex processing, you can always write a post installation REXX command file, and place it in the first installation diskette, so that it gets called after the installation process. The name of the called cmd file will be *POSTINST.CMD* for new installs, and *POSTSERV.CMD* for service (upgrade) procedures. At invocation time, this hooks will be called from the SecureEntryPath\INSTALL directory.

After doing this, you can proceed with the regular installation process as explained in the previous chapter.

Workstation installation utility

INSTALB.EXE is the workstation installation utility which will do the file unpacking process, and setup the workstation with the installation options choosen in the previous step. To invoke it, type :

```
INSTALB [SERVICE] [OVERFROM:sourcepath] [OVERTO:destpath]
        [response file path name] [SERVER] [BATCH]
```

Use the 'SERVER' parameter to install a SecureEntry server machine in lan environments, and the 'BATCH' parameter if you want unattended installations (no questions asked).

The SERVICE parameter, together with the OVERFROM: and OVERTO: parameters are used only to provide service to an already installed machine, by means of overriding the source files and destination path with those specified through the command line.

Note that it is not normally necessary to invoke this process manually if interactive installations are being done, since the installation setup utility (INSTALL.EXE) will already chain to this module if the user selects to do it. Keep on reading to know what to do after this process ends.

The normal procedure to follow after this utility has run is to reboot the machine, but if you are doing a network installation, you must make sure that the file 'EDYSTART.CMD', located in the root directory of the boot drive contains the appropriate sentences to start your network software before the machine reboot. (i.e, 'NET START REQ' for requester installations, 'NET START SRV' for server installations).

Besides, if you are installing UCM and/or RACF validation features, you may want to check that the comms manager startup command is also in the SecureEntry startup file (EDYSTART.CMD). In the same case, and if you are installing the administrator's machine for UCM within the control LAN, you will have to manually setup this machine by following the Setting up the User Centralized Management workstation procedure.

Setting up the User Centralized Management workstation

You can skip this procedure if not installing the UCM feature for SecureEntry.

To set up the user centralized management workstation, go through the following steps:

1. Install DDCS/2.
2. Install LAN Server.
3. Configure Communications Manager. For details on configuring CM/2, refer to the UCM administrator's guide.
4. Configure DDCS/2 For details on configuring DDCS/2, refer to the UCM administrator's guide.
5. Install SecureEntry Release 3.0.
6. Log on as a user with administrator privileges.
7. Open the **SecureEntry Workbench** folder.

8. Make a copy of the **Users and Groups Management** folder and rename the new object to **UCM Users and Groups Management**. Open the new object settings notebook and change *EDYSNADM.EXE* to *UCMADM.CMD* in the Path and file name entry field of the Program tab. Enter as parameter : *EDYSNADM.EXE*.
9. If you want to use your own ASCII to EBCDIC and EBCDIC to ASCII conversion tables, you will have to place the following files in a directory named INSTALL of the installation diskettes:
10. `EDYA2E.DAT`
11. `This file contains 256 character codes for ASCII to EBCDIC`
12. `translation.`
13. `EDYE2A.DAT`
14. `This file contains 256 character codes for EBCDIC to ASCII`
15. `translation.`
16. By default, UCM uses its own Translation tables. Refer to the UCM administrator's guide to get information about how to install these translation tables at Host site.
17. From an OS/2 command prompt enter the statements to Bind the UCM API.
18. Run the *INSTSUB.EXE* program, which adds the host subsystems information to the UCM database, using the API previously installed on the workstation, and enables to verify the installation correctness. For do this, you can follow the next statements explained hereafter from an OS/2 command prompt:
19. `db2start`
20. `instsub`
21. `db2stop`
22. From an OS/2 command prompt, enter the statements to Bind the EDYQRYBR program.
23. From an OS/2 command prompt, enter the statements to Bind the EDYRVUCM program.

Refer to the UCM administrator's guide for details on how to set up the UCM product itself.

Setting up a backup domain controller for Lan Server

If your installation environment is of Lan Server type, you may wish to set up a machine as a backup domain controller, so that in the event that the main domain controller is down, your users may still be able to work with their machines.

This kind of support is intrinsically provided by Lan Server, but and as SecureEntry extends Lan Server repository with its own security profiles database, it becomes necessary to make sure that this database is available also when only the backup domain controller is working.

The process to follow in order to set up the backup domain controller is as follows:

Regular Lan Server backup domain controller setup

1. When installing Lan Server:

   select "Backup domain controller" when it is requested.

   install REPLICATOR and DCDB REPLICATOR.

2. Once Lan Server is installed

   Define the Backup Domain Controller in the domain, as it should be done for any additional server.

3. Tuning: several replicator parameters, such as replication interval, can be modified editing IBMLAN.INI. Refer to the Lan Server handbook for further information.

4. Runtime operative suggestions:

To set the Backup Domain Controller as the Domain Controller (when the main domain controller is down) do the following sentences in the Backup Domain Controller:

```
NET STOP  NETLOGON
NET ACCOUNTS   /ROLE:PRIMARY
NET START  NETLOGON
```

To force DCDB replication:

```
NET STOP    DCDBREPL
NET START   DCDBREPL
```

To resynchronize local and LAN Backup Domain Controller machine passwords (they can mismatch after several days of Power-Off, and this blocks replication)

```
LOGON an administrator
NET ACCOUNTS   /ROLE:STANDALONE
NET USER  machine_name  newpassword
NET ACCOUNTS   /ROLE:BACKUP
NET USER  machine_name  newpassword
```

SecureEntry specific setup

Create the backup database repository alias in the backup domain controller. You can do this by running the command :

```
NET ALIAS  SGMSHBAK \\bdcname backupdirectory  /WHEN:STARTUP
```

Where *bdcname* is the server name of the backup domain controller.

Copy the SecureEntry database to the backup controller. Do this with a regular copy command :

```
COPY   \\dcname\SGMSHELL\EDYREGDB.VLB  \\bdcname\SGMSHBAK\EDYREGDB.VLB
```

Where *dcname* is the server name of the main domain controller, and *bdcname* is the server name of the backup domain controller.

Establish a mechanism by which the above is done in a regular basis. You can use the Lan Server replicator services to do this. Refer to the Lan Server documentation for how to do this.

Grant access to this backup repository to all desired users. As a sample, you could do it for all the Lan Server users with the following command :

```
NET ACCESS SGMSHBAK /ADD USERS:RWA
```

After all of the above is done, and in the event that the domain controller fails, users will still be able to logon and be validated to the backup controller.

Operating considerations

Take into account :

There is no necessity to change the role of the backup domain controller once the main controller fails.

When logging on to the backup domain controller, you will receive a message indicating so. Similarly, logon and logoff operations will be slower than normal.

If you have logged on to the main controller and it fails, you will not be able to use the administration tools until you logoff and logon again against the backup controller.

If you use the administration tools against the backup controller, the changes you make will not be automatically updated against the main controller, unless, when the main controller is available again you do:

1. Follow the regular resynchronization procedures for Lan Server users information.
2. Copy the modified .VLB SecureEntry repository file back from the backup controller on to the main controller.

When logging on to the backup domain controller, it may be that the received error messages are less indicative than when using the main controller, unless you start the EDYSRV program in the backup controller. Note that only one EDYSRV program can be running within a given domain at the same time, so before restarting the main controller, you should unload it with the EDYFREE utility. Refer to EDYSRV and EDYFREE.

Providing service to a SecureEntry 3.0 machine

You can use the supplied diskettes to provide service and update a previous version of SecureEntry 3.0, as long as it was beta version at least. Do the following :

1. Insert the first SecureEntry installation diskette
2. Type (supposing you are providing service from diskette drive A) :

```
A:\SERVICE [sourcepath] [BATCH]
```

Use the *sourcepath* parameter to specify the path where the new files are located. By default the location of the SERVICE.CMD is used, unless executing the installed copy of SERVICE.CMD within the SecureEntry path, in which case 'A:\' is used.

Use the 'BATCH' parameter to run SERVICE.CMD in unattended mode.

The service procedure will be started and the SecureEntry software will be updated.

This service procedure will respect your users definitions and profiles, as well as your current CONFIG.SYS settings. Any personalized user exit file (EDYCUST.xxx) or naming filter (EDYFILT.DLL) will have to be migrated manually, since the service procedure will not override the one that is currently being used.

If the files EDYKILL.NOT or EDYLOGS.STR residing in SecureEntryPath\NOUSER are different from the default ones (have been customized for your installation), they won't be automatically migrated either, so you will have to manually apply the new changes incorporated into the default file, which will be located in the SecureEntry path\EXEC directory for your reference.

WARNING: If you do code development in your machine, note that the *API* directory will be overwritten by this procedure, so take a minute to backup your modified sources before servicing.

Note that the service process will not be completed until you reboot the machine after running the above mentioned command. After such reboot, all previously locked DLLs will be replaced, and the temporary service directories will be deleted, plus a subprocess named 'EdyStart' will be launched as soon as the Workplace Shell is loaded. This subprocess mission is to make sure that all workbench objects are refreshed and service changes applied to its definition, so please let it run until it ends by its own and do not interrupt it.

Note also that calling the SERVICE.CMD program is an alternative to calling the INSTALB program with the appropriate parameters for servicing an already installed machine, as explained in the previous section.

As a last comment, you will normally require administrator privilege for running this process. For remote distribution environments, the service procedure will work correctly independently of the logged on user as long as :

1. It is invoked in batch mode ('BATCH' parameter)
2. It is invoked from a superuser context process (i.e, started as a daemon from EDYSTART, CONFIG.SYS or any superuser context user exit). Read About processes and running contexts to know more about running contexts.

What to do if the installation fails

In any case, if the installation does not succeed, you will be able to browse the 'SENTRY.LOG' file located either in the boot drive root directory (for preliminary installation errors), or in your 'x:\SGMSHELL\INSTALL' directory, and look for the cause of the error. Note that if the error happens before the 'setup' phase, (the one immediately after the disk copy process), then you ought to be able to delete the directory, correct the error and retry. If the error happens after that point, the installation utility will do its best to leave you with an 'usable machine', but you are advised to take a quick look to your 'config.sys' file, and verify that it does not contain any SES-related line, and looks normal so that you can reboot the machine if necessary. If this was not the case, this

critical file is backed-up into your 'x:\SGMSHELL\INSTALL' as 'config.sen', so you can recover it. (Same thing with the 'startup.cmd' file).

If all of the above fails, please contact us and we will be very pleased to help you.

What to do after installing

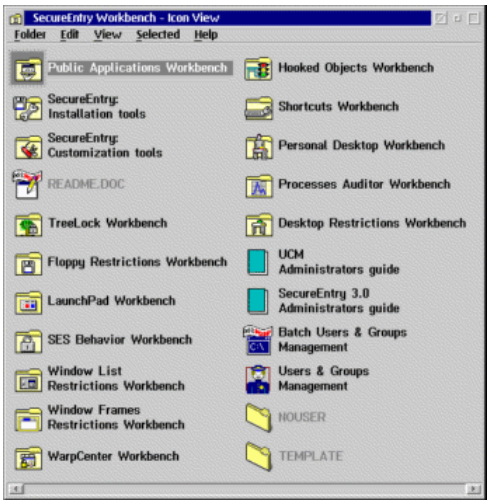
After the installation is completed, you will have to reboot the machine, and on power up, a logon panel will be shown. The process to follow is :

Standalone workstations : Sign on as user 'EDYADMIN', password 'PASSWORD', changing it (this userid is created with the password expired). Proceed to define other users, groups and security profiles.

Network server machines : Sign on as a network administrator, and proceed to define the SecureEntry users, groups, and security profiles.

Network non server machines : You can start working straight away as with any other net machine.

Once you have signed on as an administrator, you will have a newly created folder on your desktop : SecureEntry Workbench. It contains :



One folder with the SecureEntry installation tools object, which also contains a shadow to your original launchpad, plus a shadow of the SecureEntry signature file.

One folder with the SecureEntry customization tools object, which contains already customized program objects for you to use while making a custom solution, with functions such as logon, logoff, etc..

An icon for reviewing the last minute changes file (README.DOC)

A folder for each of the available components with its particular workbench. Each of this folders contains basically the necessary icons to test, create, and edit profiles of its kind through drag and drop operations. Note that the 'Desktop restrictions workbench' folder is a bit more complicated than that, since you can deal with text and binary profiles.

Two icons representing the online reference documentation.

Two icons for invoking the administration tools (batch and interactive).

A shadow of the 'TEMPLATE' directory, where some sample profiles are located, plus templates of the standard SecureEntry components for you to create new profiles.

A shadow of the 'NOUSER' directory where you may wish to place the machine-level security component profiles (default values). Note that in order to be recognized, these profiles have to have a default name, which is the same as that of the template for the given component that can be found in the 'TEMPLATES' shadow folder. You will find within this folder the EdyStart object, where you can place a shadow of all the objects that you were starting previously through the system's startup folder.

```
*********************************************
 IMPORTANT

 Whenever you change a  NOUSER  security  profile,
 remember to manually activate (test) it, in order
 to grant that the profile is effectively read.
*********************************************
```

The general idea is that you work and test with each of the component's workbench, until familiar with it, and then create the profiles which you later will assign to the different users and groups by dropping those into the graphical (interactive) administration tool components container of a given user or group.

To define the users, groups and security components, you can open the newly created 'SecureEntry WorkBench' folder of the desktop, and use either the batch or interactive administration tool. Refer to the SecureEntry documentation on how to configure each component and how to assign them to users and / or groups.

After installing, note that all components have their own default restriction settings active at startup, being those :

For the floppy access, to allow no access to the diskette drive. Note that the installation program places already a profile with full access into the NOUSER template, so although the default is no access, you will have access to it after signon.

For desktop restrictions, to apply no restrictions to existing objects.

For the Launchpad, to not present any launchpad. Note however, that SecureEntry sets up a copy of your original launchpad in the *installation tools* folder, so that if you wish to use it you can, either drop it into the opened *EdyStart* object, or create a launchpad security profile with the same contents and place it in the *NOUSER* folder or assign it to a user/group, as explained afterwards.

The WarpCenter component will create a SecureEntry WarpCenter security profile during the installation process with the same contents as your original WarpCenter, and place it into the *NOUSER* folder, making it the default WarpCenter.

For the System Menus, to not restrict any application.

For the Window list, to behave as standard in OS/2.

For SES behavior :

1. Lockup timeout set to 3 minutes, with standard size unlock dialog and without logoff/shutdown buttons nor a maximum number of unlock attempts.

2. Show options dialog at Ctrl-Alt-Del.
3. Show default bitmaps for lockup/screensaver.

For the File Access (treelock), to allow full access to anything except the Implicit restrictions.

For the others, no special definitions active.

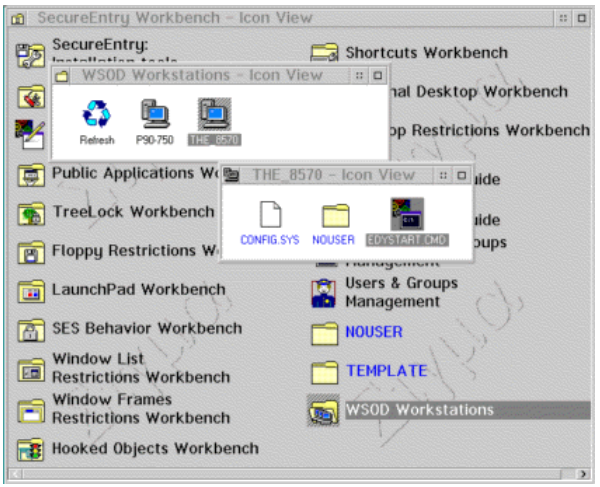
So, if you wish to use any of those, you may have to define and assign a profile either to your group or userid, or to the machine configuration. The easiest way to do it is to drag from the given component profile template object located in the 'template' folder within the SecureEntry workbench folder, an instance of the desired profile, and dropping it into the nouser folder or your own user/group component container of the Users and Groups Management application, and edit it from there. You can edit the profile either by drag and dropping it into its editor (located in its component's private workbench), or by double clicking over it (if within the Users and Groups graphic administration tool).

WorkSpace On-Demand support

If you have installed SecureEntry in a WorkSpace On-Demand server (you can read how to do it under Regular installations), and wish to use it in the different WorkSpace On-Demand client machines, you will have to use the WorkSpace On-Demand SecureEntry enabler program to configure the different client machine images so that they effectively load SecureEntry at RIPL time. This utility is located within the *SecureEntry installation tools* folder of the *SecureEntry workbench* of the WSOD server machine.

Note that all your WorkSpace On-Demand client machines will then use the server's copy of SecureEntry code, thus avoiding the necessity for installing SecureEntry separately for each client, and also making software updates easier.

Under this environment you will also find a new folder in the *SecureEntry workbench*, named *WSOD Workstations*, that contains shadows of the different *NOUSER* folders for the different machines, which allow you to easily assign security components that define the default behavior for the related client workstations. You will find there also a shadow of the related workstation *CONFIG.SYS* file, and, if any, the startup command script file, *EDYSTART.CMD*.



Note the *Refresh* program object present in this folder. It must be used whenever you enable your workstations indirectly, creating a workstation instance by deriving it from a machine type that has been SecureEntry enabled. In this case, the folder contents can be updated to reflect the new workstations by double-clicking over this icon.

Once you have your workstations SecureEntry enabled, then configuring a specific WorkSpace On-Demand desktop is as easy as administering a *SecureEntry public applications* security profile using The Public Applications component, which you can then assign to the different users, SecureEntry groups, or even place them as default desktop profile for the different WSOD client machines, by copying it to the related workstation's *NOUSER* folder.

The WSOD enabler

The WSOD enabler

The *WSOD enabler* application allows you to enable/disable WorkSpace On-Demand workstations or machine classes so that they effectively load SecureEntry at RIPL time. Note that you need administrator privilege to be able to run this utility.

The first time the *WSOD enabler* is run, it will ask whether you want to enable the SecureEntry server for WorkSpace On-Demand. Upon positive response, it will process the required steps so that subsequently regular WSOD workstations and/or machine classes can be SecureEntry enabled.

You can have at any moment any combination of enabled/disabled machine classes and/or workstations. Enabling a given machine class is only useful for avoiding the enable step for workstations of its kind that will be defined afterwards.

Interactively calling the WSOD enabler

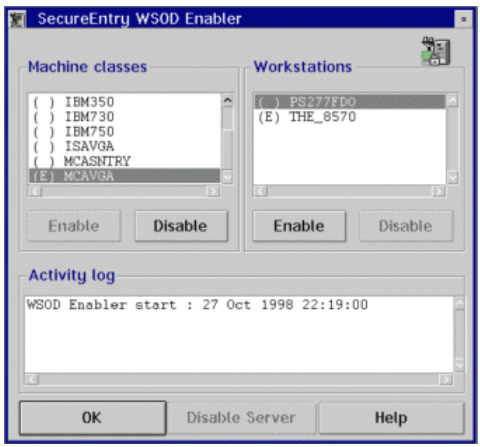
The WSOD enabler, when called without parameters, will work in interactive mode, allowing for manual enabling/disabling of WSOD workstations or machine classes.

You can run the program by typing :

```
SEWSODEN
```

From the SecureEntry path, *INSTALL* directory, or by double clicking its related object located within the *SecureEntry Installation tools* folder of the *SecureEntry workbench*

You are then presented with the following dialog :



Where the upper left listbox represents the currently defined WSOD machine classes, with an indication on whether that particular machine class has already been SecureEntry enabled. The upper right listbox represents

the currently defined WSOD workstations for the selected machine class, also with an indicator of its current enabling state. The lower listbox contains information about the carried-on actions. You can then proceed with the following actions:

Enable class This button will cause the SecureEntry enabling of the currently selected machine class, and up on request, also of its related workstations. Once a given machine class is SecureEntry enabled, all the workstations subsequently derived from it will also be SecureEntry enabled.

Disable class This button will cause a given (selected) machine class to be SecureEntry disabled, and also upon request, the disabling of its related workstations.

Enable workstation This button will cause the selected workstation to be SecureEntry enabled, so that upon the following boot, the workstation will load SecureEntry as part of its RIPL process.

Disable workstation By pressing this button you can SecureEntry disable a given (selected) workstation, so that it becomes again a 'regular' WorkSpace On-Demand client, without SecureEntry.

Disable server If all your machine classes and workstations are SecureEntry disabled, you will be able to use this button in order to undo the server enabling process that was carried out the first time that the *WSOD enabler* was run. When this process is ran, the application terminates automatically.

Batch calling the WSOD enabler

Alternatively, you can call the WSOD enabler in batch mode, by invoking it with the appropriate parameters :

```
SEWSODEN [ES | DS | EA | DA | EC:classname | DC:classname |
          ECW:classname | DCW:classname | EW:wsname | DW:wsname | EX ]
```

Where

ES

Can be used to force server enabling at load time, if necessary

DS

Can be used to force server disabling. Note that the program will terminate afterwards

EA

Enable All, enables for SecureEntry all machine classes and workstations found disabled

DA

Disable All, disables all machine classes and workstations found SecureEntry enabled enabled

EC

Enable Class, enables for SecureEntry the specified machine class

DC

Disable Class, disables for SecureEntry the specified machine class

ECW

Enable Class Workstations, enables for SecureEntry the related workstations of the specified machine class

DCW

Disable Class Workstations, disables for SecureEntry the related workstations of the specified machine class

EW

Enable Workstation, enables for SecureEntry specified workstation

DW

Disable Workstation, disables for SecureEntry the specified workstation

EX

When this parameter is processed, the program is exited

Note that all parameters are processed sequentially. For instance, the command :

```
SEWSODEN  ES EC:MCAVGA ECW:MCAVGA EX
```

Means : Run the enabler, enable the server if necessary, then SecureEntry enable the WSOD machine class *MCAVGA*, follow on by enabling all the WSOD workstations of the same type, and finally, exit, all in unattended mode.

Return codes and problem determination

The program will return 0 if no errors have been posted, otherwise the return code will be the number of enable/disable errors encountered.

For problem determination, all activity will be logged in the SecureEntry installation log file *SENTRY.LOG*, located in the *INSTALL* directory of the current SecureEntry path.

Tips for administering SecureEntry WSOD environments

Personalizing workstations

If you want to enable your machine classes or workstations with a different set of environment variables than those provided by default, you can place these in a file named *CONFWSOD.ADD*, with the same format as previously explained under Personalized installations for the file *CONFIG.ADD*. This file must reside in the *INSTALL* subdirectory of your SecureEntry path.

Additionally, if you want to run some more complex processes during the enabling and disabling of workstations and/or machine classes, the *WSOD enabler* calls, if found, the following user exit commands:

POSTENCL.CMD Called after a machine class has been enabled for SecureEntry, receives the class name as a parameter.

POSTDSCL.CMD Called after a machine class has been disabled for SecureEntry, receives the class name as a parameter.

POSTENWS.CMD Called after a WSOD workstation has been enabled for SecureEntry, receives the workstation name as a parameter.

POSTDSWS.CMD Called after a WSOD workstation has been disabled for SecureEntry, receives the workstation name as a parameter.

This files must be placed also in the *INSTALL* subdirectory of your SecureEntry path.

If you want to place a specific read only file as available for all enabled workstations, such as an *EDYCUST* user exits file, just put it in the server, SecureEntry path, *EXEC* or *DLL* directory.

The SecureEntry R/W tree is located under *x:\IBMLAN\RPLUSER\machinename\SGMSHELL*, for if any read/write additional SecureEntry file is required.

For the startup file, you can place a *EDYSTART.CMD* file in the *x:\IBMLAN\RPLUSER\workstationname* directory for it to be processed by SecureEntry startup. Note that if a *STARTUP.CMD* file exists in the same directory before enabling the intended workstation, it will be migrated to *EDYSTART.CMD* during the SecureEntry enabling process.

Which components will/will not work

You can use all components and features of SecureEntry while running under a WorkSpace on Demand environment, with the following exceptions/restrictions :

> **The WarpCenter component** This component is not supported, since if it were, it could only be used as an 'applications launcher' due to WSOD restrictive working strategy. For this reason, if you need such functionality, we suggest you to use a SecureEntry launchpad instead.
>
> **The EdyStart object** Because the Workplace Shell is restarted every time a WSOD workstation logons, SecureEntry ignores the EdyStart object contents to avoid restarting those at every session signon. We suggest you to use the command file EDYSTART.CMD to provide for machine power up automated process launching, or use the appropriate user exit to do so. If you want to provide for per user session process launching, use then a personal desktop profile with a dynamic folder configured as 'autostart'.
>
> **All desktop related components** Specifically speaking, the desktop restrictions, launchpad or the personal desktop component are fully supported in this environment, but you should take into account that the functionality for those is also restricted by WSOD itself. For instance, you will not be able to drop/delete objects in a launchpad, or create new shadows inside a personal folder unless you have signed on as an administrator.

Besides, and for customization purposes, it is possible to create a *SecureEntry Workbench* within a WSOD workstation, provided that :

1. You have signed on as an administrator
2. You have access to a command line or drives object
3. You are using WSOD version 2.0 (minimum)

To do so, run the program EDYCRWRK located in your SecureEntry path, INSTALL directory.

This way, you will be able to configure and assign SecureEntry profiles to your groups and users, based on the *real objects* and environment of your client workstations. This is very useful for configuring workplace shell related SecureEntry components, such as Desktop restrictions, launchpads,... since it is often impossible to create a desktop with exactly the same client objects in the WSOD host server machine, due to the fact that the operating system base code is different.

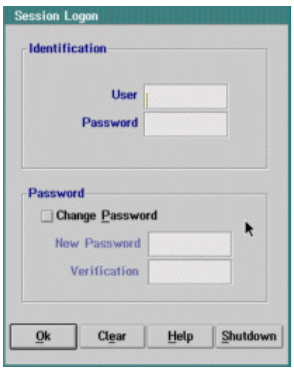
Using SecureEntry

If you are a SecureEntry user, most of the product will work transparently for you. Only the session control functions will show that the machine is enhanced with SecureEntry.

- Session logon
- Session unlock
- Session logoff or shutdown
- Ctl-Alt-Del

Session Logon

At machine startup, or after an end session event, you will be prompted with the following panel :



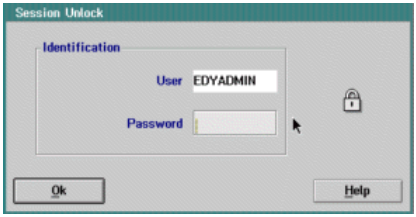
Just enter here your userid and password, and press <OK>, and you will be able to start your regular work session.

Session unlock

Several events can cause the session to lockup, being those :

- Pressing the 'lockup' button on the launchpad
- Selecting the 'lockup' function within the desktop popup menu
- Automatic lockup function being activated after a period of system inactivity.

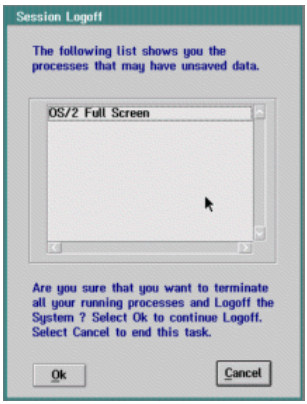
If this happens, you will see the following panel displayed :



Just reenter the same password you used to signon, press <OK> and the work session will continue.

Session logoff or shutdown

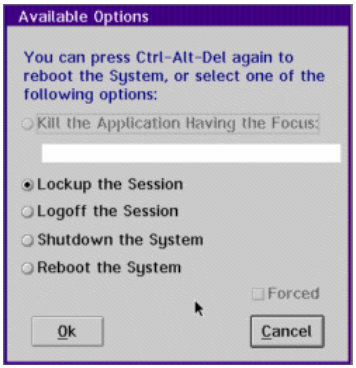
You can, at any time select 'logoff' or 'shutdown' from the desktop popup menu, or any of the system buttons placed on the launchpad. The difference between both functions is evident. Shutdown will close all active applications and allow for machine power-off, whereas logoff will reenter the session logon (signon) panel. When issuing any of this functions, the system may prompt you with the following panel :



Which reminds you of those applications which you may have to close in an orderly way before being able to grant that a logoff or shutdown process will not lose any data. Press <OK> if you do not care about closing these applications, or <Cancel> if you are not sure and want to close the applications manually.

Ctl-Alt-Del

Pressing this key combination at any time can cause the system to prompt you with the appropriate action to process. Note that this only happens if you have this function setup for you in your personal profile.



You can then choose among :

Killing (terminating) the application with focus.

Session logoff.

Machine Shutdown.

Lockup the machine.

Machine reboot. This should be considered an emergency shutdown.

Choose one of the options and press <OK> to continue processing.

Security components - Base and PM

Supposing you have just installed SecureEntry, and you are responsible for defining and administering the installation, you need to do the following tasks :

1. Define a security policy for your installation, that is, the roles that will be playing the different users. For instance, you could envisage several groups of users with common needs (i.e, tellers, developers, administrators,...) This will give you a basic idea on the SecureEntry groups that you will need to define.
2. Define, in a per group basis, what kind of interactions will each of the groups require with the machine, at a component level. That is :

   What applications need to be visible for each group, (required for work), and the degree of flexibility you want to give to each one in order to personalize his desktop. (Desktop restrictions component).

   What will be a good launchpad/smartcenter content for each group, which contains the most frequently required functions.

   What kind of behavior will you allow for standard window frame applications, i.e, you may not want to let users to mess with the font selections of a 3270 emulator. Also, if you want to be strict with the positioning and sizing of these frame window applications (very useful for inexperienced groups of users). This is what the Window restrictions component allows you to customize.

   Look and feel of your installation during working sessions. i.e, background bitmaps at lockup, behavior at Ctl-Alt-Del pressing, Screen saving feature, present a shutdown button at lockup,... This is what the SES behavior component lets you define.

   Floppy access policy. I.e, which users may have access to the floppy drive, and whether this should be encrypted or not. (Floppy restrictions component).

   Define, at a file system level, which files/directories will a given group of users be able to access, also if necessary in a per process basis. (Tree lock component)

   Define which accelerator keys to define for fast object opening (shortcuts component).

   Establish which objects should be hooked at open time to require a password or similar (hooked objects).

   Define the aspect and required contents of the window list (window list component).

   What about memory folders and personal folders ? will those be required in your installation ? (personal desktop component).
3. Extract, from the above definitions, the common profiles for all groups, and the exceptions which should be assigned in a per user basis. Discard any component for which you do not want to establish restrictions.
4. Build a table describing the different needs you have, as the following one :

Group	Requirements	Component
Default	Have the standard lockup bitmaps, ctl-alt-del to prompt	SES
ADMINS	Do not need any restriction, Special	Launchpad

	launchpad to launch ADMIN tools.	
TELLERS	Access to the banking application. Disable moving of icons. Enable access to AMIPRO. Only read floppy/encrypted. ctl-alt-del to do nothing. Launchpad with bank application. Disable Fonts manipulation from emulators	Desktop, Treelock, Floppy, SES, Launchpad, Windows B.
DEVELOPERS	Have general access. Launchpad special with development tools	Desktop, Launchpad
Individual requests	Some developers may need access to floppy in R/W encrypted mode	Floppy

5. Create and test the different profiles using each component workbench, assigning names to those in the previous table.
6. Build a table like the following :

Component	Default	ADMINS	TELLERS	DEVELOPERS	Individual
DESKTOP	All invisible	All visible	DESK1.INI	DESK2.INI	
LAUNCHPAD		LP1.INI	LP2.INI	LP3.INI	
WINDOWS			WB1.INI		
SES	SES1.INI		SES2.INI		
FLOPPY			FLOP1.INI		FLOP2.INI (*)
TREELOCK			TLK1.INI		

```
(*) For the specific developers which need diskette access
```

7. Use the interactive or batch users and groups administration tool to create the different users and groups. Take into account that SecureEntry groups for IBM Lan Server environment should begin with the letters 'SG'
8. Use the interactive or batch users and groups administration tool to drop into the different users and groups their assigned profiles.
9. Drop into the NOUSER folder the profiles for default restrictions (valid for all users/groups without a specific profile for a given component). Remember to rename those profiles to the default component profile name to be identified by the system. That is the name of the given component's template profile within the TEMPLATE folder.
10. Decide on whether it is necessary/how to install any of the other components supplied such as :

    Boot protection

    File Integrity check

    Startup process

    User exits

11. Test your solution
12. At any time, if you want to change any assigned component profile, once it has already been assigned to a user, you can do that by directly double clicking the component icon within the interactive administration tool, or writing a batch process file with the change to be processed by the batch users and group management tool.

Follows a list and description of how to configure and work with the different components, as described under this and the following chapter :

Base and presentation manager related components :

- Floppy restrictions component
- SES behavior component
- Windows behavior component
- The Window list component
- Treelock component
- Startup process
- Boot protection
- File integrity
- Processes auditor component

Components related to WorkPlace Shell :

- Desktop restrictions component
- The Personal desktop component
- The Hooked objects component
- The Launchpad component
- The WarpCenter component
- The shortcuts component
- The Public Applications component

The Floppy restrictions component

Being this one the easiest component to work with, we will describe it first.

The Floppy restrictions component allows to configure the way that the user will be able to 'see' the floppy diskette drive, with several options :

- No access at all
- Read/write access

Read/write access with encryption

Read only access

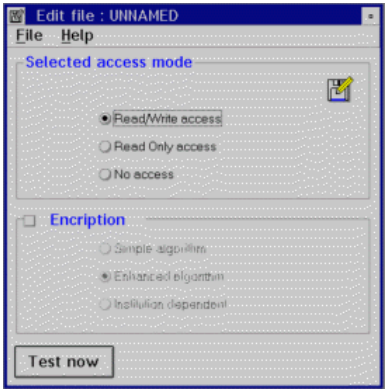
Read only access with encryption

Besides, and if encryption access mode is selected, then you have te option to choose either a generic encryption algorithm, or one which is dependant on the institution and configuration of your SecureEntry network, so that diskettes created using this mode will not be readable from another SecureEntry configured workstation unless it belongs to your institution. If you choose a generic algorithm, then it can be the default one ('light encryption'), or an enhanced one which, although less performing, is somewhat more secure.

In order to work with this component, please open the Floppy restrictions Workbench folder located in the SecureEntry Workbench folder.

You can create a profile for this component by dragging a copy from its component profile template (name is EDYFLOPP.INI).

You can edit the different options by dropping the profile into the 'Floppy restrictions editor' program object, which has the following aspect :



You can test a given profile by dropping it into the program object named 'Test floppy restrictions'.

By last, you can translate disks from one format to another one with the 'Floppy translator' utility object, located also within this folder, without the need to reformat the diskette. Please run the program and refer to the associated help to know how it works. Note that this utility is specially efficient as it can do a translation of only the logical found data on the disk, skipping the need to write all of it unless completely filled up. SecureEntry supplies you with two versions of this program, being one graphical, and the other command line. The names are :

EDYTFLOP.EXE For the graphical one, and

TRANDISK.EXE For the command line one.

They both reside within the EXEC directory of the SecureEntry path.

API and modules description

API and modules description

What follows is the description of the Floppy restrictions component modules:

Floppy disk filter

Loaded at CONFIG.SYS processing time through the following sentence:

```
BASEDEV=EDYFLPY.FLT
```

Note that to avoid problems with the **XDF** filter, this filter should be loaded after the XDFLOPPY.FLT filter.

Messages:

Successful operation:

```
IBM SecureEntry Floppy Disk Filter V 1.00
```

Warning: none.

Error: none.

Return codes: successful operation lets the system to continue the boot process. So does at failure, but in this case the Floppy disk filter remains unloaded.

EDYFLINI

For each profile (INI file) you want to create you must run the following utility:

```
EDYFLINI.EXE [IniFileName]
```

If the Floppy disk filter is loaded, you can set a floppy disk access immediately through the Push-Button TEST NOW.

You have the facility to load, save and create new INI files.

EDYFLPY

To activate an access right, you can build your own application calling the Floppy disk filter API, or you can run the provided program:

```
EDYFLPY.EXE [/I:[drive:][pathname][filename][.INI]]  | [/V:Value]
```

If passing an INI file name, you will activate the access right stored in it. If passing a value, you will activate its related access right.

Messages:

Successful operation:

```
Ok
```

Warning: none.

Errors:

```
Usage: EDYFLPY [/I:[drive:][pathname]filename.[.INI]]  | [/V:Value]]
               Value in [10,20,21,22,30,40,41,42,50]

System Error (hex)=__ (dec)=__ querying for file ________

The Floppy Disk Filter API returned  (hex)=__ (dec)=__
See EDYFLAPI.ERR for more information
```

The file EDYFLAPI.ERR is written in appended mode. For each logged error, the following information is kept:

```
Error produced on WEEKDAY, DATE AND TIME
----------------------------------------
 Source Module Name: SOURCE MODULE NAME
 Compilation Date  : COMPILATION DATE
 Compilation Time  : COMPILATION TIME
 Source Line Number: SOURCE LINE NUMBER
 Logged Error      :  (hex)=RC (dec)=RC
```

Return codes: successful operation or warning returns the value 0 to the command line. A failure returns its value to the command line.

Floppy disk filter API

The Floppy disk filter API has the following entry points, defined in EDYFLAPI.H:

```
ULONG _System EDYFloppyAccessFromIni(PSZ  FileName);
```

If FileName is not NULL, it must point to the name of an INI file gathering data about the access rights of a given user. If FileName is NULL, if the file %SGM_SHELL%\NOUSER\EDYFLOPP.INI exists, the settings in that file will be activated; if that file does not exist, no rights to the floppy drive will be given.

```
ULONG _System EDYFloppyAccessFromValue(LONG  Value);
```

Value must be one of the five access rights.

```
ULONG _System EDYGetIniValue(PSZ FileName, PLONG Value);
```

Reads the access right defined in the INI file FileName, but it does not update current access rights to floppy disk.

```
ULONG _System EDYSetIniValue(PSZ FileName, LONG Value);
```

Sets an access right in the INI file FileName, but it does not update current access rights to floppy disk.

Messages:

Successful operation: none.

Warning: none.

Error: none.

Return codes: successful operation returns the value 0 to the caller. A failure returns its value.

The SES behavior component

The SES behavior component allows you to configure 'system level' settings for the behavior of the system during session control functions, such as :

Secure startup bitmaps

Desktop background bitmap

Lockup behavior and appearance

Screen saving function

Systemwide hot keys disabling and/or hooking

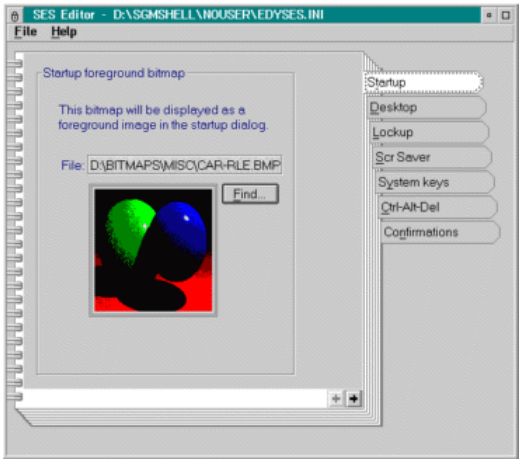
Ctl-Alt-Del behavior

Whether to show which processes at logoff/shutdown time

In order to work with this component, please open the SES Behavior Workbench folder located in the SecureEntry Workbench folder.

In order to create a security profile for this component, drag one from the component profile template object (named EDYSES.INI).

In order to edit one of these profiles, drop it into the 'SES Behavior restrictions editor' program object, set up your preferences and then save it as with any other editor. This editor looks as follows :



Note that this component allows you to configure very different machine behavior characteristics, such as control-alt-del behavior, and/or which processes to show at logoff/shutdown time. Take into account :

This component can be used to configure the startup application bitmaps, which will be the 'background bitmap' used for lockup, and the specific 'startup bitmap' as the logo one, when a profile of its kind is placed in the NOUSER directory.

Within the 'Show applications at logoff' page, you ought to configure those applications for which you want to override the prompting behavior, i.e, whether to show them in the logoff applications list or not. Those can be configured by process name, or by session title, Noting that all VDMs programs are seen as the same application process, named either *SYSINIT*, or *VDM*. by SecureEntry.

In order to understand what will be the behavior of the mentioned dialog at shutdown or logoff, follows a step by step description of its process :

1. First, an initial process list is being built up with the processes to kill. This list includes all processes running in the system except :

   All superuser context running processes, if performing a logoff (no shutdown) operation. (see About processes and running contexts)

   Processes already dying.

   Processes running under superuser context, detached.

   Reserved processes, such as PMSPOOL, PMSHELL or SecureEntry daemons.

   All processes belonging to screengroup ID 0, such as HARDERR.EXE.

2. Then, and one by one, all the elements remaining in the list (those that will be killed by the ongoing process) are passed through the configured list of filter restrictions, marking it as 'to show', 'not to show' or 'default action', in order to build the list of processes to show.

3. Those processes marked as 'default action', i.e, not directly specified by the security restriction profile filter, are handled as follows :

Processes running under a Dos Box (VDM) will be shown.

PM or detached processes will not be shown.

OS/2 fullscreen or windowed processes will show only its more deeper descendant process.

Note that you can specify within the filter list, either the process name, or the process title, by enclosing it between double quotes. In both cases you can use wildcards. If you want to know the process name for a given title, make it appear in the list at logoff or shutdown, and then double click within the listbox area so that the list will switch between showing process titles and process executable names.

Default processing for the foreground startup bitmap is a bit tricky, since the default value will be the 'Moving bitmap' of the screen saver page, if any. Failing that, a default OS/2 bitmap will be presented.

The 'Test SES behavior' program object can be used to test any profile for this component. Drop a given profile into it to do so.

For last, this component is fully supported, independently of the fact that the real Security Enabling Services are being used. If they are not, SecureEntry will provide with the equivalent function.

The Windows behavior component

The windows behavior component allows you to configure part of the general behavior of application frame windows whenever a given security profile of its kind is active. Among other things you can set up the frame window initial size and position, and restrict some or all of the system menu options that the application may display.

In order to work with this component, please open the Window Restrictions Workbench folder located in the SecureEntry Workbench folder.

In order to create a security profile for this component, drag one from the component profile template object (named EDYWIN.INI).

To modify a profile of this type, you can drop it into the 'Windows Restrictions Editor' program object, edit the frame window settings you wish to, and then close it as if working with any regular editor :

In order to test a given Window Restrictions profile, just drop it into the 'Test window restrictions' program object.

There are several details to take into account when working with this component :

1. The 'delay' configurable parameter for the Window position/size setting should only be used for those applications which 'refuse' to position/size themselves with delay value of 0. This is because some applications reseize their windows some time after window creation.
2. You can setup restrictions in a per window title name basis, or in a per process basis. To do so :

   If you choose to configure by title name, you should configure the window title names exactly as they appear in the frame window title, enclosed between double quotes.

   If you do configure by process name, the process name has to be that of the process owning the dialog procedure for the desired window. Do not enclose the process name between quotes. Note that this option is only advised when configuring by title results impossible, since restrictions may end up applied to more than the desired windows. Note that all DOS Box and OS/2 windowed sessions are 'owned' by the first pmshell (SESSHELL.EXE when working with the Security Enabling Services).
3. Both window title names, process names and window Menu strings that you wish to deactivate accept wildcards in their specification, and are processed in a case insensitive manner, orderly from the first to the last one, as appears within the appropriate editor's listbox. Note that only the first matching entry is processed.
4. You can only restrict reseize capabilities or initial position and size for windows which are 'sizeable', that is, have a sizeable border.
5. In order to configure menu strings to disable, the following tricks may be of interest :

   The scope of menu controls affected is :

   1. For window frames specified by window frame title, all menus and submenus that are owned by a frame with the same title.

2. For window frames specified by process name, all menus whose window procedure executes within the context of the specified process (same process id).

The frame 'Desktop' owns all first level menus of the desktop, whereas the process 'PMSHELL.EXE' owns all menus for all desktop objects, including opened folders menus.

You can configure default menu strings to disable, to be disabled in all cases, by doing so within the definitions for the <DEFAULT> window.

As a samples of all this,

To disable the *End Session* menu item from the desktop system menu:

```
Configure the menu string End Session for a window named  "Desktop"
```

To disable the *Find* item from all the desktop menus, including opened folders:

```
Configure the menu string  *Find* for the process named  *pmshell.exe
```

Note the asterisks, since the *F* is frequently an accelerator key, and the menu item can have also usually the form *Find*...

To universally disable the *Find* item from all desktop, folders, and applications menus of the system:

```
Configure the menu string  *Find* for the <DEFAULT> window name entry.
```

6. When configuring restrictions for WIN-OS/2 applications, take into account :

   You can set an initial position for those, and restrict the reseize, move, close, maximize, minimize or hide capabilities of the window, but the effect on the screen will be different from the same restrictions applied to an OS/2 PM application, since the menu entries will not be disabled, although any attempt to invoke those will revert to an allowed state for the window.

   You can not deactivate menu items by name (lower right corner list on the editor), since the WIN-OS/2 applications menu items do not belong to the OS/2 frame window which owns them.

   Remember to mark the application entry as 'DOS' type.

7. Finally, a brief description of the initial state radio buttons :

   If you choose to fix an initial window position and select the 'Restored' radio button, then the window will be restored to that specified size and position right after creation plus the configured 'delay' time.

   If you choose to fix an initial window position and select the 'Maximized' radio button, then the window will be maximized to its default maximized position and size right after creation plus the configured 'delay' time.

   If you choose to fix an initial window position and select the 'Minimized' radio button, then the window will be minimized to its default minimized position and size right after creation plus the configured 'delay' time. The window frames behavior component will then wait until an attempt is made to maximize or restore the window, and then force the 'restored' state with the configured size and position, unless either

      The window is configured to not allow the maximize state, or

      The configured width and height are both set to 1 (minimum)

   In this case the forced state will be the maximized one.

In any case, it becomes clear that in order to be able to choose a given initial state, the corresponding state allowance flag must be enabled, i.e you can not set an initial state of 'maximized' if the 'maximize' checkbox is unchecked.

8. When restricting position and sizing of windows, special care has to be taken with applications that do their own adjustments. A clear such sample are the CM/2 3270 emulator windows when configured with 'variable font size'. There is the risk in this case that both applications (SecureEntry and the window owner) start 'fighting' to setup a different position or size for the window. If such a problem appears, there is normally an application option that you can deactivate, which is the cause for the application autoadjust. This is solved in the CM/2 sample, by setting a fixed font size for the emulators.
9. Be careful when 'forcing' initial window states that applications do not allow in normal operation (i.e, without SecureEntry). They will possibly not work.

And for last, take into account that this component does heavily rely on defacto system behavior assumptions, and pushes the Presentation Manager limits quite high, being designed to cover a high percentage of the requirements in this area. What this all means is that the end result may just not be the desired one in a few cases, without real possible solutions. In any case, we would like to hear about the specific case, and try to find out one if possible.

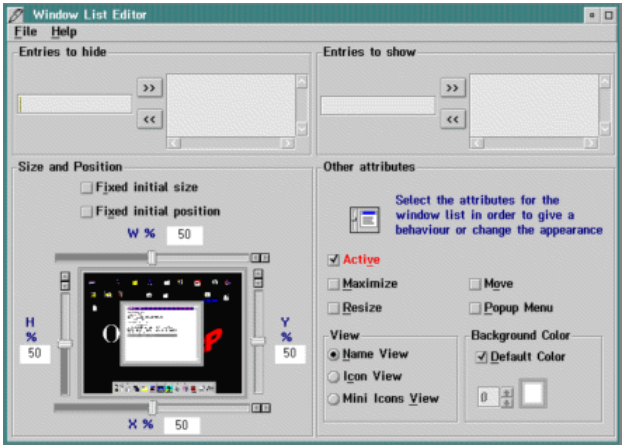
The Window List component

The windows list component allows you to configure the behavior of the window list application whenever a given security profile of its kind is active. Among other things you can specify which tasks are to be shown or hidden, whether to allow task management from itself, and some aesthetic configuration of its appearance.

In order to work with this component, please open the Window List Restrictions Workbench folder located in the SecureEntry Workbench folder.

In order to create a security profile for this component, drag one from the component profile template object (named EDYWDLST.INI).

To modify a profile of this type, you can drop it into the 'Windows List Restrictions Editor' program object, edit the settings you wish to, and then close it as if working with any regular editor :



In order to test a given Window List Restrictions profile, just drop it into the 'Test window List restrictions' program object. You can test it also from within the editor application through its menu options.

When configuring profiles for this component, keep in mind :

The Lists of processes to hide/show are used to decide whether a new entry in the switch list should be visible or not. The following rules are applied:

1. Wildcards '*' and '?' are allowed and the pattern matching is done case insensitive.
2. The matched string is in some instances not exactly what you see in the switch list container, but instead the real title of the application. This may be because, depending on the session type, OS/2 does preppend some session identification string to the title. You can use the supplied Switch list utilities : EDYSWL2 program to see the real string titles.
3. If the application title is not in both lists, the title is left untouched. (visible or invisible depending on the application itself).
4. If the application title matches an entry in the 'Entries to show' list, but not in the 'Entries to hide' list, then the application will be visible within the task list.
5. If the application title matches an entry in the 'Entries to hide' list, but not in the 'Entries to show' list, then the application will not be visible within the task list.
6. If the application title matches an entry in both lists, then the list to which the longest pattern belongs mandates what to do. So, for instance :

```
Entries to hide       Entries to show

     *                    *CMD*
                          *Desktop*
```

Will hide everything except what matches '*CMD*' or '*Desktop*', similarly

```
Entries to hide       Entries to show

  *CMD*                    *
  *Communic*
```

Would show everything in the task list, except those applications with switch list titles matching '*CMD*' or '*Communic*'.

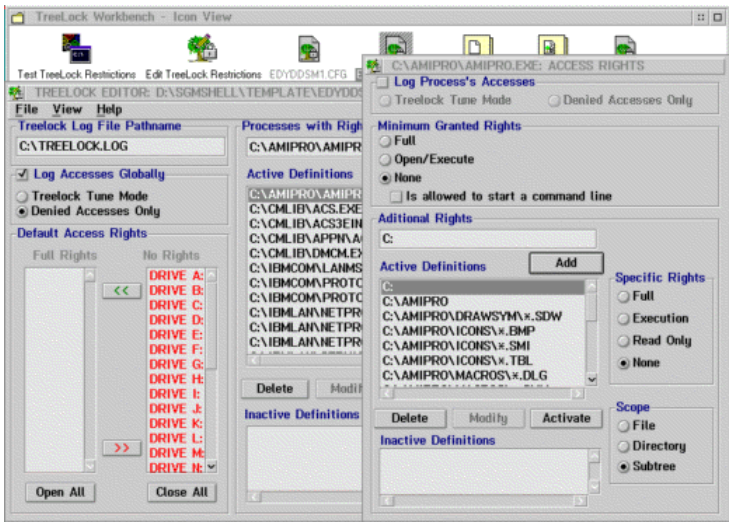
Due to the standard behavior of the task list, you should make sure that if it is active, then there is at least one item showable at all times, or the <ESC> key will not let you dismiss the dialog.

The 'Popup menu' checkbox refers to the ability to work with the context menus of the entries of the switch list. If the state is unchecked, then when using the profile, the task list will only be able to be used as a task switcher without the ability to move, close.... the affected windows.

The Treelock component

The treelock component allows you to control through an access control list profile, the parts of the logical drive and/or devices that the system should allow access to, once the component is activated with a given profile.

In order to work with this component, please open the Treelock Workbench folder located in the SecureEntry Workbench folder.



Within the treelock workbench folder, you will find :

Shadows to open the two sample profile files.

Templates to create treelock restriction profiles and audit files.

A program object to invoke the Treelock Editor by drag and drop in order to edit a given profile.

A program object to test a given treelock profile, named 'Test treelock'.

The Treelock Editor allows you to edit treelock profiles. A Treelock profile is basically an access rights list that specifies which file system resources will be accessible and which will not, once the profile is activated.

Note that the treelock audit file is treated as any other component, which you can assign to users or at a machine level through the users and groups administration tools.

Profile, overview

A Treelock Profile specifies:

Which file system resources will be accessible and which will not.

Which accesses to the file system will be logged and which will not.

Control over the file system is achieved through the definition of a default access rights section and the definition of an explicit access rights section, which is an access rights list structured in a process by process basis. **Once the profile is activated, each time a process attempts an access to the file system, the explicit access rights list is queried first and, if possible, applied, otherwise, the default access rights are used.**

Control over the log activity is achieved by means of a global definition and/or many process by process definitions called private log mode definitions. The global log mode acts as a default log mode that applies to all user processes that do not have a private log mode and, optionally, to all superuser processes. Either global o private definitions can specify to **log all the attempted accesses or only the denied ones.**

Note: a Treelock profile is in fact a text file that must fit a given syntax. So, a profile may have syntax errors. The Treelock Editor detects them at load time, and may display them at any time, giving the erroneous line number and an error message pointing to a possible cause.

Note: a process running in superuser mode is not subject to Treelock profile restrictions.

Profile, Superuser Processes

A process running in superuser mode is not related to any user and hence should not be restricted by the active Treelock profile (which is always related to a user session). To enforce this feature, every time a process attempts an access to a resource, the Treelock device driver finds out if the process is running in superuser or user mode. If the process turns out to be a superuser process then the Treelock device driver grants to it the access no matter what the active profile says about this process's access rights, otherwise the access is solved as defined by the profile.

So, superuser processes have always full rights on all file system resources, you can not change this, but you can configure the profile so as to log all accesses attempted by a superuser process. This allows you to monitor which processes are running in superuser mode in a given user session. See the Treelock Editor on-line help for more information.

Profile, Default Access Rights

A Treelock profile provides two ways to define the default access rights:

An explicit access rights definition related to the reserved process name: EXES.

A list of logical drives, from A to Z, that classifies each drive as accessible or not.

When a process P attempts an access to a resource R and its explicit rights are not applicable, then the EXES process explicit rights are queried, if possible they are applied, if not then the drive list is queried and the access is granted only if R is located in an accessible drive.

Profile, Explicit Access Rights

A **Treelock profile's explicit access rights section** is designed to define access rights in a process by process basis. It is a **list of access rights definitions**.

An **access rights definition** is an assignment of the same access rights to a particular set of processes.

The access rights are defined in two steps. First the **minimum granted rights** are specified and then the **specific access rights** are specified. The minimum granted rights, if defined, act as the default access rights for the set of processes, and are applied only if the specific rights can not be applied. The specific access rights cannot be more

restrictive than the minimum granted ones, and they are an access rights list structured in a resource by resource basis.

The set of processes affected by an access rights definition is specified by means of a complete file system pathname, all processes that match this pathname belong to this set of processes. **So, a process P has explicit access rights defined, only if P belongs to a set of processes specified by at least one of the profile's access rights definition (except EXES).**

Minimum Granted Rights

The minimum granted rights, if defined, act as default access rights for the set of processes, they apply whenever any of these processes attempts an access to a resource not specified in the access rights definition's specific access rights list. Can take the following values:

> **FULL** : the processes will have full access rights (open, execute, delete, move, and subdirectories creation and removal) **on all resources** and will be able to start a command line.
>
> **OPEN/EXECUTE** : the processes will have open and execute rights (but not delete, nor move, nor subdirectories creation and removal) **on all resources except those included in the specific access rights list**, and will be able to start a command line.
>
> **NONE** : the processes will have no minimum granted rights defined, and **all accesses to resources not included in the specific access rights list will be resolved using the logical drives default access rights list.**

Specific Access Rights

The specific access rights section of an access rights definition is designed to define access rights in a resource by resource basis, it is list of arguments, <**R**,**D**,**T**>, specifying what kind of access the processes have to what kind of resources:

R is a pathname to a file o directory.

> Processes will have access rights, **D**, on all resources that match this pathname.

D are the access rights.

> Processes will have access rights **D** on the resources defined by **R** These access rights can not be more restrictive than the minimum granted ones.

T is the resource type.

> It is used to modify the set of resources specified by **R**. May take three values:
>
> File
>
> > The set of resources is not modified.
>
> Directory
>
> > Processes have **D** access rights also on all resources located inside the directory specified by **R**.
>
> Subtree
>
> > Is the same as specifying the pathname **R***. Processes have **D** access rights also on all resources located in the subtree that hangs from **R**

Note: all pathnames that reference processes or resources in a profile can accept wildcards ('*' ,'?') , and may be parameterized to unrelate them from their file system's location.

Note: see the Treelock Editor's on-line help for more information about how to define access rights definitions.

Profile, Inactive Definitions

The Treelock Editor allows you to deactivate any access rights definition and any specific access rights definition.

An inactive definition acts as not defined when the profile is activated, however, the Treelock Editor recognizes it, and shows it inside the profile's inactive definitions section.

Profile, Parameterization

All pathnames that reference processes or resources in a profile may be parameterized to unrelate them from fixed file system's locations. After that, a profile becomes also a template and can be used to easily generate new profiles adapted to other file system configurations.

Parameterization is done by relating pathname's substrings to arbitrary parameter names.

```
c:\os2 could be related to %os_dir%
```

```
c:\sgmshell could be related to %sgm_shell%
```

Note: to avoid ambiguity, a parameterized string cannot be a complete substring of another parameterized string.

After being parameterized, a profile is still a profile, the difference is that it can also behave as a template, that is, it can be loaded as a template with the parameterized strings being substituted by their related parameter names, and you can assign new values to them to generate new profiles.

Note: see the Treelock Editor's on-line help for more information about how to parameterize a profile.

Profile, The ASCII syntax

The Treelock Editor detects a profile's syntax errors at load time. Once the profile is loaded, syntax errors can be displayed at any time: each erroneous line is showed, and for each error the line number and a an error message is given.

A profile with syntax errors is not trusty: the Treelock Editor does not save profiles with syntax errors, so it can be assumed that the profile has been edited as a text file and modified by hand.

What follows is the grammar that the ASCII profile must adjust to:

```
; This is a comment, and lasts until the end of a line

; GLOBAL parameters section
; -------------------------

[Set Debug=d:\fullpathname\filename]

; "set debug" establishes the name of the logging file. This file is
```

```
; recreated whenever the system is loaded or a profile is enabled and
; will contain all the actions the Device Driver logs. If not specified,
; no log file will be used, so nothing will be logged.

[Set Globdebug={a|d}]

; "set globdebug" directs the Device Driver to log ALL actions/accesses
; or only DENIED ones

;           ZYXWVUTSRQPONMLKJIHGFEDCBA
[Set Drives=00000011011011001001010111]

; "set drives" is a map of bits indicating whether access to a logical
; unit must be given (1) or not (0). This variable will be scanned when
; access to a resource could not be decided.

[Set Mode={s|c}]

; "set mode" establishes whether unspecified processes have no access
; rights (STRICT), so that all accesses will be denied; or full access
; rights (CALCULATED), so that all accesses will be allowed. The Device
; Driver will scan this variable only if no global variable "set drives"
; was present.
; NOTE : the Treelock Editor does not use this variable because profiles
; are always saved with DRIVES defined.
; NOTE : This line alone does NOT override the implicit  treelock
; restrictions (i.e, access to config.sys for regular users).

; PER PROCESS parameters section
; ------------------------------

[A|D] {F|O|Y|N} <executable|EXES|VDMS>  [args]*
[args] = [A|R|N] {F|D|S} <filepath>

; A = analyze flag for this line: all access performed by this process
;     will be logged, but not prohibited.
; D = debug flag for this line: all failing access performed by this
;     process will be logged.
; F = this <executable> has full access: open, execute, delete, move,
;     and subdirectory creation/removal.
; O = this <executable> has full open and execute rights, but no delete,
;     move,  mkdir nor rmdir.
; N = start of a command line will not be allowed for this  <executable>.
; Y = start of a command line will be allowed for this  <executable>.
;     This is the default value.
; <executable> = complete path and name of file, including drive letter
;                and colon (may contain the  wildcards '*' and '?').
; <VDMS>       = Real-Mode executables have the kind of access on data
;                files present in  <filepath>. If a data file is not
;                present there, Real-Mode executables will have the
;                access right depending on  EXES statement.
; <EXES>       = non specified executables (even Real-Mode ones) have
;                the kind of access on data files present in  <filepath>.

;                If a data file is not present there, non specified
;                executables will have the access right depending on
;                SET DRIVES value. If SET DRIVES variable is not specified,
;                non specified executables will have the access rights
;                depending on SET MODE value.
; For each process, there are zero or more  args possible. Each argument
; has two or three arguments:
; o Optionally the kind of access to a  <filepath>:
;   o A means that All accesses are allowed
```

```
;   o R means that Read-only accesses are allowed
;   o X means that Execute-only accesses are allowed
;   o N means that No accesses at all are allowed
;   A is the default value if none of the previous parameters is specified.
; o How to behave with  <filepath>:
;   o F means access to a File.
;   o D means access to all files in a Directory
;   o S means access to all files in a directory and its  Subdirectories
; o The <filepath>, which may contain the  wildcards '*' and '?'.
;
; You can break a line into more lines using the character '/'
```

Profile, Log Activity Configuration

From a Treelock profile you can configure the log activity you want to get done once the profile is activated. You can specify which processes do you want to monitor and which of their accesses do you want to log.

For a given process, there are two log modes that specify which accesses to log:

Log only denied accesses.

Log all attempted accesses (and grant them, no matter what the profile says)

A log mode may be globally or privately defined. A private log definition assigns a log mode to an access rights definition and takes effect on all **user** processes related to that definition; a global log definition assigns a log mode to all **user** processes that do not have a private log mode and, optionally, to all superuser processes.

Note: a private log definition does not apply to a process running in superuser mode. Superuser processes can only be monitorized by specifying it in the profile's global log mode definition.

Note: see the Treelock Editor's on-line help for more information about how to define log modes for processes.

Designing Treelock profiles

When assigning file access permissions, the system administrator must edit a treelock profile specifying what rights must be active. This profile will be saved as an ASCII file by the treelock editor and may be read by the Treelock device driver either when loading, or at run-time through the use of the Treelock device driver API.

You must be aware that the process of closing the system to unprivileged accesses consists of a fine tuning of the profiles using the method of "trial and error". For example, if you want to let the user execute the program Amipro/2, but restricting its access to some working directories, then you'll have to repeatedly test your profile until you get confident that:

1. You have identified all the resources that Amipro/2 internally accesses, giving it full access to those resources that it creates or updates, and giving it read-only access to those resources that it does not need to create nor update.
2. You have identified all the resources that Amipro/2 accesses as a result of the user accessing different directories/files. You must grant Amipro/2 a particular kind of access depending on what resources you want to protect/unprotect.

You can manage all the applications by giving specific access rights to some of these applications, and then by giving specific access rights to the rest of the non-specified applications (they all will have the same access rights).

Programs running in VDMs, even in the Windows subsystem, are indistinguishable one from the other, since the OS/2 Warp kernel only sees different instances of the Protected Shell process running in real mode. Therefore, all programs running in real mode will match the "joker" process **VDMS**. If the Treelock device driver cannot decide whether or not to give access depending on what the VDMS "joker" process restrictions state, it will eventually end up managing the process trying with the "joker" process EXES.

The process of closing the system heavily depends on your needs. You may need a relatively closed system where you specify restrictions on few processes and assign the rest of the process few restrictions; or you may need a more closed system, where you specify the restrictions you need for the majority of the processes, and let no open doors to non-specified processes. The first case leads to a relaxed, easy to configure, system profile; the second one will take more of your time to fine-tune it. To help you in this task **the Treelock Editor allows you to activate profiles and to edit log files in such a way that the displayed information may be directly used to modify the profile's data.**

Anyway, the best method to follow when approaching the tuning of a process in a profile is:

1. Specify a log file name for the profile.
2. Specify the global log mode as "log only denied accesses".
3. Specify all logical drives as accessible (open system).
4. For the process you want to tune specify the private log mode as "tune mode"
5. Give to the process you want to tune "NO" minimum granted rights.
6. Enable the profile and begin extensively using the program.
7. Edit the log file from the Treelock Editor and find out what resources are accessed by this process.
8. Add the internally used resources in the access rights definition of the process in the profile and set their specific access rights (you can do it directly from the log file editor)
9. Change the minimum granted rights's value to the one that best fit your needs.
10. Go to step 6 until you see that you have fine-tuned this program.

Testing Treelock profiles

The Treelock Editor allows you to activate and deactivate Treelock profiles without quitting it. Activating a profile means applying all its restrictions, so care must be taken when working with closed profiles.

To ensure normal operation the Treelock editor grants himself full rights on all profiles before activating them (it adds the the line "F <pathname>\tlcked.exe"). This is the only difference between a real activation and one done from the Treelock editor.

Besides, the active Treelock profile can only be deactivated by activating another Treelock profile, so, the Treelock editor deactivates it by activating the profile which was active before it. So, deactivating a profile from the Treelock editor is nothing but restoring the Treelock profile assigned at logon time to the current user.

See the Treelock editor's on-line help for more information about how to activate/deactivate a profile.

Profile, Sample 1

The configuration file EDYDDSM1.CFG sets no restrictions on accesses, writing them all to the file c:\treelock.log:

```
set debug=c:\treelock.log
set globdebug=a
set mode=c
```

Profile, Sample 2

The configuration file EDYDDSM2.CFG sets some restrictions on accesses, writing only denied accesses to the file c:\treelock.log. Unspecified processes will behave depending on what's specified in EXES; if the access specified in EXES fails it will fail, since mode is strict:

```
set debug=c:\treelock.log
set globdebug=d
set mode=s

; ====================
; Base OS/2 (required)
; ====================

; Give c:\os2\pmshell.exe open/execute rights to any resource in c:\spool,
; k:\post and h:\ . Give full access to c:\os2\os2.ini, c:\os2\os2.###,
; c:\os2\os2.!!!, c:\os2\os2sys.ini, c:\os2\os2sys.###, c:\os2\os2sys.!!!,
; c:\os2\pmdiary.###, c:\os2\pmdiary.!!!, c:\os2\pmdiary.ini,
; c:\os2\install\reinstal.ini, c:\os2\install\reinstal.### and
; c:\os2\install\reinstal.!!!

O c:\os2\pmshell.exe            s c:\spool          s k:\post     s H:   /
  f c:\os2\os2.ini              f c:\os2\os2.###    f c:\os2\os2.!!!     /
  f c:\os2\os2sys.ini           f c:\os2\os2sys.### f c:\os2\os2sys.!!!  /
  f C:\OS2\PMDIARY.###          f C:\OS2\PMDIARY.!!! f C:\OS2\PMDIARY.INI /
  f c:\os2\install\reinstal.ini f c:\os2\install\reinstal.###             /
  f c:\os2\install\reinstal.!!!

; Give c:\os2\harderr.exe, c:\os2\chkdsk.com and c:\os2\pmspool.exe
; full rights.

F c:\os2\system\harderr.exe
F c:\os2\chkdsk.com
F c:\os2\pmspool.exe

; ======================
; Base OS/2 (additional)
; ======================

; Don't allow c:\os2\apps\pmdcalc.exe to start a command line, giving it
; full access to c:\os2\apps\dll\pmdiary.dll, c:\os2\pmdiary.###,
; c:\os2\pmdiary.ini, c:\os2\pmdiary.!!!, c:\os2\help\pmdiary.hlp and
; c:\os2\help\hmhelp.hlp

n C:\OS2\APPS\PMDCALC.EXE  f C:\OS2\APPS\DLL\PMDIARY.DLL         /
  f C:\OS2\PMDIARY.### f C:\OS2\PMDIARY.INI f C:\OS2\PMDIARY.!!! /
  f C:\OS2\HELP\pmdiary.hlp f C:\OS2\HELP\HMHELP.HLP

; Don't allow c:\os2\apps\pmsticky.exe to start a command line, giving it
; full access to c:\os2\apps\dll\pmdiary.dll, h:\os2\pmdiary.###,
; h:\os2\pmdiary.ini, h:\os2\pmdiary.!!!, c:\os2\help\pmdiary.hlp,
; c:\os2\help\hmhelp.hlp and h:\pmsticky.$$p

n C:\OS2\APPS\PMSTICKY.EXE f C:\OS2\APPS\DLL\PMDIARY.DLL         /
  f H:\OS2\PMDIARY.### f H:\OS2\PMDIARY.INI f H:\OS2\PMDIARY.!!! /
  f C:\OS2\HELP\pmdiary.hlp f C:\OS2\HELP\HMHELP.HLP f H:\PMSTICKY.$$P
```

```
; Don't allow c:\os2\more.com to start a command line, giving it
; full access to c:\os2\system\oso001.msg and c:\oso0001.msg

N c:\os2\MORE.COM f C:\OS2\SYSTEM\OSO001.MSG f c:\oso0001.msg

; The following line would prevent cmd.exe from starting another cmd.exe,
; would give no access to program format.com, would give access to any
; other exe and com in c:, would give Read-Only access to treelock.log
; and full access to d: drive.
; With that line, exes and coms different from format.com in c: could be
; created and deleted, but no other files nor directories could be created
; nor deleted.
; To keep the same restrictions, but allowing the command prompt to start
; another command prompt, you must change the initial "n" to a "y"

;n c:\os2\cmd.exe n f c:\os2\format.com                    /
;  a s c:\*.exe a s c:\*.com r f c:\treelock.log a s d:

; The following line will will allow to start any program and open or create
; any file, even if you add some restrictions after cmd.exe. What you will
; not be allowed to do is erase nor move files, neither create nor delete
; nor move directories.

;o c:\os2\cmd.exe

; The following line gives full access to d: drive. Since nothing is stated
; about other drives, no access is given to them.

;n c:\os2\cmd.exe a s d:

; The following line makes cmd.exe to behave as the past line, but it will
; allow to start another command prompt.

;y c:\os2\cmd.exe a f c:\os2\cmd.exe a s d:

; The following line sets no restrictions on a command prompt:

f c:\os2\cmd.exe

; ==============
; VDMS statement
; ==============

; Give any Real-Mode program Read-Only access to files C:\CONFIG.SYS and
; C:\STARTUP.CMD, full access to C:\DOSUTILS and its subdirectories,
; full access to C:\OS2\MDOS\DOSKRNL and C:\OS2\BOOT\VIOTBL.DCP, etc.
; Note that No access will be given to any file/subdirectory in unit A:.
; Note that wildcards are accepted.
; Note that this statement manages IBM Anitvirus for DOS and DW4.

N VDMS R F C:\CONFIG.SYS              R F C:\STARTUP.CMD                      /
    S C:\DOSUTILS                     F C:\OS2\MDOS\DOSKRNL                   /
  A F C:\OS2\BOOT\VIOTBL.DCP          F C:\OS2\SYSTEM\COUNTRY.SYS             /
  A F C:\OS2\MDOS\COMMAND.COM         F C:\OS2\MDOS\APPEND.EXE                /
  A F C:\OS2\MDOS\WINOS2\WINOS2.COM   F C:\OS2\MDOS\WINOS2\SYSTEM\OS2K386.EXE /
  A F C:\OS2\MDOS\SYSTEM\*.DRV        F C:\OS2\MDOS\WINOS2\*.INI              /
  A F C:\OS2\MDOS\SYSTEM\*.FON                                               /
  A F C:\OS2\MDOS\WINOS2\*.GRP                                              /
  A F C:\OS2\MDOS\WINOS2\SYSTEM\SETUP.INF                                    /
  A D C:\OS2\MDOS                                                           /
  N S A:                                                                   /
  a s c:\ibmav                                                            /
```

```
  N F C:\DW4\*.doc                                                          /
  N F C:\DW4\*.rft                                                          /
  A F C:\DW4\*.LST                                                          /
  A F C:\DW4\*.PG1                                                          /
  A F C:\DW4\*.PRF                                                          /
  A F C:\DW4\DW4ODIR.BAT                                                    /
  A F C:\*.$$P                                                              /
  R S C:

; ==============
; EXES statement
; ==============

; Give to non-specified  EXES no access to a command line, plus no access
; to C:\CONFIG.SYS at all, Read-only access to  C:\STARTUP.CMD,  full access
; to directory  C:\TOOLS, etc.
; Note that  wildcards are accepted.

N EXES N F C:\CONFIG.SYS   R F C:\STARTUP.CMD         N F C:\AUTOEXEC.BAT  /
  A D C:\TOOLS            R F  C:\OS2\HELP\EPM.HLP       S C:\OS2\APPS     /
  A D C:\OS2\APPS\DLL        S C:\OS2\DLL                S C:\OS2UTILS     /
  N S A:                  N F  C:\OS2\MDOS\WINOS2\*.INI

; ==========
; Lan Server
; ==========

; And so on ...

F c:\os2\epw.exe
F C:\IBMLAN\NETPROG\LSDAEMON.EXE
F C:\IBMLAN\NETPROG\STOPLAN.EXE
F C:\IBMLAN\NETPROG\NET.EXE
F C:\IBMLAN\SERVICES\WKSTA.EXE
F C:\IBMLAN\SERVICES\MSRVINIT.EXE
F c:\IBMLAN\SERVICES\MSRV.EXE
N C:\MUGLIB\MUGLRQST.EXE  S C:\MUGLIB S C:\IBMLAN

;=====================================================================
; Communications Manager
;=====================================================================

F c:\ibmcom\lanmsgex.exe
F c:\ibmcom\protocol\netbind.exe
F c:\ibmcom\protocol\LANDLL.EXE
n c:\CMLIB\DMCM.EXE          S C:\CMLIB
n C:\CMLIB\ACS.EXE           S C:\CMLIB F C:\IBMLVL.### F C:\IBMLVL.INI F C:\IBMLVL.!!!
n C:\CMLIB\APPN\ACSAPLDR.EXE  S C:\CMLIB
N C:\CMLIB\ACS3EINI.EXE      S C:\CMLIB F C:\OS2\HELP\HMHELP.HLP

; ==========
; OS2 AmiPro
; ==========

N C:\AMIPRO\AMIPRO.EXE          A D  C:\AMIPRO                          /
A F C:\AMIPRO\ICONS\*.BMP                                              /
A F C:\AMIPRO\ICONS\*.TBL       A F  C:\AMIPRO\ICONS\*.SMI              /
A F C:\AMIPRO\MACROS\*.DLG      A F  C:\AMIPRO\MACROS\*.SMM             /
A F C:\AMIPRO\STYLES\*.STY      A F  C:\AMIPRO\DRAWSYM\*.SDW            /
A F C:\OS2\AMI*.*               A D  C:\OS2\DLL                         /
A F C:\OS2\*.BIN                A S  C:\SPOOL                           /
A F C:\OS2\HELP\HMHELP.HLP      A D  C:\PSFONTS                         /
A F C:\OS2\INSTALL\SYSLEVEL.OS2  A F  C:\OS2\SYSTEM\COUNTRY.SYS         /
```

```
A F C:\OS2\MDOS\WINOS2\*.INI     A F c:\amipro\teller\*.sam                /
N S C:

; ============
; OS2 IbmWorks
; ============

n c:\ibmworks\ibmworks.exe   a f c:\ibmworks\ibmworks.ini                 /
                            a f  c:\ibmworks\ibmworks.hlp                 /
                            a f  c:\os2\help\hmhelp.hlp                   /
                            a f  c:\os2\dll\*.drv                         /
                            a f  c:\os2\dll\*.dev                         /
                            a f  c:\os2\dll\*.prf                         /
                            a f  c:\ibmworks\*.def                        /
                            a f  c:\ibmworks\*.tmp                        /
                            a s  c:\delete                               /
                            a f  c:\ibmworks\*.ctn                        /
                            a s  c:\psfonts                              /
                            a s d:

N c:\ibmworks\fpwpim.exe     a f c:\ibmworks\ibmworks.ini                 /
                            a f  c:\ibmworks\fpwpim.*                     /
                            a f  c:\os2\help\hmhelp.hlp                   /
                            a f  c:\ibmworks\*.tmp                        /
                            a s  c:\delete                               /
                            a s  c:\ibmworks\DATA                         /
                            a s  c:\psfonts                              /
                            a s d:

N c:\ibmworks\fpwmon.exe     a f c:\ibmworks\ibmworks.ini                 /
                            a f  c:\ibmworks\fpwpim.*                     /
                            a f  c:\os2\help\hmhelp.hlp                   /
                            a f  c:\ibmworks\*.tmp                        /
                            a s  c:\delete                               /
                            a s  c:\ibmworks\DATA                         /
                            a s  c:\psfonts                              /
                            a s d:

; =============
; OS2 Lotus 123
; =============

n C:\LOTUS\123G\123G.EXE  S C:\LOTUS\123G S D:\DADES\123 D C:\PSFONTS     /
F C:\OS2\SYSTEM\OSO001.MSG

; ========
; FaxWorks
; ========

N c:\PBA\FXSVIEW.EXE   s c:\PBA f M:\FAXWORKS\FAX.LOG

N G:\FAXWORKS\FAXWORKS.EXE  S G:\FAXWORKS  S C:\FAXWORKS                  /
 F M:\FaxWorks\Fax.log  F M:\FaxWorks\User.Spl

; ======================
; Other OS2 applications
; ======================

n c:\os2\view.exe  a s c:\*.inf a f c:\os2\viewdoc.exe

N C:\OS2\VIEWDOC.EXE  a s c:\*.inf a s c:\*.cp S C:\OS2\HELP              /
  f C:\OS2\SYSTEM\COUNTRY.SYS   f C:\OS2\VIEWDOC.EXE
```

```
f m:\tools\inie.exe
f m:\tools\inimaint.exe
```

Ambiguity Resolution

A treelock profile may be ambiguous for one of two reasons:

A process exists that belongs to more than one access rights definition.

A resource exists that belongs to more than one definition in the specific access rights section of the same access rights definition.

The problem arises from the fact that two pathnames in the same profile may have a match intersection set not null. All processes or resources belonging to this set will be subject to ambiguity because more than one definition can be applied to solve their accesses.

The best solution would be to apply the definition with the most specific pathname, but due to the inherent complexity of that solution, the following approximation has been implemented: Whenever a process tries to access a resource, the treelock device driver begins to search sequentially all the definitions in the access rights definitions list. The first definition's pathname that matches the process pathname will be applied.

With this in mind, all we need to know is the sort method used by the treelock device driver to store the profile's access rights definitions.

The sorting is done on the access rights definition's pathname that specifies the set of processes related to the definition, and the sort algorithm is as follows:

Pathnames with no wildcards are stored first.

Pathnames with wildcards are stored after with the following sorting scheme:.

Longest pathnames are stored first.

For equal length pathnames a special alphabetical sorting is applied: x<?<* (where x is any other character different from '?' and '*')

Templates

A template is simply a Treelock profile that the Treelock Editor has loaded as a template. At load time the Treelock Editor substitutes all the profile's parameterized substrings with the parameter names related to them, and assigns default values to each of these parameters. Usually a parameter's default value value is the substring the parameter has substituted at load time; but if the parameter name is equal to a defined environment variable name at load time, then the default value assigned is not the environment variable's name but the environment variable's value. This allows to define lasting relations between parameters and environment variables.
Note: a profile not parameterized is still a template, but a template with no parameters

You can assign new values to the parameters to adapt the profile to many different file system configurations. Note that if a value assigned to a parameter is equal to a currently defined environment variable name, the value assigned is not the variable's name but the variable's value.

To generate profiles from templates you must
merge the template to a Treelock profile. The merge operation adds the template's access rights definitions to the profile with the parameters substituted by their current values but without modifying any other profile's data. Through this operation you can create new profiles (if the target profile is the empty profile) or easily update already existing profiles.

Note: see the Treelock Editor's on-line help for more information about how to load and work with templates.

The Log Debug File

The log debug file related to a Treelock profile, is useful when tuning restrictions for a specific profile. It has the following line format:

Result: it can be one of the following:

Ok: the access has been granted.

Nok: the access has not been granted.

Process Name: eight characters at most specifying the name of the process attempting the access.

Access Resolution Information: it can be one of the following:

The profile's access rights definition's pathname that matched the process (and was applied by the Treelock device driver) to solve the access.

SUPERUSER: the process was running in superuser mode (so the access was granted).

EMPTY: the profile's default rights were applied to solve the access.

Function Attempted: it can be one of the following:

DEL: a delete was attempted.

REN: a rename was attempted.

EXECP: a process creation was attempted.

OPEN: an open was attempted.

MKDIR: a create directory was attempted.

RMDIR: a remove directory was attempted.

CHDIR: a change of directory was attempted.

FINDF: a FindFirst was attempted.

FINDN: a FindNext was attempted.

FINDV: a Find from a DOS Box was attempted.

Flags: it can be a combination of the following and they are positional:

AN: analyze flag if it was specified for the process.

DB: debug flag if it was specified for the process.

BX: the process is running in Real-Mode.

DA: you must ignore this flag.

OE: open access.

NC: no command line starting allowed.

FA: full access.

NA: no access.

Full Filename: the full file name that was attempted to be accessed.

Take a look at the following example:

```
Nok CMD  C:\OS2\*     OPEN  '  DB          NA' A:\
Nok E                 OPEN  '  DB          NA'  C:\CONFIG.SYS
ok   E   SUPERUSER    EXECP '  DB          NC'  C:\OS2\CMD.EXE
```

The example states:

From a command line, an access to drive A: was denied, and the profile's "c:\os2*" access rights definition was used to solve the action.

From program E, an access to file C:\CONFIG.SYS was denied, and the profile's default rights were used to solve the access.

From program E, starting a command prompt was granted, and program was running in superuser mode.

The Treelock Editor allows you to edit any log file either as a text file or as log file editor file. As a text file the log file is displayed as it is, and cannot be modified; as a log file editor file, the log file contents are interpreted and structured so as to allow you to use the information to tune the currently loaded profile. The log file editor shows all this information indexing the logged accesses by the process that made them, and by whether they were granted or denied, and allows you to use the displayed processes and resources pathnames to directly add them to the currently loaded profile.

Note: see the Treelock Editor's on-line help for more information about how to edit log files.

The audit file

The audit file contains information of denied accesses only. This file contains a header and a tail with the denied actions:

Header:

Optional reserved word TLOCK_AUDIT: specifies that this file is an audit file.

Optional reserved word MAX_LINES: specifies that this audit file will purge a number of lines each time it's opened. If not specified, it will keep the most recent 1000 lines in the file, thus purging all previous lines.

Tail:

TimeStamp: audited date and time of denied access.

UserID: the userId that attempted the failed access.

Administrator: whether or not the user is Administrator.

Process Name: eight characters at most specifying the name of the process attempting the access.

Function Attempted: it can be one of the following:

DEL: a delete was attempted.

REN: a rename was attempted.

EXECP: a process creation was attempted.

OPEN R/O: an open for ReadOnly was attempted.

OPEN W/O: an open for WriteOnly was attempted.

OPEN R/W: an open for ReadWrite was attempted.

OPEN DLL: an open for a DLL was attempted.

MKDIR: a create directory was attempted.

RMDIR: a remove directory was attempted.

CHDIR: a change of directory was attempted.

FINDF: a FindFirst was attempted.

FINDN: a FindNext was attempted.

FINDV: a Find from a DOS Box was attempted.

Full Filename: the full file name that was attempted to be accessed.

Take a look at the following example:

```
MAX_LINES 100
07-08-1996 16:25:42 USER_A    No Admin E        OPEN R/W C:\CONFIG.SYS
07-08-1996 16:25:42 USER_A    No Admin TEDIT    OPEN R/O C:\CONFIG.SYS
07-08-1996 16:25:43 USER_A    No Admin CMD      FINDF    C:\CONFIG.SYS
```

The header states that this file will be purged each time it's opened, leaving only unpurged the most recent 100 lines at most.

The tail states that UserID USER_A, who is not an administrator UserID, was denied access to the file C:\CONFIG.SYS three times:

First, attempting to open for ReadWrite the file C:\CONFIG.SYS from the program E

Second, attempting to open for ReadOnly the file C:\CONFIG.SYS from the program TEDIT

Third, attempting to see whether or not the file C:\CONFIG.SYS exists from a command line (probably with a DIR command).

Usage Samples

This section shows how to build these three common treelock profiles:

1. **Hide a directory to a user**
2. **Restrict a user to a home directory**
3. **Give a user execute-only rights to an applications directory**

To **hide a directory to a user** you must:

1. Give access to all logical drives
2. Specify the EXES access rights definition with NO minimum granted rights
3. Add to the definition's specific rights section the directory you want to hide with:

   Access rights = none

   Resource Type = subtree

Let's see why this works out: any attempt to access a resource outside the hided directory is resolved using the logical drives access list because the EXES definition has no minimum granted rights defined and its specific access rights list does not include the accessed resource. As the logical drives list specifies all drives to be accessible the access is granted. However, if a process attempts to access the hided directory, the EXES definition can be applied to resolve the access, and as it specifies null rights on the resource, the access is denied.

Note: the resource type must be "subtree" to assure that all subdirectories will also be hided. We could also have specified the resource type to be "file" and the resource to be "dir_to_hide*"

To **restrict a user to a home directory** you must:

1. Give no access to all logical drives (except drive A if you want the user to be able to use the floppy)
2. Specify the EXES access rights definition with NO minimum granted rights
3. Add to the definition's specific rights section the home directory with:

   Access rights = full

   Resource Type = subtree

Once again, any attempt to access a resource outside the home directory is resolved using the logical drives access list because the EXES definition has no minimum granted rights defined and its specific access rights list does not include the accessed resource. As the logical drives list specifies all drives to be unaccessible the access is denied. However, if a process attempts to access a resource inside the home directory, the EXES definition can be applied to resolve the access, and as it specifies full rights on the resource, the access is granted.

Note: the resource type must be "subtree" to allow the user to work with subdirectories inside his home directory.

Give a user execute-only rights to an applications directory

Let's assume we want a closed system (drive c: is not accessible by default) and the applications directory name is "c:\exes". We want the user to be able to execute all applications located on that directory, but not to delete them. So, we must:

1. Set drive c: as not accessible by default.
2. Specify the EXES access rights definition with NO minimum granted rights.
3. Add "c:\exes" to the EXES's specific access rights section with:

   Access Rights=Execution

   Resource Type=Directory

   Now the user could execute any application located in the "c:\exes" directory and he could not delete them. That would be ok, but we must also tune each application the user is going to use. For instance, the

application "my_app.exe" could need to modify its INI file which could be located in the directory "c:\exes". By now it could not done it because "my_app.exe" has no explicit rights and the default ones give it execute-only rights to "c:\exes". So we should:

4. Add "c:\exes\my_app.exe" to the access rights definition list with NO minimum granted rights.
5. Add "c:\exes\my_app.ini" to the above definition's specific access rights list with:

   Access Rights=Full

   Resource Type=File.

The Treelock API

The Treelock API has three entry points, defined in EDYDDAPI.H:

Set an Access Control List, or reset it.

```
ULONG _System EDYPDDSetAccessToFiles(PSZ  FileName);
```

If FileName is NULL it will tell the Treelock device driver to check accesses depending on what is specified in the file %SGM_SHELL%\NOUSER\EDYDD32.INI; if this file does not exist, it will tell the device driver to not check any access. If FileName is not NULL, it will tell the treelock device driver to check accesses depending on what is specified in FileName; if this file does not exist, it will return an error.

Set an Audit File, or reset it.

```
ULONG _System EDYPDDSetAuditFile(PSZ  FileName);
```

If FileName is NULL it will tell the Treelock device driver to append denied accesses in the file %SGM_SHELL%\NOUSER\EDYDD32.AUD; if this file does not exist, it will tell the device driver to not audit any access. If FileName is not NULL, it will tell the device driver to append denied accesses to FileName; if this file does not exist, it will return an error.

Close the device driver log debug and audit files:

```
ULONG _System EDYPDDCloseLogDebugFile(VOID);
```

Successful operation returns the value 0 to the caller. A failure returns its value to the caller. Only 32-bit callers are supported.

The program EDYDDUTL.EXE is provided as a sample using the API API. Usage:

```
EDYDDUTL { {/F|/A}[drive:][pathname][filename][.ext]  | /C }
```

Where:

/Ffilename activates the access rights in filename.

/F disables the access rights checking.

/Afilename activates the auditing file filename.

/A disables the auditing file.

/C closes the log and audit files.

Implicit restrictions

In order to provide for an enough safe system, the treelock provides for the ultimate security restrictions which grant that a non-administrator user is not allowed to play with the different SecureEntry tools and interfaces, and thus jeopardize all of the integrity features that the product provides.

The restrictions are :

```
- Administrators will not have any restriction.
- Non-administrators will have the following restrictions:

  o No access to any resource located inside the paths given by the
    environment variable  RUNWORKPLACE except SES\PSSDMON.EXE,
    e.g. C:\OS2\SECURITY,  will be denied.

  o Any access to the following resources will be denied:

    o OS2\BOOT\SESDD32.SYS
    o OS2\BOOT\EDYDD32.SYS
    o OS2\BOOT\EDYFLPY.FLT

  o All files within the  SGM_SHELL environment variable path,
    DLL subdirectory will have Read Only and DLL permission.

  o All files within the  SGM_SHELL environment variable path,
    WORK and TEMP  subdirectories will have full access.

  o All files within the  SGM_SHELL environment variable path,
    TOOLS subdirectory will have read and execute permission.

  o The file EDYUTIL.EXE,  located within the  SGM_SHELL
    environment variable path, EXEC  subdirectory will have
    execute permission, as well as the  SecureEntry trace
    modules.

  o The following resources will have Read-only access:

    o EDYSTART.CMD
    o CONFIG.SYS
    o STARTUP.CMD
    o All files within the  SGM_SHELL environment variable path
      except those subject to previous less restrictive
      rules.
```

Note that you can override this restrictions by using a treelock profile which explicitly includes your preferences, although you are encouraged not to do so. For instance, supposing you want to override the restriction of not letting regular users modify the config.sys file, you could use the following treelock profile :

```
set mode=c
Y EXES A F C:\CONFIG.SYS
```

Which means : Let unspecified processes do all type of operations with config.sys, plus opening command lines.

The startup process

The regular OS/2 startup process is not secure because it can be interrupted at any time, and it could leave the workstation in a non stable state. The objective of the Secure Startup process is to provide a secure startup sequence. You should do this by placing the contents of what you would have put in the 'STARTUP.CMD' file inside a file called 'EDYSTART.CMD' within the boot drive's root directory. The startup process component has some commands which you can use to give feedback to the user while processing this startup file.

The Secure Startup process cannot be interrupted and it is a system modal application which it gives to the user information about the different programs that are being started. The application has also a slider which informs of the percent value completed of the startup process and a multiline entry field used to give messages of the process being started at any time.

A log file named 'EDYLKINI.LOG', located on the same directory of 'EDYSTART.CMD' is created each time the Secure Startup application is launched. It contains the messages displayed on the multiline entry field of the main window. It also contains information of any error produced during the execution of the Secure Startup process.

EDYLKINI.EXE

EDYLKSLD.EXE

EDYLKMSG.EXE

EDYLKBLK.EXE

EDYLKSWT.EXE

EDYWFWPS.EXE

Sample startup process

EDYLKINI.EXE

Description

This file is to be used in testing period in order to test the startup sequence.

Syntax

```
EDYLKINI [/NOMODAL]
```

where /NOMODAL means the startup process won't be system modal.

Return codes

None.

EDYLKSLD.EXE

Description

Executable used for moving the slider of the secure startup dialog.

Syntax

```
EDYLKSLD [number]
```

where number is the slider's position. If omitted, it slides to the next position.

Return codes

None

EDYLKMSG.EXE

Description

Executable used to show an informational message in the multiline entry field of the secure startup window.

Syntax

```
EDYLKMSG message
```

where message is the message to be displayed on the multiline entry field (MLE) of the secure startup process.

Return codes

None

EDYLKBLK.EXE

Description

Executable used to lockup the workstation. It can be used if some error occurred.

Syntax

```
EDYLKBLK message
```

where message is the last message to be displayed on the multiline entry field (MLE) of the secure startup window.

Return codes

None.

EDYLKSWT.EXE

Description

Stores the name of the process to be launched after the startup command is finished.

Syntax

```
EDYLKSWT parameters
```

```
parameters:
-D:program working directory
-P:program.exe
-A:program arguments
-T:session title
-U:x program startup time in seconds, default 0 seconds
-F:x foreground/background   0=foreground (default)
                                1=background
-S:x  session type  1=OS/2 full screen (default)
                          2=OS/2 Windowed
                          3=OS/2 PM
                          4=VDM (DOS or WINOS2)
                          7=Windowed VDM
-X:x window position X
-Y:x window position Y
-W:x window width
-H:x window height
```

Return codes

None.

EDYWFWPS.EXE

This command allows you to wait for completion of the WorkPlace Shell initialization process, and is designed to be used **exclusively** during EDYSTART.CMD processing. Once executed, it will not return control until it is safe to use the WorkPlace Shell. It has no optional parameters, so the calling syntax is :

```
EDYWFWPS
```

Note that you can only use this utility if the environment variable **RESTARTUSERSHELL** is set to *NO* (default value). Otherwise, the startup code would never complete.

This executable resides in the SecureEntry installation path, EXEC directory.

Sample startup process

This is an example of a secure startup cmd. It should be named 'EDYSTART.CMD' and placed in the boot drive's root directory.

```
rem move the slider to the first position
edylksld
rem show a message
edylkmsg "Starting Server services"
rem move the slider to the third position
edylksld 3
rem starting net
net start requester
edylksld
rem show a message
edylkmsg "Start communications manager"
rem start sessions
cmstart
edylksld 6
edylkmsg "Starting background processes."
start myserver.exe
```

```
edylksld 8
edylkmsg "Starting daemon"
rem  Next instruction will start our own daemon program  whensecure startup cmd ends but
rem  will remain the secure startup window for 10 seconds.
edylkswt -P:C:\MYAPPS\MYDAEMON.EXE  -S:3 -D:C:\MYAPPS -U:10
rem move slider
edylksld 9
rem assign resources
net use J: MyAlias
rem Slider : 100% completed
edylksld
rem  end the secure startup process. After executed, all the programs specified with the
rem  edylkswt instruction will be launched.
rem  exit the cmd
@exit
```

Boot protection

The SecureEntry Boot Protection tools allow you to protect the system from attempts to boot with uncontrolled operating systems through diskettes. There are basically three ways to accomplish this :

1. By using directly the machine system menus to define the startup sequence of the machine not to look for diskette drives at startup and define an administrator password to enter the system menus. This option does not require SecureEntry help at all, but unfortunately not all available hardware supports this features.

2. By changing the startup sequence through the machine BIOS, so that the diskette drive is not 'seen' by the system at startup. This is the safest one but can only be used with PS/2 models that have system programs on a hard disk partition. This is called the BIOS boot control.

3. By protecting the master boot record of the machine so that 'alien' operating systems can not see the protected partition. This is the only left solution for machines which can not benefit from any of the previous two methods. This is called software boot protection.

You have to decide which of the three methods to use for your hardware.

SecureEntry BIOS Boot control

SecureEntry Software boot protection

By last, and although not directly related to boot protection, note that you can use the Master boot record saver provided utility, if what you are looking for is also a means for preventing misbehaved programs from overwriting your system disks master boot records.

SecureEntry BIOS Boot control

Description

The BIOS boot control feature prevents end users from booting the system from a diskette drive and accessing the system setup menus. This feature applies **only** to the PS/2 models that have system programs on a hard disk partition.

This security feature is implemented by changing the startup sequence through the system setup menus and controlling user access to the setup menus that reside on the system partition. The following should be fulfilled:

Obtaining the *installation diskettes* to be used on the production workstations to install the boot control feature.

Setting up the production workstations. The startup sequence is changed and the boot control feature is installed on each workstation.

Generating *password diskettes* to be used at the production sites by system administrators. When a password is entered by the user and matches the password on the diskette, the user can access the setup menus.

Obtaining installation diskettes

The installation diskettes for the boot control feature are obtained from a reference diskette.

To generate reference diskettes, you must perform the following on a workstation that has system programs on a hard disk partition:

Access the setup menus in one of the following ways, depending on the system, during system startup:

Pressing Ctrl-Alt-Ins when the cursor appears on the top right screen corner.

Pressing the required function key, usually F1.

Get a copy of the system programs. Choose the following sequence of options from the setup menus:

Backup/Restore system programs

Backup the System Partition.

To get a copy of the system programs, you will need one or two diskettes. The first one is the reference diskette, which will be used to obtain the installation diskette. The second one is the diagnostics diskette, which is not required for setting up the production workstations.

To generate installation diskettes, you must perform the following:

Ensure that the files EDYSREFD.EXE and EDYCHKPW.EXE are located on the same directory, either on a hard disk or on a diskette. They can be obtained from the SecureEntryPath\TOOLS path on the workstation.

From the directory where EDYSREFD.EXE and EDYCHKPW.EXE are located, run EDYSREFD. This utility requests you to insert the reference diskette into the diskette drive.

Upon completion, the contents of the diskette are modified and it becomes the installation diskette.

Setting up the production workstations

To modify the startup boot sequence and prevent booting from diskette drive, perform the following on every production workstation:

Access the setup menus.

Choose the following sequence of options from the setup menus:

Set features

Set startup sequence.

Select the hard disk as the primary startup device and the diskette drive as the secondary startup device.

Save the new startup boot sequence.

To install the boot control feature, insert the installation diskette into the diskette drive and restart the system.

Upon completion, the boot control feature is operative. To access the system setup menus, a password diskette is necessary.

Generating password diskettes

To generate password diskettes, you must perform the following:

Ensure that the file EDYSETPW.EXE is available. It can be obtained from the SecureEntryPath\TOOLS path on the workstation.

Insert an empty diskette into the diskette drive.

From the directory where EDYSETPW.EXE is located, run EDYSETPW.

Enter a password, which can be up to 8-characters long.

Enter the same password a second time for verification.

Upon completion, the diskette contains the password in encrypted mode. It becomes the password diskette and can be used to access the system setup menus on those workstations where the boot control feature is operative.

Accessing system setup menus

To access the system setup menus on those workstations where the boot control feature is operative, a password diskette is necessary.

System administrators can provide end users with password diskettes so that they can access the system setup menus as follows:

Insert the password diskette in a diskette drive.

Start up the system.

Access the setup menus.

Enter the password corresponding to the password diskette, when prompted. The maximum number of attempts is three.

If the password is correct, the user can choose one of the following options:

1. Access setup menus

   The setup menus appear on the screen. The user can change any settings, even the startup sequence, which should be reestablished for security reasons.

2. Change password

   The user must provide first a new password for the password diskette and then the same password for verification. If the passwords match, the password on the password diskette changes.

3. Restricted operation

   This option displays the following menu choices:

1. Return to the previous menu
2. Deactivate protection process

   This option should be restricted to system administrators. If selected, the boot control feature becomes not operative.

SecureEntry Software Boot protection

Description and procedures

The SecureEntry software Boot Protection component makes it impossible to access partitions in the hard disk drives in a machine if the system has been booted from a floppy drive. Although booting from diskette can be carried out, the system will not be able to access the data in the partitions of the hard disk drives.

Note that you can directly install this component through the program icons located within the SecureEntry installation tools folder.

To enable the Boot Protection component run the following program from an OS/2 session:

```
EDYNOTA.EXE /Ppassword
```

password

This is your own keyword that enables the Boot Protection component

Then, add a line in your CONFIG.SYS file calling EDYNOTA.EXE program. For example, if SGM_SHELL=E:\SGM then add the following line:

```
CALL=E:\SGM\EXEC\EDYNOTA.EXE  /R
```

At this point, the hard disks will be prevented to be accessed if the system is booted from a diskette.

To inhibit the Boot Protection component, you must REM (or delete) the line that you previously added to the CONFIG.SYS file, and then run the following program from an OS/2 session:

```
EDYYESA.EXE /Ppassword
```

password

This is the keyword that you previously used to enable the Software Boot Protection component.

Once the Boot Protection component is installed, using EDYNOTA.EXE with a password different from the keyword used when the Boot Protection component was installed will not make any change to the Boot Protection component. Similarly, the Boot Protection component will not be deinstalled if EDYYESA.EXE is called with a password different from the keyword used when the Boot Protection component was installed.

Messages

Notice that having the Boot Protection component enabled produces some side effects: any program you run that reads the master boot record., e.g. FDISK, will show you meaningless data since all partitions are marked as unused.

The following messages may appear:

EDYNOTA.EXE (run-time)

Successful operation (even with a warning):

```
Ok
```

Warning:

```
Boot protection already installed on disk X - Password ignored
Password ignored
Boot protection not installed on disk X
Boot protection already installed on disk X
```

Error:

```
Error produced on WEEKDAY, DATE AND TIME
----------------------------------------
 Source Module Name: SOURCE MODULE NAME
 Compilation Date  : COMPILATION DATE
 Compilation Time  : COMPILATION TIME
 Source Line Number: SOURCE LINE NUMBER
 Logged Error      :  (hex)=RC (dec)=RC
```

In this case, this error message will also be output to an ASCII file named EDYLOGA.ERR.

EDYYESA.EXE (run-time)

Successful operation (even with a warning):

```
Ok
```

Warning:

```
Boot protection not installed on disk X
Invalid password
```

Error: the same type of message shown with EDYNOTA.EXE.

EDYBOOT.SYS (boot-time)

Successful operation: none.

Warning: none.

Error:

```
Cannot read partition
Invalid partition
```

Compatibility with OS/2 standalone dump facility

Note that if you use the standard OS/2 feature **TRAPDUMP** to initiate a system dump of the machine memory to a hard disk (SADUMP partition) automatically whenever a system fatal fault is produced, then if SecureEntry

boot protection is installed, the dump partition may not be recognized by the dump facility program. To overcome this situation, either :

Remove SecureEntry boot protection before reproducing the situation that causes the memory dump to be initiated, or

Use the supplied command *EDYPDUMP.CMD* to patch the OS/2 dump facility code so that it will recognize SecureEntry boot protected partitions. This program is located within your SecureEntry path, *EXEC* directory.. You can invoke it without parameters to read a usage description. Basically, running :

```
EDYPDUMP SET
```

will patch the dump code, and

```
EDYPDUMP RESET
```

will unpatch it. If you intend to use this approach in production machines, you should **TEST** it with your hardware before implementing it. There is a small risk that this technique is not compatible with certain types of SCSI adapters.

Return Codes

At run-time, successful operation or warning returns the value 0 to the command line. A failure returns 1 to the command line.

At boot-time, successful operation lets the system to continue the boot process. A failure makes the boot strap to enter an endless loop.

File integrity check

The SecureEntry file integrity check component provides a means of validate the integrity of all static data and program files at either start-up time or prior to the use of the files. A variety of integrity-checking schemes is offered, including fast and effective checking for virus changes and thorough cryptographic validation.

The SecureEntry file integrity check component configuration table is used to specify which files in the system will be checked for integrity by comparing checksums during startup. These checksums must be computed beforehand using an utility.

For a particular file, you can select one of two possible checksum algorithms:

XOR the 64th bit

MD4 Message Digest Algorithm

XORing the 64th bit is fastest than applying MD4, but MD4 is safest.

Components description

Components

To calculate a file checksum:

```
SAF2INGN.EXE <Method> <FileName>

where: <Method> is 1 for XORing the 64th bit, or 3 for  MD4
       <FileName> is the name of the file to calculate the checksum
```

Messages:

Successful operation:

```
Calculated checksum:
```

Error:

```
Usage: SAF2INGN Method Filename
Invalid method
Open file error
Invalid Checksum
```

Return Codes: Successful operation returns the value 0 to the command line. A failure returns the corresponding Return Code to the command line.

To verify a series of files:

```
SAF2INCK.EXE <ParmFile> <CheckID> <OutFile> [<OptProgram>]

where: <ParmFile> is the name of the file with filenames to check
       <CheckID> is the line(s) with ID in file to be checked
       <OutFile> is the output results file, which is  stdout if
                 it cannot be opened
       <OptProgram> is an optional program that is started on
       successful check
```

The <ParmFile> must adjust to the following syntax:

```
<CheckID> <CheckSumMethod> <CheckSum> <FileName>
    |           |             |          |
    |           |             |          +-> Name of the file to be checked
    |           |             +----------> Hexadecimal  CheckSum
    |           +--------------------------> 00001:  XOR the 64th bit
    |                                        00003:  MD4
    +-------------------------------------->  CheckID for current line
```

In the following example, the files C:CONFIG.SYS and C:AUTOEXEC.BAT will be checked for checksums of 77 and 40, respectively, XORing the 64th bit if SAF2INCK.EXE is invoked with a CheckID of 10000; and the files C:OS2\SYSTEM\CONFIG.DOS and C:OS2\SYSTEM\AUTOEXEC.DOS will be checked for checksums of 7F89AA02751EA8908348C4C38D9D390A and 680A5CA3E0F9B8A5372B39EB856CE268, respectively, using MD4 if SAF2INCK.EXE is invoked with a CheckID of 1000A.

```
10000 00001 77                                 C:CONFIG.SYS
10000 00001 40                                 C:AUTOEXEC.BAT
1000A 00003 7F89AA02751EA8908348C4C38D9D390A  C:OS2\SYSTEM\CONFIG.DOS
1000A 00003 680A5CA3E0F9B8A5372B39EB856CE268  C:OS2\SYSTEM\AUTOEXEC.DOS
* Comments can be inserted with a  '*' at the beginning of a line
```

Messages:

Successful operation:

```
Check OK !!!!
```

Error:

```
Wrong usage
Unable to open configuration file
Error(s) found during check
Invalid method
Open file error
Invalid Checksum
Unknown checkid
System stopped
```

Return Codes: Successful operation returns the value 0 to the command line. A failure returns the corresponding Return Code to the command line.

To calculate a series of files checksums:

```
SAF2GEN.CMD <CheckID> <FileSpec[+Filespec..]> <Method> [/S] [/H] [/R] [/T]

where: <CheckID>  is the check id to generate on output for each processed file
       <Method>   is 1 for XORing the 64th bit, or 3 for  MD4
       <FileSpec> are the names of the files to calculate checksum for, with
                  wildcards allowed
       /S         optionally does recursive searching in the included directories
       /H         include also hidden files
       /R         include also  readonly files
       /T         include also system files
```

On output, this command generates the expected output for running SAF2INCK, so you can redirect it to a file and use it as a <ParmFile>. Any error found will be output with a beginning '*' character, so that SAF2INCK will interpret it as a comment (i.e, when the checksum could not be calculated because, for instance, another process is using the file).

Return Codes: Successful operation returns the value 0 to the command line. If the checksum for all the intended files could not be calculated, 1 is returned. Syntax errors in the calling command cause 2 to be returned.

In the following sample, all files in C:\OS2\DLL and C:\CMLIB\DLL are having their checksum calculated, and appended to a param file where we calculate also the checksums for all the tree in C:\MYDIR with id 00001 and method 3. The output parameter file will be PARMFILE.DAT.

```
SAF2GEN 1 C:\MYDIR\* 3 /S >PARMFILE.DAT
SAF2GEN 1 C:\OS2\DLL\*+C:\CMLIB\DLL\*  3 /S >>PARMFILE.DAT

Note that checkid will be left padded with 0's. in the output.
```

Processes auditor component

The SecureEntry Processes Auditor Component allows you to get answers to questions such as:

Are there some processes hogging the CPU? Which?

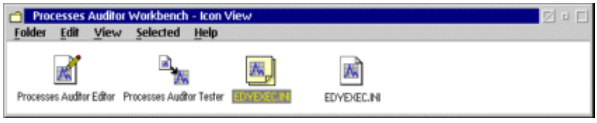
Are the users actually using the tools the Corporation I work for has invested on? Which of them have more acceptance?

Are there users who have a real CPU shortage in their systems?

Are there some users wasting their time when performing iterative tasks?

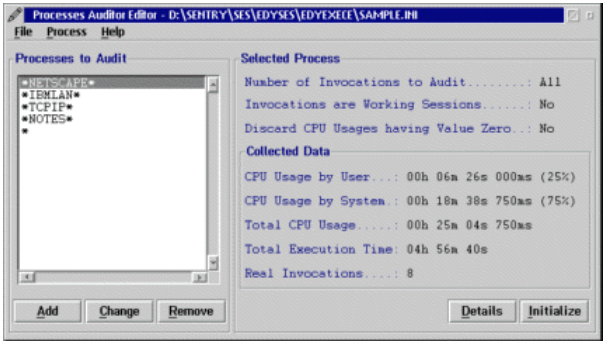
Are explicitly forbidden programs used in the Corporation's systems?

In order to work with this component, open the *Processes Auditor Workbench*, located inside the *SecureEntry Workbench*.



You can create a profile for this component by dragging a copy from its component profile template (*EDYEXEC.INI*) and dropping it.

In order to modify a processes auditor profile, drag and drop it into the *Processes Auditor Editor*, or just double click it. Make the necessary changes and then save the modified profile. Follow the editor's help for further advice.

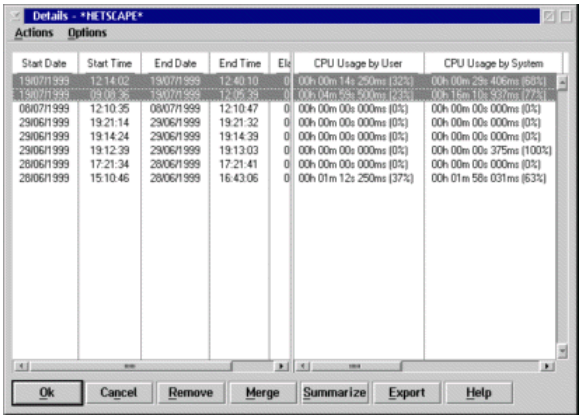


In order to verify the profile behavior, drag and drop it on the object *Processes Auditor Tester*, or select the option test through the editor menu bar. The defined profile will be activated.

Once configured, associate the profile to the desired user or group, by dragging and dropping it into the adequate container of the users and groups administration tool, or establish it as the default for the system, copying it with its default name (EDYEXEC.INI) to the *NOUSER* folder.

When a user logs on the system, its related profile will be activated, yet his or his SecureEntry group's if his does not exist. If the user does not have any defined processes auditor profile, then the profile residing in the *NOUSER* folder will be activated. If the *NOUSER* folder does not contain any processes auditor profile, then no processes will be audited.

Since this is one of the SecureEntry Components that feedbacks the profile with output data, a profile is not only downloaded at user logon time, but it is also uploaded at user logoff time. You can use the Processes Auditor Editor to display SecureEntry detailed collected data.



The Editor also provides additional extensions for exporting to or importing from an external file information about detailed collected data.

Refer to Error codes and messages for a description of the error codes the Processes Auditor Editor can return.

The collected information

Description of the modules and the API

Auditable Processes Dumper Tool: EDYEXEDM

The collected information

Output **Collected Data** by the SecureEntry Processes Auditor component is made up of:

CPU Usage by User By definition, this is the accumulated time when a process is running at Ring 2 or Ring 3. In practice, you can think of it as the amount of time a process has been executing, probably as a consequence of the *action of the user*, using the resources **owned** by the application.

CPU Usage by System By definition, this is the accumulated time when a process is running at Ring 0. In practice, you can think of it as the amount of time a process has been executing using **system** resources and code, i.e., at device driver and kernel level.

Total CPU Usage This is the sum of both User and System usage times, and it measures the amount of time a specific process has been running at task time.

Total Execution Time This measures the elapsed amount of time between the process has started and its termination.

Real Invocations This is the number of times a different instance of the selected process has been found running.

As a rule of thumb, since the total CPU usage reflects the amount of time a process has been running at task time, subtracting this amount from the elapsed time gets the value for interrupt and disabled time processing.

The system clock granularity, which is 31.25 ms per timer tick, makes CPU User and System times rather useless for anything but gross measurement, so beware you can get meaningless information if you audit your profile during a relatively short period of time.

Description of the modules and the API

What follows is the description of the modules integrated in this component:

EDYEXECE

This is the program editor of profiles. It accepts the following syntax:

```
EDYEXECE.EXE [Profile]
```

Where *Profile* is the path and file name of the desired profile. The editor allows for testing a profile directly.

EDYEXECT

This is the program activator of profiles. It accepts the following syntax:

```
EDYEXECT.EXE /F[Profile]
```

/F*Profile* activates the profile defined in *Profile*. If *Profile* does not exist, no processes are audited.

/F activates the profile EDYEXEC.INI defined in the *NOUSER*. If the profile does not exist, no processes are audited.

Processes auditor component interface

The component exports the following routine, defined in *EDYEXECD.H*. The routine executes correctly if it returns zero; otherwise it returns the gotten error:

```
APIRET _System ActivateExecsAuditor(PSZ  pszFileName);
```

Activates the profile defined in pszFileName. If pszFileName contents is empty, then no processes are audited.

If pszFileName points to "", then NOUSER ini file will be handled.

If the file does not exist, then an error is returned and no processes are audited.

Auditable Processes Dumper Tool: EDYEXEDM

The *EDYEXEDM* tool can be used to extract and/or to reset in batch mode the collected information by the processes auditor component. The extracted information will be stored in a text file with the same format as the one used by the *Processes Auditor Editor* to import and export the detailed collected data.

The invocation syntax for this command is as follows:

```
EDYEXEDM ? | /Help | [/Outfile:f] [/Users:{mask[+mask...]|@file.lst}]
         [/Groups:{mask[+mask...]|@file.lst}]
         [/Process:{process[+process...]|@file.lst}]
         [/Cpu:{U|S}ratio] [/Reset | /ONLYReset]
         [/Info:{[U|G] [M|S] [D|D[..]date{[..]|[..date...]}|D@file.lst]
                 [OCU|OCS|OCT]}]


    /Help and ?  : Usage.
    /Outfile     : Output file path and name. Defaults to  EDYEXEDM.TXT.
    /Users       : Only extract the information of the processes belonging
                   to the users matching the specified masks.
    /Groups      : Only extract the information of the processes belonging
                   to the users of the groups matching the
                   specified masks.
    /Process     : Only extract the information of the specified processes.
    /Cpu         : Only store the extracted information of the processes
                   with:
                   U  a  Cpu usage by user ratio larger or equal
                      than the  specidied ratio (0-100).
                   S  a  Cpu usage by system ratio larger or equal
                      than the specified ratio (0-100).
    /Reset       : Also reset in the profile the extracted information
                   of the processes.
    /ONLYReset   : Reset and do not extract the information of
                   the processes.
    /Info        : Determines how is stored in the output file the extracted
                   information. At least one of the next values
                   must be specified:
                   U  join the information of the processes of all the users.
                   G  join the information of the processes of all the users
                      belonging to the same group.
                   M  merge all the invocations belonging to the same process.
                   S  summarize all the invocations of the same process.
                   D  merge the invocations belonging to the same process that
                      start the same day.
                    D[..]date{[..]|[..date...]}  and D@file.list
                         merge the invocations belonging to the same process
                         that start in the same interval of time.
                         The border interval date have the  "dd-mm-yyyy" format.
                    OCU  order the invocations by  cpu usage by user.
                    OCS  order the invocations by  cpu usage by system.
                    OCT  order the invocations by the  cpu usage total.
                       (by default the invocations are ordered by start date)
```

So for instance: `EDYEXEDM /U:EDYADMIN+U* /G:@GROUP.LST /C:U30 /INFO:G S OCT`

Extract the process information for the *EDYADMIN* user, the users matching with *U** and the users of the groups matching the masks specified in the *GROUP.LST* file. Also, only the processes with a cpu usage by user ratio larger than *30* will be considered. The information extracted will be stored by groups, summarizing the invocations and sorting it by total cpu usage. `EDYEXEDM /U:* /R /INFO:D..12-09-1999..12-10-1999..`

Extract and reset the process information for all the users. The information is stored merging all the invocations that start before *12-09-1999* in one line. Between *12-09-1999* and *12-10-1999* in another line, and after *12-10-1999* as the third summary line. `EDYEXEDM /U:@USER.LST /P:* /ONLYR`

Reset the information of the process * for the users matching the masks specified in the USER.LST file.

Note that the *M* and *S* values of the */INFO* parameter perform exactly the same task as the *Merge* and *Summarize* commands of the *Processes Auditor Editor*.

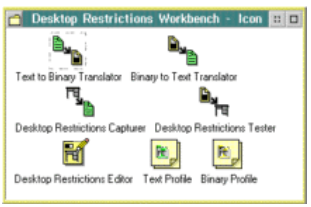
This module resides in the SecureEntryPath, *EXEC* directory.

Security components - WorkPlace shell

The Desktop restrictions component

The Desktop restrictions component allows you to place restrictions on the machine's desktop once a given profile is activated. You can, for instance set some objects invisible or disallow dragging of them, or completely/partially suppress the popup menu for the desired objects.

In order to work with this component, please open the Desktop restrictions Workbench folder located in the SecureEntry Workbench folder :



The Restrictions profiles come in two different fashions : Text profiles and binary profiles. You can only assign to users and groups the binary profiles.

You can create a text profile for this component by dragging a copy from its component profile template (name is Text Profile).

You can create a binary profile for this component by dragging a copy from its component profile template (name is Binary Profile).

The tools that the desktop restrictions workbench provide are :

A desktop capturer, to capture the contents of your desktop (the objects and their current settings). Capture a desktop by dropping a text profile into the 'Desktop Restrictions Capturer' program object.

Both 'compilers', from binary to text and from text to binary, in order to convert one profile to the contrary format. Do this by dropping the given profile into the appropriate compiler program object (names are 'Binary to Text translator' and 'Text to binary translator').

The 'Desktop restrictions tester' program object, to be able to test the effects on the desktop of a given profile. Just drop a given binary profile into this object to test it.

The 'Desktop restrictions editor' program object, which allows you to set the desktop into edition mode, and be able to edit the restrictions of a given binary profile. Use it by dropping into it a given binary profile. The edit window will open and you can then edit any desktop object restrictions for that profile by opening its restrictions notebook by clicking into the 'Edit restrictions' menu item of its pop-up menu.

The direct editing feature : allows you to drop a given object into a binary profile and edit then directly the object's restrictions for that profile.

```
**************************************************************************
* Note : Most of the processes explained underneath are only maintained *
*        for compatibility reasons. Specifically, working your desktop  *
*        restrictions  through text desktop profiles is no longer  the  *
*        suggested way to do it,  since  working directly  with binary  *
*        profiles is  a  far better and more  efficient way to achieve  *
*        the same result.                                               *
**************************************************************************
```

Configuring desktop profiles starting with a text desktop profile

Testing desktop profiles

Modifying desktop profiles

Text desktop profile format

Managing desktop objects position

Workplace shell personalization

Desktop profile activation API

Commands syntax

Configuring desktop profiles starting with a text desktop profile

To configure a desktop profile, that is, to produce a binary file with the desktop profile information, you can either start with a binary profile and use the desktop profile editor to provide it with your desired restrictions, or start with a text desktop profile. In this later case, go through the following steps:

1. Editing the desktop profile
2. Compiling the desktop profile

Editing the text desktop profile

To edit and obtain a text desktop profile, go through the following steps:

1. Create an instance of the Text desktop profile object. Specify as object title the name of the file where the desktop profile configuration data will be stored.

   You can also use a text file corresponding to a desktop profile already configured. In this case, you may get the text desktop profile as follows:

The text desktop profile is stored on the system and hence, already available.

To re-use the active desktop profile, drop a text desktop profile object into the Desktop profile capturer object. The tool stores the information in the active binary desktop profile in the dropped text desktop profile.

To re-use the desktop profile, you can also select the Capture desktop option that appears under the Open option in a text desktop profile object pop-up menu.

The tool can also be started from the command line.

When the tool is started, an OS/2 session is opened where it runs; when the processing ends, the session window remains open.

If only the binary desktop profile is available, drop the binary desktop profile object into the Binary-to-text translator object.

If you drop only a binary desktop profile, the tool creates a text desktop profile with the binary desktop profile information. The text desktop profile has the same filename as the binary desktop profile and the TXT extension. If the TXT file already exists, the tool creates a new one.

If you drop both a binary desktop profile and a text desktop profile, the tool creates a new text desktop profile, with the same filename and extension, that only contains the information in the binary desktop profile.

To obtain the text desktop profile, you can also select the Translate to text option that appears under the Open option in the binary desktop profile object pop-up menu.

The tool can also be started from the command line.

When the tool is started, an OS/2 session is opened where it runs; when the processing ends, the session window remains open.

2. Open the text desktop profile by double-clicking on the text desktop profile object.

   You can also edit the file with any text editor that preserves the extended attributes.

3. Specify the desktop profile objects and their properties in the text desktop profile.
4. Save the text desktop profile.

Compiling the text desktop profile

To compile a text desktop profile and obtain a binary desktop profile, drop the text desktop profile object into the Text-to-binary translator object.

If you drop only a text desktop profile, the tool creates a binary desktop profile with the text desktop profile configuration data. The binary desktop profile has the same filename as the text desktop profile and the INI extension. If the INI file already exists, the tool updates the file with the configuration data in the text desktop profile.

If you drop both a text desktop profile and a binary desktop profile, the tool updates the binary desktop profile with the configuration data in the text desktop profile. On an object basis, new data is added or existing data is modified; objects can not be deleted.

To obtain the binary desktop profile, you can also select the Translate to binary option that appears under the Open option in the text desktop profile object pop-up menu.

The tool can also be started from the command line.

When the tool is started, an OS/2 session is opened where it runs; when the processing ends, the session window remains open.

The resulting binary file contains the information about the desktop profile and is ready for test.

Testing desktop profiles

To test desktop profiles, that is, to try the binary files with desktop profile information before releasing them for production, go through the following steps:

1. Capture the original desktop profile so that it can be restored after the test activities. Alternatively, you can always plan on logging off and on again to reset the desktop restrictions after the test.

   You can drop a text desktop profile into the Desktop profile capturer object of the Desktop restrictions workbench folder. The tool stores the information in the active binary desktop profile in the dropped text desktop profile.

   To capture the desktop profile, you can also select the Capture desktop option that appears under the Open option in a text desktop profile object pop-up menu.

   The tool can also be started from the command line.

   When the tool is started, an OS/2 session is opened where it runs; when the processing ends, the session window remains open.

2. Activate the desktop profile to be tested.

   You can drop the binary desktop profile object into the Desktop profile tester object of the Desktop restrictions folder.

   The Workplace Shell is updated with the desktop profile information. The objects provided in the Desktop restrictions folder are not affected.

   To activate the desktop profile, you can also select the Test desktop profile option that appears under the Open option in the binary desktop profile object pop-up menu.

   The tool can also be started from the command line.

3. Check that the Workplace Shell looks and works as defined, according to your test cases.

To restore the original desktop profile, drop the binary desktop profile object into the Desktop profile tester object. You may get the binary desktop profile as follows:

   The binary desktop profile is stored on the system and hence, already available.

   If only the text desktop profile is available, drop the text desktop profile object into the Text-to-binary translator object.

Modifying desktop profiles

To modify a desktop profile, that is, to update the binary file with the desktop profile information, you can go through the following steps.

1. Drop the Workplace Shell objects to be updated into the binary desktop profile object. They can be dropped one-by-one or at a time.

For each dropped object, a notebook is opened. If the object has been already defined for this desktop profile, the values that were specified will be displayed. If the object has not been previously defined for this desktop profile, the current values will be displayed.

2. Use the object notebook to specify:

   A subset of the setup string parameters (Setup tab).

   The OS/2 setup string is used to specify parameters that define the behavior of the object. The SecureEntry parameter list also provides for modifying the link property and enabling to make shadow copy of the object **only** to user folders.

   The options that can be selected from the object pop-up menu (Pop-up tab).

   The pages that can be modified in the object settings notebook (Settings tab).

   For information on how to navigate through the notebook pages and specify configuration data, display help information by selecting Help from the system menu, clicking on the Help button on the notebook page, or pressing F1 with the cursor on a particular page.

3. Close the object notebook when all object properties are specified. You should use the OK push button.

When all object notebooks are closed, the binary file contains the desktop profile information and is ready for test.

To modify a desktop profile, you can also edit the text file with the configuration data and translate the text file into a binary file, which will contain the updated information.

Text desktop profile format

Follows the grammar description for the allowable syntax to be found within a text desktop profile. Note that for non-english SecureEntry language versions the keywords name may have been translated. If so, you can find the correct spelling within the SecureEntryPath\API\NLS\WRKB\EDYNLSxx.DLG, where xx corresponds to the specific country code.

You can replicate the basic object text definition restrictions structure as many times as objects have to be defined.

The following conventions are used:

UPPERCASE CHARACTERS represent values you specify directly, without any change.

lowercased characters represent variables for which you can supply values.

```
[objectname]|[DEFAULT]

  SETUP_DELETE=YES|Y|1|NO|N|0
  SETUP_COPY=YES|Y|1|NO|N|0
  SETUP_MOVE=YES|Y|1|NO|N|0
  SETUP_LINK=YES|Y|1|NO|N|0
  SETUP_RESTRICTED_LINK=YES|Y|1|NO|N|0
  SETUP_VISIBLE=YES|Y|1|NO|N|0
  SETUP_PRINT=YES|Y|1|NO|N|0
  SETUP_RENAME=YES|Y|1|NO|N|0
  SETUP_DRAG=YES|Y|1|NO|N|0
  SETUP_DROP=YES|Y|1|NO|N|0
  SETUP_SETTINGS=YES|Y|1|NO|N|0
```

```
POPUP_OPEN=YES|Y|1|NO|N|0
POPUP_SETTINGS=YES|Y|1|NO|N|0
POPUP_ICON_VIEW=YES|Y|1|NO|N|0
POPUP_TREE_VIEW=YES|Y|1|NO|N|0
POPUP_DETAILS_VIEW=YES|Y|1|NO|N|0
POPUP_PROGRAM=YES|Y|1|NO|N|0
POPUP_PALETTE=YES|Y|1|NO|N|0
POPUP_HELP=YES|Y|1|NO|N|0
POPUP_CREATE_ANOTHER=YES|Y|1|NO|N|0
POPUP_COPY=YES|Y|1|NO|N|0
POPUP_MOVE=YES|Y|1|NO|N|0
POPUP_SHADOW=YES|Y|1|NO|N|0
POPUP_CREATE_PARTITION=YES|Y|1|NO|N|0
POPUP_DELETE=YES|Y|1|NO|N|0
POPUP_FIND=YES|Y|1|NO|N|0
POPUP_PRINT=YES|Y|1|NO|N|0
POPUP_SELECT=YES|Y|1|NO|N|0
POPUP_LOCKUP=YES|Y|1|NO|N|0
POPUP_SHUTDOWN=YES|Y|1|NO|N|0
POPUP_SETUP=YES|Y|1|NO|N|0
POPUP_SORT=YES|Y|1|NO|N|0
POPUP_ARRANGE=YES|Y|1|NO|N|0
POPUP_REFRESH=YES|Y|1|NO|N|0
POPUP_CHANGE_STATUS=YES|Y|1|NO|N|0
POPUP_SET_DEFAULT=YES|Y|1|NO|N|0
POPUP_DISABLE_SPOOLER=YES|Y|1|NO|N|0
POPUP_CHECKDISK=YES|Y|1|NO|N|0
POPUP_FORMATDISK=YES|Y|1|NO|N|0
POPUP_COPYDISK=YES|Y|1|NO|N|0
POPUP_OPEN_PARENT=YES|Y|1|NO|N|0
POPUP_PICKUP=YES|Y|1|NO|N|0

SETTINGS_GENERAL=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_WINDOW=YES|Y|1|NO|N|0
SETTINGS_CLOCK_VIEW1=YES|Y|1|NO|N|0
SETTINGS_CLOCK_VIEW2=YES|Y|1|NO|N|0
SETTINGS_COUNTRY_DATE=YES|Y|1|NO|N|0
SETTINGS_COUNTRY_NUMBER=YES|Y|1|NO|N|0
SETTINGS_COUNTRY_COUNTRY=YES|Y|1|NO|N|0
SETTINGS_COUNTRY_TIME=YES|Y|1|NO|N|0
SETTINGS_DESKTOP_LOCKUP1=YES|Y|1|NO|N|0
SETTINGS_DESKTOP_LOCKUP2=YES|Y|1|NO|N|0
SETTINGS_DESKTOP_LOCKUP3=YES|Y|1|NO|N|0

SETTINGS_DISK_DETAILS=YES|Y|1|NO|N|0
SETTINGS_FILE_MENU=YES|Y|1|NO|N|0
SETTINGS_FILE_TYPE=YES|Y|1|NO|N|0
SETTINGS_FILE_FILE1=YES|Y|1|NO|N|0
SETTINGS_FILE_FILE2=YES|Y|1|NO|N|0
SETTINGS_FILE_FILE3=YES|Y|1|NO|N|0
SETTINGS_FOLDER_BACKGROUND=YES|Y|1|NO|N|0
SETTINGS_FOLDER_INCLUDE=YES|Y|1|NO|N|0
SETTINGS_FOLDER_SORT=YES|Y|1|NO|N|0
SETTINGS_FOLDER_VIEW1=YES|Y|1|NO|N|0
SETTINGS_FOLDER_VIEW2=YES|Y|1|NO|N|0
SETTINGS_FOLDER_VIEW3=YES|Y|1|NO|N|0
SETTINGS_KEYBOARD_MAPPINGS=YES|Y|1|NO|N|0
SETTINGS_KEYBOARD_SPECIAL_NEEDS=YES|Y|1|NO|N|0
SETTINGS_KEYBOARD_TIMING=YES|Y|1|NO|N|0
SETTINGS_MOUSE_MAPPINGS=YES|Y|1|NO|N|0
SETTINGS_MOUSE_TIMING=YES|Y|1|NO|N|0
SETTINGS_MOUSE_SETUP=YES|Y|1|NO|N|0
SETTINGS_CLOCK_ALARM=YES|Y|1|NO|N|0
```

```
SETTINGS_POWER_POWER=YES|Y|1|NO|N|0
SETTINGS_POWER_VIEW=YES|Y|1|NO|N|0
SETTINGS_PROGRAM_PROGRAM=YES|Y|1|NO|N|0
SETTINGS_PROGRAM_ASSOCIATION=YES|Y|1|NO|N|0
SETTINGS_PROGRAM_SESSION=YES|Y|1|NO|N|0
SETTINGS_SOUND_WARNING_BEEP=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_CONFIRMATIONS=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_LOGO=YES|Y|1|NO|N|0
SETTINGS_CLOCK_DATA_TIME=YES|Y|1|NO|N|0
SETTINGS_PRINTER_PRINTER_DRIVER=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_PRINT_SCREEN=YES|Y|1|NO|N|0
SETTINGS_PRINTER_PRINT_OPTIONS=YES|Y|1|NO|N|0
SETTINGS_SPOOL_PRINT_PRIORITY=YES|Y|1|NO|N|0
SETTINGS_PRINTER_OUTPUT=YES|Y|1|NO|N|0
SETTINGS_PRINTER_QUEUE_OPTIONS=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_SCREEN=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_SCREEN2=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_TITLE=YES|Y|1|NO|N|0
SETTINGS_SPOOL_SPOOL_PATH=YES|Y|1|NO|N|0
SETTINGS_PRINTER_VIEW=YES|Y|1|NO|N|0
SETTINGS_GENERAL2=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_WINDOW2=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_WINDOW3=YES|Y|1|NO|N|0
SETTINGS_ARCHIVE=YES|Y|1|NO|N|0
SETTINGS_DESKTOP=YES|Y|1|NO|N|0
SETTINGS_MOUSE_COMET_CURSOR=YES|Y|1|NO|N|0
SETTINGS_MOUSE_POINTER_PAGE=YES|Y|1|NO|N|0
SETTINGS_SYSTEM_INPUT=YES|Y|1|NO|N|0
```

Note: The following keywords apply only when working with IBM OS/2 V3.0 (OS/2 Warp)
:

```
SETUP_SETTINGS
POPUP_OPEN_PARENT
POPUP_PICKUP
SETTINGS_GENERAL2
SETTINGS_SYSTEM_WINDOW2
SETTINGS_SYSTEM_WINDOW3
SETTINGS_ARCHIVE
SETTINGS_DESKTOP
SETTINGS_MOUSE_COMET_CURSOR
SETTINGS_MOUSE_POINTER_PAGE
SETTINGS_SYSTEM_INPUT
```

The objectname parameter value can be either the object title or the object identifier.

The object identifier and the object title are specified in the OS/2 setup string, which is used to specify parameters that define the behavior of the object. The object identifier stays with the object even though the object title is changed or the object is copied. The object you create is not assigned an object identifier because identifiers must be unique through the system.

Write the title or identifier **exactly** like the original, taking especial care about capitals and blanks, and do not leave blanks between the brackets, < >, and the value.

To specify an object title of more than one line, use the \n string.

If you specify DEFAULT as objectname parameter value, the properties you define through the following keywords apply to all objects that do not appear in the text desktop profile.

The first block of keywords corresponds to a subset of the setup string parameters, as the SETUP_ prefix indicates, except for the SETUP_RESTRICTED_LINK keyword.

This keyword restricts the scope of the SETUP_LINK keyword by enabling to make shadow copy of the object **only** to *user folders*.

The keyword applies only if the SecureEntry user management and access control features will be implemented in the target workgroup.

In addition, the SETUP_RESTRICTED_LINK keyword does not apply when making a shadow copy of the object (SETUP_LINK keyword) is not enabled. If the SETUP_LINK keyword is set to enable the property, the SETUP_RESTRICTED_LINK keyword can be set to restrict or not to restrict the property.

When the desktop profile is activated, those properties not defined through the SETUP_ keywords keep the value specified in the previously active desktop profile.

The second block of keywords corresponds to the options that can be selected from the object pop-up menu (POPUP_ prefix).

The third block of keywords corresponds to the pages that can be modified in the object settings notebook (SETTINGS_ prefix). Pages under the same tab are identified by using both the tab label and the page number; for example, SETTINGS_CLOCK_VIEW1 and SETTINGS_CLOCK_VIEW2.

When the desktop profile is activated, those properties not defined through the POPUP_ or SETTINGS_ keywords keep the value corresponding to the OS/2 Workplace Shell.

For all keywords the parameter value can be:

YES, Y, or 1

To enable a setup property, show a pop-up option, or add a settings page.

NO, N, or 0

To disable a setup property, hide a pop-up option, or remove a settings page.

Keywords and parameter values are not case-sensitive.

Text desktop profile sample

```
/* All objects in the desktop are not visible, except    */
/* for those that appear explicitly in the file.         */

    [DEFAULT]
    SETUP_VISIBLE=NO

/* The OS/2 System object would also be visible if the   */
/* SETUP_VISIBLE=YES  keyword-value pair did not appear  */
/* in the file.                                          */

    [OS\2 System]
    SETUP_VISIBLE=YES

    [Workbench]
    SETUP_VISIBLE=YES
    SETUP_COPY=NO
    SETUP_MOVE=NO
    SETUP_DELETE=NO
    SETUP_DRAG=NO
    POPUP_COPY=NO
```

```
POPUP_DELETE=NO

[Desktop]
POPUP_OPEN=NO
POPUP_REFRESH=NO
POPUP_HELP=YES
POPUP_SHADOW=NO
POPUP_LOCKUP=YES
POPUP_SHUTDOWN=YES
POPUP_SETUP=NO
POPUP_FIND=NO
POPUP_SELECT=NO
POPUP_SORT=NO
POPUP_ARRANGE=NO
```

Managing desktop objects position

In order to control how SecureEntry will manage object positions within the Workplace, there are three different options :

1. The checkbox 'Save Desktop Settings' is checked in the Desktop page of the settings notebook for the desktop.

   In this case, object positions are handled by the Workplace Shell, so that moving an object to another position is just a matter of really dragging it, and it will stay there across sessions and shutdowns. Of course, you will only be allowed to move it if the desktop restrictions profile for your working session allow you to do so.

2. The checkbox 'Discard all changes on folder's positions when logging off' is checked in the Desktop page of the settings notebook for the desktop.

   In this case, SecureEntry maintains the positions of folders and contents so that they will always appear in the same place for all users right after logon, but you can play with your folder's positions during your work session.

   Administrator users have the additional option then to save the positions at any moment by using the added 'save positions' option in the affected folder's view popup menu.

3. The checkbox 'Discard all changes on folder's positions as soon as they are closed' is checked in the Desktop page of the settings notebook for the desktop.

   In this case, SecureEntry maintains the positions of folders and contents so that they will always appear in the same place for all users every time they are opened.

   Administrator users have the additional option then to save the positions at any moment by using the added 'save positions' option in the affected folder's view popup menu.

Note : Changes made to the desktop page of the settings notebook for the desktop will only be honored after a WorkPlace Shell reload. In other words, you may have to reboot the machine after checking the appropriate value to observe the desired desktop behavior.

Workplace Shell personalization

When a SecureEntry user opens a SecureEntry session, the Workplace Shell is modified according to the specifications in the assigned desktop profile. The end user can then change the layout properties of the folders, the desktop folder inclusive:

Positions of the icons in the folder.

Through the Sort or Arrange option in the folder pop-up menu, through the Sort page in the Settings notebook, or by dragging and dropping.

View properties of the folder.

Through the View pages in the Settings notebook: Icon view, Tree view, and Details view.

When the SecureEntry user closes the SecureEntry session and opens a new one, the initial settings are restored so that the end user is always provided with the same Workplace Shell, regarding the properties defined through these settings. This is true provided that the workplace is configured that way, as explained under Managing desktop objects position.

```
**********************************************
IMPORTANT NOTE

What follows is  a  description of the  maintained
functionality  on how  to  create  personalized
(user associated) folders  and  memory folders
using the desktop restrictions component through
its  text  profile  definition  characteristics.
This function is  still available and  supported
for compatibility reasons, but you are encouraged
to use  The Personal desktop component to do the
same tasks more efficiently from now on.
**********************************************
```

However, SecureEntry enables the system administrator to permanently associate the new settings, that is, the settings specified during a SecureEntry session, to the end user. To enable a folder to keep the new settings, when the user closes the folder or performs logoff, the system administrator should define it as *memory folder*.

In addition, SecureEntry enables the system administrator to provide SecureEntry users with *user folders*, where the end user can create or delete object shadows through the Create shadow option or Delete option, respectively, in the object pop-up menu. Object shadows can also be created or deleted by dragging and dropping objects into the user folder or into the OS/2 Shredder object, respectively.

The end user may keep the shadows of the objects that are more frequently used in the user folder. The contents of the user folder are associated only to the SecureEntry user that is logged-on.

Memory folders

If a folder is defined as memory folder, the changes performed during a SecureEntry session to some layout properties are stored in a file associated to the user that is logged-on. When the user opens another session, the layout properties of the folder are restored according to the settings active the last time the user performed logoff.

The folder properties that are stored for the memory folders are:

Positions of the icons.

View properties that can be set through the View pages in the Settings notebook: Icon view, Tree view, and Details view.

The system administrator can define as memory folder any folder with an object identifier, the desktop folder inclusive.

User folders

User folders are a SecureEntry class of folders where object shadows can be created or deleted.

User folders can exist only in the desktop folder and can contain only shadows of objects. They are removed from the desktop folder at logoff time and re-created with all their contents when the user performs logon. When a user folder is re-created, its title and object identifier are also restored.

To create a shadow of an object to a user folder, the object has to be accordingly defined in the desktop profile assigned to the SecureEntry user that is currently logged-on. That is, the making shadow copy property must be enabled (SETUP_LINK keyword) but restricted to user folders (SETUP_RESTRICTED_LINK keyword).

If a SecureEntry user is to be provided with a user folder, the first time that logon takes place, an empty user folder is created.

The system administrator can specify whether a given SecureEntry group is provided with a user folder through the personalization profile.

In addition, user folders should be defined through a settings notebook.

Configuring personalization profiles

Personalization profiles are configured on a SecureEntry group or user basis. That is, one personalization profile can be configured for each SecureEntry group or individually to a SecureEntry user.

To configure a personalization profile, you can produce a text file with the configuration data and translate the text file into a binary file, which will contain the personalization profile information.

Go through the following steps:

Editing the text personalization profile

To edit and obtain a text personalization profile, go through the following steps:

1. Open a text file with any text editor that preserves the extended attributes. You can use the file corresponding to an existing text personalization profile.
2. Specify the configuration data in the text file.
3. Save the text file.

Text personalization profile format

You can replicate the basic structure of a memory folder as many times as memory folders are to be defined. Only one user folder can be defined. The following conventions are used: ;p.

UPPERCASE CHARACTERS represent values you specify directly, without any change.

regular characters represent variables for which you can supply values.

Text personalization profile keyword structure

```
[objectname]
  SAVE_ICONS_POSITION=YES|Y|1|NO|N|0

[_SE_PERSONAL_FOLDER_]
  CREATE_FOLDER=YES|Y|1|NO|N|0
```

The objectname parameter value can be either the object title or the object identifier corresponding to the memory folder.

The SAVE_ICONS_POSITION keyword can be set to save (YES, Y, or 1) or not to save (NO, N, or 0) the position of the objects that the memory folder contains at logoff time and to display them in that position after logon.

The CREATE_FOLDER keyword can be set to create (YES, Y, or 1) or not to create (NO, N, or 0) a user folder at startup time.

Keywords and parameter values are not case-sensitive.

Text personalization profile sample

```
/* The desktop folder keeps its layout and the productivity    */
/* folder does not keep its contents positions.  The other     */
/* folders do not keep their contents positions, either.       */

       [<WP_DESKTOP>]
         SAVE_ICONS_POSITION=YES

       [<WP_TOOLS>]
         SAVE_ICONS_POSITION=NO

/* The user is provided with a user folder.                    */

       [_SE_PERSONAL_FOLDER_]
         CREATE_FOLDER=YES
```

Compiling the text personalization profile

To compile a text personalization profile and obtain a binary personalization profile, drop the text personalization profile object into the Text-to-binary translator object.

The tool creates a binary personalization profile with the text personalization profile configuration data. The binary personalization profile has the same filename as the text personalization profile and the PER extension.

Desktop profile activation API

Description

The Desktop restrictions workbench API consists of:

A dynamic link library called EDYSHELL.DLL. Programs that issue requests to the Desktop restrictions API function must be dynamically linked to this library.

A link time library called EDYSHELL.LIB. Programs that issue requests to the Desktop restrictions API function at link time must be linked to this library.

An include file called EDYSHELL.H. Programs that use the Desktop restrictions API must include this file. This file declares information required to use the Desktop restrictions API function.

The API provides only the edy_refresh_profile function, intended to activate desktop profiles.

When your program issues this function, the objects in the Workplace Shell and the object properties are refreshed according to the information in the desktop profile that is activated.

edy_refresh_profile syntax

The following statement shows how to include this function into your programs:

```
LONG APIENTRY edy_refresh_profile (CHAR *profilename)
```

Where profilename is the full path of the file that contains the desktop profile to be activated. A NULL value is resolved to SecureEntryPath\WORK\EDYDESK.INI or, if this file does not exist, it is resolved to SecureEntryPath\NOUSER\EDYDESK.INI

edy_refresh_profile return codes

The following diagram show the codes that can be returned by this function:

```
Code         Meaning
=====        =======

0001x        Successful operation
0000x        Error refreshing Workplace Shell
FFFFx        Error catching Workplace Shell return code
Other        OS/2 error
```

Commands syntax

This section describes the commands to be entered at the OS/2 prompt and start the tools provided by the Desktop restrictions component.

Desktop activation command

The EDYREFR tool is provided to activate desktop profiles from the command line. To start the tool, enter:

```
EDYREFR [fullpath]
```

fullpath

Full path of the file that contains the desktop profile to be activated.

The default is SecureEntryPath\WORK\EDYDESK.INI or, if this file does not exist, SecureEntryPath\NOUSER\EDYDESK.INI.

Text-to-binary translator command

To start the text-to-binary translator tool, enter:

```
EDYT2B text-filename binary-filename
```

text-filename

Name of the file corresponding to the text desktop profile. This is the input file.

binary-filename

Name of the file corresponding to the binary desktop profile. This is the output file.

Binary-to-text translator command

To start the binary-to-text translator tool, enter:

```
EDYB2T binary-filename text-filename
```

binary-filename

Name of the file corresponding to the binary desktop profile. This is the input file.

text-filename

Name of the file corresponding to the text desktop profile. This is the output file.

Desktop profile capturer command

To start the desktop profile capturer tool, enter:

```
EDYSAVEE  [filename] [/ID:(YES|NO)]
```

filename

Name of the file corresponding to the text desktop profile. The default is **EDYDESK.TXT**.

ID

Specifies whether objects are to be identified by the object identifier (YES value) or by the object title (NO value). The default is **NO**.

Desktop profile tester command

To start the desktop profile tester tool, enter:

```
EDYTEST [filename]
```

filename.ext

Name of the file corresponding to the binary desktop profile.

The default is SecureEntryPath\WORK\EDYDESK.INI or, if this file does not exist, SecureEntryPath\NOUSER\EDYDESK.INI.

Note that this utility is maintained only for SecureEntry previous version compatibility reasons, so we suggest you to use the EDYREFR utility instead.

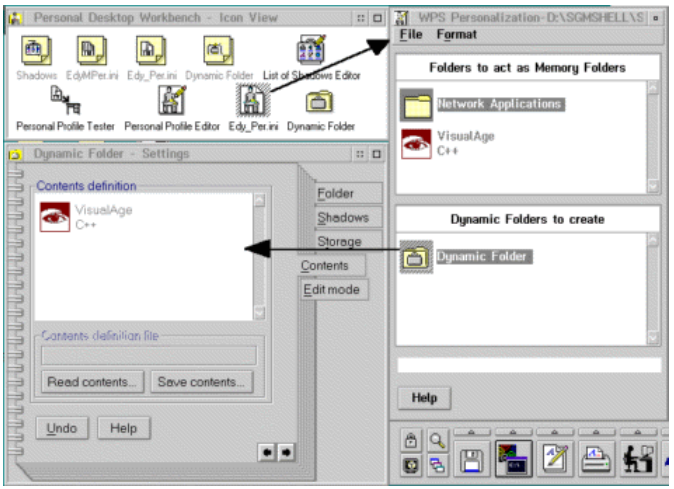
The Personal desktop component

SecureEntry enables the system administrator to permanently associate a given folder's settings, that is, the settings specified during a SecureEntry session, to the end user. To enable a folder to keep the new settings, when the user closes the folder or performs logoff, the system administrator should define it as *memory folder*.

In addition, SecureEntry enables the system administrator to provide SecureEntry users with *user folders*, where the end user can create or delete object shadows through the Create shadow option or Delete option, respectively, in the object pop-up menu.

You use the Personal desktop component to define personal desktop profiles, which specify both, memory and user folders. user folders will from now on be called *dynamic folders*, since they are effectively created and destroyed for the user at logon/logoff times. This personal desktop profiles, once created, are assigned to SecureEntry users by means of the SecureEntry administration tools.

In order to work with this component, please open the Personal Desktop workbench folder located in the SecureEntry workbench folder.



Use the objects in the Personal Desktop workbench to

Define Dynamic Folders

Edit Personalization Profiles and Models

Edit Memory folders

Edit Lists of Shadows through the list of shadows editor

Dynamic folders

Dynamic folders (also called user folders) are special folders that

can only contain shadows. Shadows can be added by drag and drop operations.

store the list of contained shadows in a file that can be used in different workstations.

are created on logon and deleted on logoff.

can be defined as startup folders, having their contents opened on logon.

The list of contained shadows can be kept in the user's personalization profiles or in an external file. When the list is kept in the user's personalization profile, the contents of the folder are defined by the user, and the user will have access to his own personal folder no matter the workstation where he is working. When the list is kept in an external file, the administrator sets if the contents are to be defined by him or by the users.

Dynamic folders are suitable for

allowing users to have a personal folder, where they add the objects they choose.

allowing users to have a personal startup folder, where they add the objects that they want to be opened on logon.

having startup folders defined by the administrator.

having workstation dependant folders, where the administrator defines contents that match different workstation configurations.

having folders with their contents defined by a set of users in the LAN.

Since dynamic folders create shadows of objects, the objects that have to be shadowed must be available in the workstation where the folders are created.

A dynamic folder is assigned to a personalization profile or model by dropping a dynamic folder model on the personalization editor's lower container.

Personalization profiles and models

The personalization profiles specify

the dynamic folders that will be created on logon

the folders that will act as memory folders on logon.

The personalization profiles (created through the object model *Edy_Per.ini*) may contain information unique to the user, and they are stored by SecureEntry when the user logs off. Administrators can only define personalization profiles on a user basis, not on a group basis or in the NOUSER folder. If this is done, the profile will be taken as a model for all users as they log on in order to create their specific personalization profile.

The personalization models (created through the object model *Edy_MPer.ini*) have the same format as the personalization profiles, but do not contain any user dependant information. They can be used by administrators to dynamically create or update personalization profiles. Personalization models can be defined on a group basis.

A default personalization model can be placed in the NOUSER folder. The name of the default personalization mode has to be **EDYMPER.INI**.

When a personalization model is available, either because it has been assigned to the user or to the user's group, or because it is placed in the NOUSER folder, the user's personalization file is updated with the definitions of the personalization model:

if the user does not have a personalization profile, a personalization profile will be created with the definitions of the personalization model. The new personalization profile will be assigned to the user on logoff.

any memory or dynamic folders defined by the profile that are not defined by the model will be deleted from the profile.

any memory or dynamic folders defined by the model that are not defined by the profile will be added to the profile.

any dynamic folders defined by the model that are also defined by the profile will have their settings updated with the information in the model, but will keep the contents stored in the profile.

Memory folders

When a folder is defined to act as a memory folder in the user's personalization profile, any changes that the user performs on the folder's layout is stored in the personalization profile, and applied to the folder whenever the user logs on to the same or another workstation. The layout information that is stored and applied includes

window position.

contents positions.

view attributes (container attributes and fonts).

Folders can only be defined to act as memory folders when the have an object ID defined.

Lists of shadows

List of shadows files are used to store the contents of dynamic folders, and to define the startup process (*EdyStart* object within the *NOUSER* folder).

List of shadows files can also be used when editing dynamic folder models.

List of shadow files can be viewed and edited with the List of shadows editor or from the Contents page of dynamic folder models.

Shadows can be added to the list by dropping them on the container, and deleted by using their popup menu.

The Hooked objects component

The hooked objects component allows you to create a security profile where you specify which of the desktop's objects to establish a hook over. This means, basically, that you can get an external program called whenever a given object is opened or closed. This external program, in the case of object open event, can optionally decide whether to honor or not honor the open request, thus making it extremely easy to configure a protection scheme based on one time passwords request for program or folder objects. The only requirement that hooked objects must have is that they have to be objects with an assigned objectID.

Within this component's workbench, SecureEntry provides a sample hook program that implements a reasonably safe one time password algorithm, that depends on the institution name and the date/time, so that just by drag and drop you can configure an object to be opened only if that password is given. The password itself can be retrieved by running the one time password generator by a SecureEntry administrator which is running SecureEntry in another machine of the same institution.

In order to establish hooks for a given component, you should create a profile of this component's type, by dragging one instance from the supplied template object. Then edit it by dropping the profile object into the editor. Refer to the help information of the editor program in order to configure a given hook.

As usual, and after you have established the desired hooks, you can assign the profile to users, groups, or as machine default by putting it in the NOUSER directory within the SecureEntry path.

The one time password utilities

One Time Password Generation

One Time Password Verification

Default OTP generation/verification

The one time password utilities

This utility allows you to work with one time passwords bound to particular <date,time> pairs.

The generation utility, **edyotpg.exe**, located in the **<sgm_shell>\exec** directory, takes as input a <date,time> pair and outputs the password bound to that <date,time> pair.

The verification utility, **edyotpv.exe**, located in the **<sgm_shell>\exec** directory, takes as input a password and checks it against the current <date,time> pair. If successful it returns zero, otherwise it returns 1.

You can supply to both utilities your own generation/verification routines or work with the default ones incorporated in them.

One time passwords are useful to allow access to restricted resources under administration supervision: when the user attempts to access the resource edyotpv.exe is executed, the user then must contact an administrator to get the one time password bound to the <date,time> pair edyotpv.exe is asking for, the administrator then runs the edyotpg.exe utility with the <date,time> pair the user has told him and gets the one time password that the user needs, finally he tells it to the user in some secure way. Of course, all this works out only if users cannot modify the system's date and time.

One Time Password Generation

This program is designed to take as input a date **(dd/mm/yyyy)** and a time **(hh:mm:ss)** and generate as output a password that should be valid only for this particular date and time ,or seed string.

```
edyotpg [/G<pathname>] [/K] [/R(S|D|M|H)] [/M]
```

<pathname>

Is the pathname of your OTP optional generation routine.

/K

Use this parameter to work with reference seed strings instead of date/time pairs.

/R

Set resolution to (S)econds, (D)ays, (M)inutes, or (H)ours

/M

Force dialog sysmodality

Your own one time password generation routine must follow some guidelines to properly interface to edyotpg.exe:

It must be an executable file. (.exe)

The first two command line arguments must be two strings, the first one being the **date** with the dd/mm/yyyy format and the second one the **time** with the hh:mm:ss format. In the case that reference seeds are used, then only one parameter, containing the **seed** (6 characters), will be passed.

It must return 0 if the generation was successful and something different from zero otherwise.

The generated password must be written to standard output and must be the first program's output to standard output.

All other messages, such as error messages, must be written to standard output if you want edyotpg.exe to show them when things go wrong, that is, when your routine returns something different from zero.

One Time Password Verification

This program is designed to check the password entered by the user against the current date and time, and, if successful, execute a program that was specified as a command line argument with the **'/E'** modifier. The other command line arguments are used to change the behavior of the program, as follows:

```
edyotpv [/E<pathname>] [/ID<string>] [/Dn(S|D|M|H)] [/K] [/R(S|D|M|H)] [/M]
```

<pathname>

is the program's pathname you want to execute on success.

/ID<string>

Replace string with an instance identifier, in the case you have more than one resource protected through this program in a given machine.

/DnX

Password valid duration time. If this parameter is specified, once a valid password has been entered, this password will be considered valid for the same ID instance during the given timeframe. i.e, /D3H means 'keep passwords valid during 3 hours'.

/K

Use this parameter to work with reference seed strings instead of date/time pairs.

/R

Set resolution to (S)econds, (D)ays, (M)inutes, or (H)ours

/M

Force dialog sysmodality. Use this option if you want to ask for a OTP password through a user exit during session flow (logon/logoff/unlock).

Note: if no command line (/E) argument is specified then no program is executed.

If the password is validated edyotpv.exe **returns** a **0**, otherwise returns a **1**

You can supply your own verification routine by naming it **edyotpv.exe** and locating it in the **<sgm_shell>\nouser** directory, edyotpv.exe looks for this file and if it exists then uses it to validate the password.

Your validation routine must follow some guidelines to properly interface with edyotpv.exe:

It must be named **edyotpv.exe** or **edyotpv.cmd** and be located in the **<sgm_shell>\nouser** directory.

The first three command line arguments must be two strings, the first one being the **date** with the dd/mm/yyyy format, the second one the **time** with the hh:mm:ss format, and the third one the **password in uppercase**, to be validated. In the case that reference seeds are used, then instead of the three parameters, only two will be passed (**seed** (6 characters) and **password** to validate).

It must return **0** if password was validated and something different from zero otherwise.

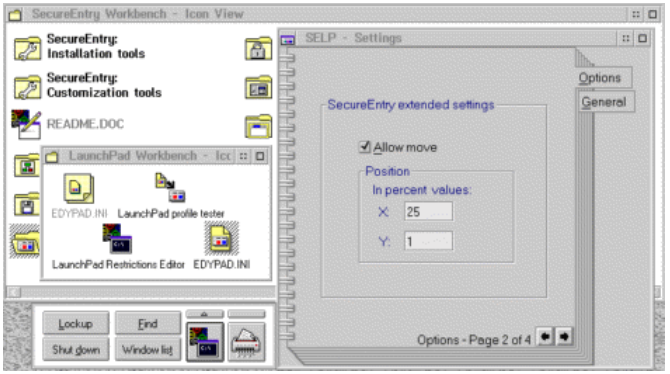
All error messages must be written to standard output if you want edyotpv.exe to show them whenever the password entered by the user is not valid. (if it is valid, edyotpv.exe will not show anything)

Default OTP generation/verification

The default OTP generation/verification routines work with **six character long passwords over the alphabet ({'A'..'Z'} - {'P','T','V','N'}) U {'0'..'9'}**. The characters 'P','T','V','N' are excluded for phonetical reasons and the default validation routine treats the characters 'V' and 'P' as the character 'B', the character 'T' as the character 'D', and the character 'N' as the character 'M'. This is to prevent possible phonetical misunderstanding in the oral communication of one time passwords between the administrator and the final user.

The Launchpad component

The launchpad component allows you to create launchpad security profiles which contain the necessary information to recreate a given launchpad with some special features once the component is activated with a given profile. This is very useful in order to create custom launchpads for your users depending on their needs or requirements.



In order to work with this component, please open the Launchpad Workbench folder located in the SecureEntry Workbench folder.

In order to create a security profile for this component, drag one from the component profile template object (named EDYPAD.INI).

To modify a profile of this type, you can drop it into the 'Launchpad Restrictions Editor' program object, and immediately a launchpad will appear in editing mode. You can distinguish this one from your regular launchpad by the fact that the one being edited has, in its popup menu, the entries 'Save' and 'Close', and a different background color (white). Edit the launchpad as you would do normally (drag and drop), and after that select the 'Save' and/or 'Close' as necessary. Note that you have some settings available for this object which are specific SecureEntry features :

Whether to allow moving of the launchpad.

Initial position of the launchpad.

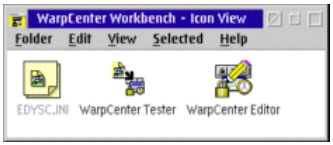
Whether to deactivate some or all of the action buttons.

In order to test a given Launchpad profile, just drop it into the 'SELP profile tester' program object.

Once the launchpad is customized, drop its profile into the appropriate place (NOUSER folder or administration tool) to assign it to a given user, group or machine configuration.

The WarpCenter component

The WarpCenter component allows you to create WarpCenter security profiles which contain the necessary information to customize the WarpCenter with some special features once the component is activated with a given profile. This is very useful in order to create customized WarpCenters for your users depending on their needs or requirements.



In order to work with this component, please open the WarpCenter Workbench folder located in the SecureEntry Workbench folder.

In order to create a security profile for this component, drag one from the component profile template object (named EDYSC.INI).

To modify a profile of this type, you can drop it into the 'WarpCenter Restrictions Editor' program object, and immediately a WarpCenter will appear in edition mode. You can distinguish this one from your regular WarpCenter by the fact that the one being edited is enclosed in a white border and its popup menu has specific customization entries.

Edit the WarpCenter as you would do normally (properties,drag and drop, ...), and after that select the 'Save' or 'Close' as necessary. Note that you have some settings available for this object which are specific SecureEntry features:

Whether to use the OS2 standard window list or the WarpCenter's window list.

Enable/Disable the WarpCenter's FIND button and the popup menu's FIND entry of the objects contained in the WarpCenter.

Enable/Disable the popup menu's ORIGINAL entry of the objects contained in the WarpCenter.

Enable/Disable the popup menu's DELETE entry of the objects contained in the WarpCenter.

Whether to allow for the shutdown icon in the warpcenter to have its functionality changed to logoff.

Note also that the original WarpCenter's lockup and shutdown procedures have been substituted by the SENTRY's lockup and shutdown procedures.

In order to test a given WarpCenter profile, just drop it into the 'WarpCenter profile tester' program object, a green border will enclose the WarpCenter while the profile is activating and will disappear as soon as activation ends.

Once the WarpCenter is customized, drop its profile into the appropriate place (NOUSER folder or administration tool) to assign it to a given user, group or machine configuration.

New Setup Strings

WarpCenter Environment Variables

New Setup Strings

These are the new setup strings defined for the SmartCenter class. They will only work on a SmartCenter class instance which has <WP_WARPCENTER> as its object ID.

EDYSCOPEN=YES

Opens the WarpCenter's default view (same as double-click).

EDYSCCLOSE=YES

Closes the WarpCenter's default view.

EDYSCDELETETRAY=tray number

Deletes the specified tray.

EDYSCTRAYNAME=tray name

Renames the current tray.

EDYSCCONFIRMSHUTDOWN={YES|NO}

Enable/disable shutdown confirmation.

EDYSCLARGEICONS={YES|NO}

Use large or small icons.

EDYSCDISPLAYALWAYS={YES|NO}

Display the WarpCenter only when mouse is over it or always.

EDYSCFLOATONTOP={YES|NO}

Makes WarpCenter to float (or not) on top of maximized windows.

EDYSCTOPPOSITION={YES|NO}

Display the WarpCenter at the top or bottom of the screen.

EDYSCBUBBLEHELP={YES|NO}

Enable/disable the WarpCenter's bubble help.

ADDTRAY=tray name [,object ID,...]

Adds a new tray with the specified name containing the specified objects.

Environment Variables

These are environment variables which can be defined in your CONFIG.SYS file to modify some of the WarpCenter's properties:

SCKillFeatureEnabled=YES

Allows to kill processes from the WarpCenter's window list when selecting it while pressing the Ctrl key. This variable works only if the WarpCenter is customized to use its original window list procedure.

SCFindUtility=executable pathname

The specified executable is started whenever the WarpCenter's FIND button is selected. This variable works only if the WarpCenter is customized with its FIND button enabled.

SCUsePrettyClock=YES

The WarpCenter's displays a more "pretty" clock than the default one.

SCCanbeNuked=1

Allows to delete SmartCenter class instances. This variable works only if SGM_EDYSC_DISABLE is defined.

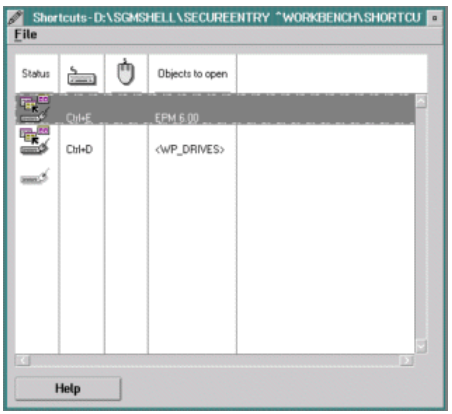
SGM_EDYSC_DISABLE=YES

Disables all the WarpCenter's SENTRY features leaving it as originally defined.

The shortcuts component

The shortcuts component allows you to define keyboard and mouse combinations which, when activated, are able to open a given object.

In order to create a shortcuts profile, obtain one from the template object located within the shortcuts workbench, and drop it into the editor, then follow the editor's help for further advice.



Once the profile has been created, you can assign it to a given user or group by following the standard SecureEntry administration procedures.

Special care has to be taken as to not define keyboard combinations which are already used by the system for other purposes.

Public Applications Component

The public applications component allows for handling, in LAN Server environments, the definition of its public applications. Furthermore, you can use this component to define a SecureEntry profile containing the LAN Server domain public applications that must be shown in a user's *Network Applications* folder as soon as she/he logs on the system.

In order to work with this component, open the *Public Applications Workbench*, located inside the *SecureEntry Workbench*.

You can create a profile for this component by dragging a copy from its component profile template (*EDYROAM.INI*) and dropping it.

In order to modify a public applications profile, drag and drop it into the *Public Applications Editor*, or just double click it. Make the necessary changes and then save the modified profile. Follow the editor's help for further advice.

The editor may be also useful to configure, not only the set of public applications that must appear in the user's *Network Applications* folder at logon time, but the public applications defined on the domain controller of the LAN Server domain where you are logged on.

You must beware of the following points:

The editor of this component obtains the name of the domain controller the administrator user is logged on. This means that if the user has done a SecureEntry logon from the domain controller to another domain, and then he manually logs off, then he will be able to continue administering the public applications of the domain controller he physically is at.

The editor of this component handles two lists of applications: the list on the left side contains the definitions of the public applications to assign to a SecureEntry profile and it uses the definitions of the existing resources in the repository SecureEntry. The list on the right side contains the definitions of the public applications that must reside in the domain controller. This means, in corporate environments using UCM, that the resources, i.e., the alias, available may differ in both lists.

The OS/2 public applications edition notepad contains the tab *Parameters*. You can provide information in this tab, no matter the environment you are working in. However, the data you enter will be effective only if you are administering public applications for WorkSpace On-Demand environments. Refer to WorkSpace On-Demand considerations for more information.

In order to verify the profile behavior, drag and drop it on the object *Test Public Applications*, or select the option test through the editor menu bar. The defined profile will be activated, and the *Network Applications* folder will be rebuilt. If the test is done from the editor, then the contents of the list of public applications defined on the domain controller will be refreshed. Otherwise you will have to select the Refresh option from the pull-down menu related to the option Domain of the menu bar of the editor.

Once configured, associate the profile to the desired user or group, by dragging and dropping it into the adequate container of the users and groups administration tool, or establish it as the default for the system, copying it with its default name (EDYROAM.INI) to the *NOUSER* folder.

When a user logs on the system, its related profile will be activated, yet his or his SecureEntry group's if his does not exist. If the user does not have any defined public applications profile, then the profile residing in the *NOUSER* folder will be activated. If the *NOUSER* folder does not contain any public applications profile, then the set of public applications the user currently has will be preserved.

Take into account that a GUEST SecureEntry logon does **not** imply the activation of the EDYROAM.INI profile residing in the folder *NOUSER*, if it exists.

Description of the modules and the API

The names of this component include de word *ROAM*, since the set of definitions of public applications assigned to a user or group of users travel to the workstation the user uses to log on. This is especially important in corporate environments, where the users administration is centralized through UCM.

What follows is the description of the modules integrated in this component:

EDYROAME

This is the program editor of profiles. It accepts the following syntax:

```
EDYROAME.EXE [Profile]
```

Where *Profile* is the path and file name of the desired profile. The editor allows for testing a profile directly.

EDYROAMT

This is the program activator of profiles. It accepts the following syntax:

```
EDYROAMT.EXE /F[Profile] [/T]
```

/F*Profile* activates the roaming desktop defined in *Profile*. If *Profile* does not exist, the set of user's current public applications will be preserved.

/F activates the roaming desktop EDYROAM.INI defined in the *NOUSER*. If the profile does not exist, the set of user's current public applications will be preserved.

/T indicates that the roaming desktop must be activated in test mode. This parameter restores back to the user the set of public applications she/he had before the test.

Public applications component interface

The component exports the following routine, defined in *EDYROAM.H*. The routine executes correctly if it returns zero; otherwise it returns the gotten error:

```
APIRET _System SetRoamingDesktop(PSZ  pszFileName, BOOL Testing);
```

Activates the user Roaming desktop defined in pszFileName. If pszFileName contents is empty, then any defined Public Application assigned to current user is deleted.

If pszFileName points to "", then NOUSER ini file will be handled.

If the file does not exist, then current user Public Applications will be preserved.

The parameter Testing must state whether or not doing a real logon (FALSE) or doing a profile test (TRUE).

The Log File

The log file takes the name EDYROAM.LOG and resides in the path specified by the environment variable SGM_ROAM_LOGPATH. This file contains the errors and warnings found by the component during its execution.

```
   Date      Time     Program    Severity   Error           Explanation
---------- -------- ------------ -------- ---------- -------------------------
29/09/1998 16:03:01  EDYROAME.EXE ERROR    0X0000000D Could not determine  logon domain.
                                           Probably there's no user logged on.
29/09/1998 16:04:23  EDYROAME.EXE WARNING  0X00000000  NetServerEnum2  returned 0 entries.
                                           Retrying with   NetGetDCName.
29/09/1998 16:05:48  EDYROAME.EXE ERROR    0X00000015  DosQueryPathInfo:  could not get
                                           file information about   "A:\NOALIAS.INI".
29/09/1998 16:06:32  EDYROAME.EXE ERROR    0X00000ADF NetAliasGetInfo:  the alias  "ARBRE"
                                           is not defined in the Domain Controller.
```

This sample shows that the public applications editor got three errors and one warning during its execution.

The first error states that the domain against which the logon was done could not be determined and suggests that probably a non-SecureEntry manual logoff was done. The OS/2 error 0X0000000D was returned to the editor, meaning that *Data is invalid*.

The only existing warning states that a LAN Server API returned and error and that an alternative API was used. Since it was a warning, no error was returned to the public applications editor.

Remember that you can control the size of the SecureEntry log files, as well as any other ASCII file by using the EDYLOGFS utility.

WorkSpace On-Demand considerations

Like with any other SecureEntry component, you can administer the set of public applications to assign to a user or group from any workstation. However, if you are administering or testing public applications with parameters from a **non** WorkSpace On-Demand administrator workstation, then the update of the parameters of the public applications will **not** succeed in the WorkSpace On-Demand server if there's not an administrator user logged on in the mentioned server.

The best suggestion is to administer from the WorkSpace On-Demand Administration Client, or from the WorkSpace On-Demand Server -if the LAN Server administration tools were installed there-, if you must administer public applications with parameters.

If a user, yet administrator or not, having assigned a set of public applications with parameters logs on in a **non** WorkSpace On-Demand workstation, the possible update of the public applications parameters will **neither** succeed in the WorkSpace On-Demand server if there's not an administrator user logged on in the mentioned server.

Notice that, in this case, the folder of public applications assigned to the user will remain containing the assigned applications, since the parameters of the public applications are meaningless in these workstations.

WorkSpace On-Demand specific parameters for public applications

What follows is the list of WorkSpace On-Demand specific parameters that may be helpful when configuring public applications:

WSOD_LAUNCH_MINIMIZED. Assign the value 1 to this environment variable to make the application to start minimized.

WSOD_LAUNCH_NOCLOSE. Assign the value 1 to this environment variable to make the DOS or OS/2 VIO window to not close when the application ends. This may be useful when having to determine the cause of a problem with the application definition.

WSOD_LAUNCH_SESSION. This parameter gives the administrator full control over the type of session in which WorkSpace On-Demand will launch the application. The application may not work at all if you choose the wrong type of session. This parameter may take the following values:

1. Full-screen OS/2

2. Windowed OS/2

3. OS/2 PM

4. Full-screen DOS

7. Windowed DOS

10. WinOS2 Real Mode

12. WinOS2 Auto

15. WinOS2 Standard, seamless, separate VDM

16. WinOS2 Standard, seamless, common VDM

17. WinOS2 Enhanced, seamless, separate VDM

18. WinOS2 Enhanced, seamless, common VDM

19. WinOS2 Enhanced, full-screen

20. WinOS2 Standard, full-screen

WSOD_LAUNCH_NODROP. Assign the value 1 to this environment variable to signal the application launcher to leave the network resources attached.

NCC_SETUP_POST. This environment variable allows to provide setup strings to the application. The parameter value can consist of any of the setup strings specified below. Each setup string consists of a key name, followed immediately by an equal ('=') sign and a value. Multiple setup strings can be passed in the environment variable by separating each setup string with a semicolon (';'). Some of the most useful setup strings are:

CCVIEW. It can take the values *DEFAULT*, *YES* or *NO*.

ICONFILE. It must take the file name that sets the object's icon.

ICONPOS. It must take the percentage coordinates pair (x,y) where the object must be placed in the desktop.

ICONRESOURCE. It must take the pair (id,module) that designates the resource Id and the module name that contains the object's icon.

For example, to make the object to initially appear at ten percent from the origin of the coordinates, assign the following value to the variable **NCC_SETUP_POST** :

```
ICONPOS=10,10;
```

If you are configuring public applications to be executed on WorkSpace On-Demand 2.0 requesters, then you can also use the following parameters:

NCC_PREFITS. Assign a FIT file name to this environment variable. The FIT file entries contained within this file are preppended to the default user FIT file entries for the logged on user. Use this file to force FIT file entries to be found before similar entries in the default user FIT file, or to introduce new FIT file entries.

NCC_POSTFITS. Assign a FIT file name to this environment variable. The FIT file entries contained within this file are appended to the default user FIT file entries for the logged on user. Use this file to force FIT file entries to be after similar entries in the default user FIT file, or to introduce new FIT file entries.

NCC_FOLDER. This environment variable allows to provide setup strings to the application. Use this parameter to place a program object in a folder on the Desktop. You can use any WPFolder and WPObject setup strings. For example, to make the application to reside inside the *Managers* folder, assign the following value to the variable **NCC_FOLDER** :

```
NCC_FOLDER=TITLE=Managers;
```

Since WorkSpace On-Demand 2.0 allows for multilevel folders, you can use the **NCC_CHILD_SETUP** setup string, which is part of the **NCC_FOLDER** parameter, to specify a child folder. For example, if **NCC_FOLDER** contained the following setup string:

```
NCC_FOLDER=TITLE=Secretaries;NCC_CHILD_SETUP=BOOKKEEP;
```

and another parameter BOOKKEEP also existed and contained the following value:

```
BOOKKEEP=TITLE=BookKeeping;
```

then a folder called *Secretaries* would be created on the Desktop, a folder called *BookKeeping* would be created in the *Secretaries* folder, and the program object for the application would be created in the *BookKeeping* folder.

To get more information on WorkSpace On-Demand specific parameters for public applications, refer to the publications *WorkSpace On-Demand HandBook* and *WorkSpace On-Demand Administrator's Guide*.

To get more information on setup strings, refer to the publication *Workplace Shell Programming Reference*.

Public Applications object IDs

In conventional LAN Server environments, public applications assigned to a user are located in the *Network Applications* folder. The object ID of this folder is **<\NETAPPS\NETAPPFLDR>**. Each program object inside this folder is created with the object ID **<***_ServerName_*****_ApplicationId_**>**, where *ServerName* is the the server name of the user logon domain, and *ApplicationId* is the name that the public application was created with.

In WorkSpace On-Demand environments, each program object related to a public application appearing in the user's desktop has the object ID **<NCID_***ApplicationId***>**, where *ApplicationId* is the name that the public application was created with.

Refer to About object IDs to get more information about object IDs.

Administering users and groups

Interactive administration

Batch administration

Users and groups interactive administration tool

The Users and groups interactive administration tool allows you to easily manage users and groups for your installation, assign security profiles and edit those once assigned.

If you ever want to invoke this utility from the command line, just enter :

```
EDYSNADM
```

The following pages briefly describe the generics of working with this application. Note, however, that a detailed description for each field can be found within the application's help itself.

There are several issues to take into account when working with this application :

The 'Lan Server specifics' panels will only display in IBM Lan Server installed environments.

In the same environment (Lan Server), all Lan Server groups will be displayed, but and due to the fact that groups and users are defined to UPM, you have to observe :

You can only assign SecureEntry components to SecureEntry groups (those that start with the letters 'SG').

You can not define first level users, that is, users not belonging to any group.

You can not define groups belonging to other groups.

You have more flexibility defining user/group relationships, in the sense that a given user can belong to more than one group.

The following pages give more detailed information of the different panels that the application provides:

Startup panel

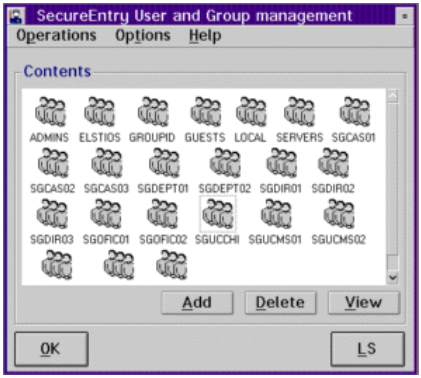
Group definition panel

User definition panel

Privilege handling

User exits

Startup panel



This panel displays the main groups and users defined within your users and groups database, that is, those defined at first level (They have no 'parent' group assigned). You can view the details of any user or group displayed here just by double-clicking on its icon.

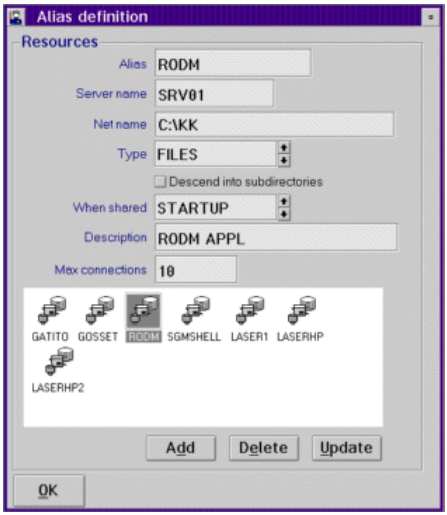
You can also add, view or delete a new group or user by selecting it if appropriate, and then pressing the appropriate button.

To add a new user associated with a given group, open the group panel by double-clicking on its group definition icon, and add the user from that panel. Note that if you are using UCM, adding of a new user requires also that you supply a 'branch name', i.e the branch that the new user belongs to.

There is an **'operations' menu** within this panel, which allows you, among other things, to directly find a user given his user id.

There is an **'options' menu** within this panel, which allows you to enable or disable displaying of warning messages that may be reported by the different subsystems.

The **'Lan Server' specifics button** can be used, under IBM Lan Server environments, to access Lan Server specific data. In this case, from the main panel, it can be used to define ALIAS definitions, as shown by the following panel :



In this panel you define the resources, i.e, Lan Server alias, that may be used by your defined users. Refer to the appropriate Lan Server documentation for the meaning of this fields, and also take into account the following :

If you do not give a server name for a given alias, the current server (where the SecureEntry definitions database reside) will be used.

You can drag and drop alias definitions from/into the resources container. An alias definition is a text file with the format required to define a resource through the batch administration tool. This file is also accepted as a drop source within the groups and users Lan Server specifics panel, to ease the assignment of such resources.

If you are using UCM, there is an **UCM menu** where you will be able to define the dynamic refresh feature for your corporate branches. If you select this menu you will then be able to choose between one of the three available options:

Branch Data

Through this panel you will be able to see the branch name as known by the UCM Database, the current SecureEntry branch information level and whether there was an error in the last dynamic refresh operation for this server. If you don't have the dynamic refresh feature activated, then you will receive an error.

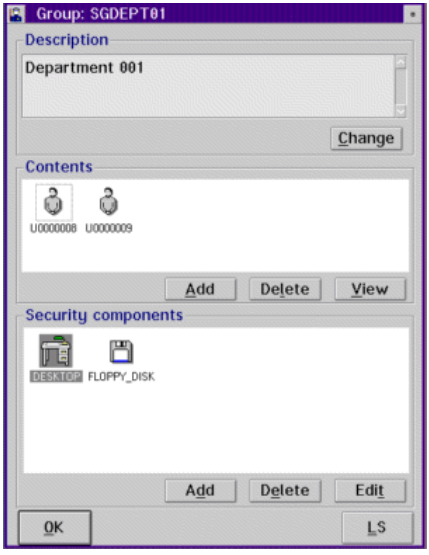
Policy Refresh

Through this panel you will be able to manage the Corporate refresh policy customized for all your branches. Also you can manage the threshold number of branches that will launch the purge process for obsolete data in the UCM tables once all of them are found to be at a minimum refresh level. In addition, you will be able to see the active refresh policy in the UCM administrator workstation (which can differ from the corporate refresh policy as customized in the UCM Database).

LOG Activity

Through this panel you will be able to activate the UCM LOG feature. If you activate this feature, then a new UCM LOG table will be updated by UCM whenever necessary.The only way to delete the obsolete rows in this table will be through the EDYEXLOG process as explained in the UCM Administrator's Guide.

Group definition panel



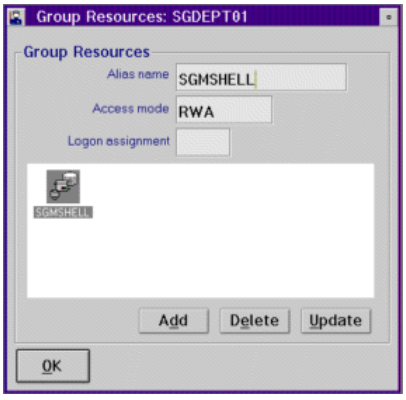
Within this panel, you can modify a group definition by adding other groups or users, or assigning SecureEntry security components for the group. There is also the description field which you can use to add a short description to a given group.

In order to add, view or delete users or groups, just click on the appropriate button.

To add a security component to the group, just drag and drop the file into the component's container.

You can also edit a security component by double-clicking on its icon.

The 'Lan Server specifics' button can be used here to define resource accesses at a group level, that is, allow users belonging to the group to be able to access the specified resources. The panel looks as follows :



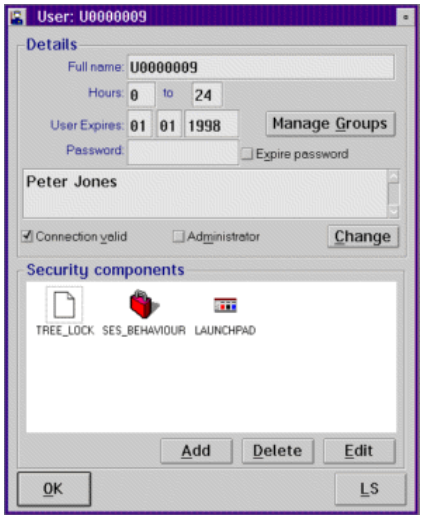
Refer to the appropriate Lan Server documentation for the meaning of the different fields displayed by this panel. Also take into account that :

Group logon assignment is not supported directly by Lan Server, but is a feature implemented and supported within SecureEntry. This basically means that you will not be able to see those from within the Lan Server administration tools.

Ensure that resources are available and powered on at administration time, or you will not be able to see their associations.

The container allows for drag and drop of definition files, with the same format as that required for batch administration.

User definition panel

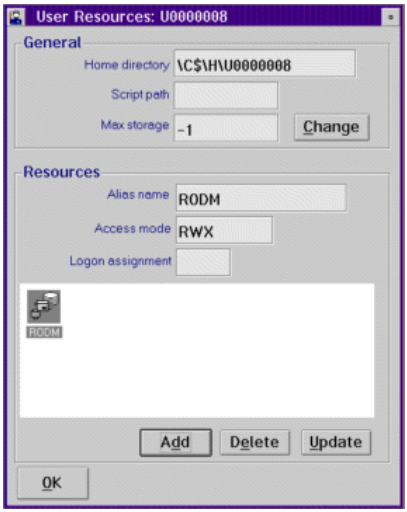


This panel is very similar to the one for groups definition, allowing you to change the specific data for the user, and optionally assign security components at a user level.

To add a security component to the user, just drag and drop the file into the component's container.

You can also edit a security component by double-clicking on its icon.

The 'Lan Server specifics' button can be used here to define resource accesses at a user level, that is, allow the user to access the specified resources. This panel looks like :



Refer to the appropriate Lan Server documentation for the meaning of the different fields displayed by this panel. Also take into account that :

SecureEntry extends the syntax of the home directory specification by allowing you to leave the server name (*MachineName*) blank, taking by default the current server name at assignment time. This is especially important when working with UCM, facilitating administration and management. Besides, it attempts to create the directory if not found, and does an instant access apply to all existing subbdirs when creating it. So, the syntax for the home directory becomes :

`x:\[MachineName]\y$\path`

or

`\\[MachineName]\y$\path`

where *x* is the drive assigned at logon to the home directory, *MachineName* is the server machine name where the home directory resides, (which you can leave blank to indicate the domain server machine name), *y* is the drive within the server, and *path* is the physical path within that drive.

Ensure that resources are available and powered on at administration time, or you will not be able to see their associations.

The container allows for drag and drop of definition files, with the same format as that required for batch administration.

Privilege handling

You can set permissions on single administrative operations by defining the administration specific environment variable **SGM_ADM_PRIV**. The users and groups interactive administration enforces this permissions by automatically disabling the controls that give access to operations to which rights are not held. Anyway, if the user succeeds on requesting an operation to which it has no rights, the operation will not be performed and the user will be prompted with a message box saying that he has not enough rights to perform the operation.

For instance, if you set the environment variable SGM_ADM_PRIV to 0xFFFF555F on a particular machine, you're configuring that machine to be able to administrate all existing groups, users and resources, but not to add new ones neither to delete existing ones.
Note: to be able to use the users and groups interactive application, the rights

EDYUCM_PRIVILEGE_SUB_VIEW
EDYUCM_PRIVILEGE_USER_GRP_VIEW
EDYUCM_PRIVILEGE_GRP_GRP_VIEW

must be enabled

User Exits

You can install administration user exits for the users and groups interactive administration (not for any other administration tool, but only this one) by filling the file **EDYADMUS.CMD** and placing it into SecureEntry installation path, EXEC subdirectory. An empty sample file is located within the API\SOURCES\EDYADMUS directory. The following user exit points are supported: :dthd. :ddhd.

UserExitBeforeADD_USER

Called before adding a user

UserExitBeforeADD_USERGROUP

Called before linking a user to a group

UserExitBeforeADD_GROUP

Called before adding a group

UserExitBeforeADD_GROUPGROUP

Called before linking a group to another group

UserExitBeforeADD_RESOURCE

Called before adding a resource

UserExitBeforeADD_RESOURCEUSER

Called before linking a resource to a user

UserExitBeforeADD_RESOURCEGROUP

Called before linking a resource to a group

UserExitBeforeADD_SUBSYSTEM

Called before adding a subsystem

UserExitBeforeUPDATE_USER

Called before updating a user data

UserExitBeforeUPDATE_USERGROUP

Called before updating a user-group link data

UserExitBeforeUPDATE_GROUP

Called before updating a group data

UserExitBeforeUPDATE_GROUPGROUP

Called before updating a group-group link data

UserExitBeforeUPDATE_RESOURCE

Called before updating a resource data

UserExitBeforeUPDATE_RESOURCEGROUP

Called before updating a resource-group link data

UserExitBeforeUPDATE_RESOURCEUSER

Called before updating a resource-user link data

UserExitBeforeUPDATE_SUBSYSTEM

Called before updating a subsystem data

UserExitBeforeDELETE_USER

Called before deleting a user

UserExitBeforeDELETE_USERGROUP

Called before deleting a user-group link

UserExitBeforeDELETE_GROUP

Called before deleting a group

UserExitBeforeDELETE_GROUPGROUP

Called before deleting a group-group link

UserExitBeforeDELETE_RESOURCE

Called before deleting a resource

UserExitBeforeDELETE_RESOURCEUSER

Called before deleting a resource-user link

UserExitBeforeDELETE_RESOURCEGROUP

Called before deleting a resource-group link

UserExitBeforeDELETE_SUBSYSTEM

Called before deleting a subsystem

These exit points allow you to insert any code just before the specified operation, being able to cancel it or change any desired parameters.

Please edit the edyadmus.cmd file to know more about the implementation guidelines in order to write your own administration user exits.

Users and groups batch administration tools

SecureEntry provides with tools to allow easily administration of its definitions in batch mode. This chapter describes all of them.

EDYDEFS is a program which will add, update or delete the database definitions as described within a supplied response file. This is the tool that you can find in the SecureEntry workbench folder named 'Batch Users and Groups management'

EDYERASE is a cleanup tool that allows you to selectively erase all or part of the definitions stored in the SecureEntry users and groups repository.

EDYDUMP is a tool that allows you extract part or all of the definitions stored in the SecureEntry users and groups repository, in a format suitable for later processing through the EDYDEFS utility.

EDYADMIN is a generic purpose administration tool that can help you in writing easily your own administration jobs.

All of this tools can be located within the SecureEntry path, EXEC directory, and require administrator privilege for its successful execution.

The definitions processing tool

The batch users and groups administration tool allows you to process administration tasks in an unattended way, by means of processing a batch job description file which specifies the actions to process.

The invocation command to process batch administration files is :

```
EDYDEFS Help | ([/]userid [[/]Password:p] [[/]File:f] [[/]Action:a]
        [[/Trace] [[/]Warn] [[/]Ignore'

Userid and password : Ignored. You should have logged on as  ADMIN
File   : Definitions file name. defaults to  EDYDEFS.TXT
Action : Startup action. defaults to ADD. Allowed values are
         ADD, UPDATE, DELETE
Trace  : Trace calls to display
Warn   : Do not process calls, just syntax check the file
Ignore : Ignore errors and continue
```

The best way to show the format of the definitions job file is through some sample files, as follows :

Sample definitions file for standalone environment.

```
// ********************************************************************
// * Sample batch file for Secure Entry standalone users administration *
// ********************************************************************

[ACTION=ADD]                          // Allowed action types are ADD, UPDATE
                                      // and DELETE. ACTION keyword means
                                      // sets the action until otherwise said
// Lets define one tellers group

[GROUP=TELLERS]
  FULL_NAME=Tellers  group
  DESCRIPTION=Group  for tellers
```

```
  DESKTOP=EDYDESK.INI               // Any standard SE component is allowed
  LAUNCHPAD=SELP.INI                // here : DESKTOP,  PERS_DESKTOP,  LAUNCHPAD
  FLOPPY_DISK=FPY.INI               //  SYSTEM_MENUS,  FLOPPY_DISK,  TREE_LOCK and
                                    //  SES_BEHAVIOUR
// Now define two users
[USER=PEPE]
  GROUP=TELLERS                     // The group should exist when adding user
  PRIV_USER=USER                    // Only values : USER or  ADMIN
  FULL_NAME=Pepito de los palotes
  DESCRIPTION=A user
  USER_EXPIRE=NEVER                 // NEVER or  dd-mm-yyyy format
  CONNECTION=1                      // 1=>  Signon permitted 0=> Not permitted
  HOUR_START=8                      // These hours are for all days of the week.
  HOUR_END=20                       // set to 0-24 for  Signon permitted anytime.
  PASSWORD=password                 // 'password' is the default  signon password
  LAUNCHPAD=SELP.INI                // Any standard SE component is allowed
  FLOPPY_DISK=FPY.INI               // here : DESKTOP,  PERS_DESKTOP,  LAUNCHPAD
                                    //  SYSTEM_MENUS,  FLOPPY_DISK,  TREE_LOCK and
[USER=MIGUEL]                       //  SES_BEHAVIOUR
  GROUP=TELLERS
  PRIV_USER=USER
  FULL_NAME=Miguel Indurain
  DESCRIPTION=A power user
  USER_EXPIRE=NEVER
  CONNECTION=1
  HOUR_START=8
  HOUR_END=20

// Now a managers group and a manager definition

[GROUP=MANAGERS]
  FULL_NAME=Managers group          // The rest of possible keywords will take
                                    // the default values

[USER=ANDRES]
  GROUP=MANAGERS
  PRIV_USER=USER                    // We could have used  GROUP_LS instead of
  FULL_NAME=Andy Garcia             //  PRIV_USER with the same meaning
  DESCRIPTION=A manager
  USER_EXPIRE=1-1-1999
  CONNECTION=1
  HOUR_START=8
  HOUR_END=20

// Now update Miguel adding a personal  launchpad, and move it to another group

[ACTION=UPDATE]
[USER=MIGUEL]
  GROUP=MANAGERS                    // Move user to group managers
  LAUNCHPAD=SELP.INI                // And add/modify  launchpad

// Now delete the  launchpad component for user  Miguel

[ACTION=DELETE]
[USER=MIGUEL]                       // Not a user deletion, but only component
  LAUNCHPAD                         // deletion because a keyword follows.

// Now delete completely the created groups and users

[USER=MIGUEL]                       // Full user delete
[USER=ANDRES]
[USER=PEPE]
[GROUP=TELLERS]                     // Group delete (must be empty)
```

```
[GROUP=MANAGERS]
```

Sample definitions file for Lan Server environment.

```
// ********************************************************************
// * Sample batch file for Secure Entry  Lan Server users administration *
// ********************************************************************

[ACTION=ADD]                          // Allowed action types are ADD, UPDATE
                                      // and DELETE. ACTION keyword means
                                      // sets the action until otherwise said
// Lets define one alias

[ALIAS=TSTALIAS]
  TYPE=FILES                          // Type of  resource.'FILES','SERIAL',
                                      //  'PRINTER' or 'TREE'
                                      // for a given alias definition
  NETNAME=C:\MAR\SGMSHELL             // This is the net name path
                                      // Can use  RESNAME as a synonym
  SERVER=SERVER1                      // The server of the alias without '\\'
  WHEN_SHARED=STARTUP                 // STARTUP,  BYADMIN or DYNAMIC
  DESCRIPTION=Alias  sample           // Put the description here
  MAX_CONN=45                         // Max. Connections

// Lets define one tellers group

[GROUP=SGTELLE]                       // Remember that Secure Entry group names
                                      // should start with 'SG'
  FULL_NAME=Tellers  group            // Under  Lan Server this is ignored
  DESCRIPTION=Group  for tellers      // Put the group description here
  ACCESS_FILE=ACCESS.DAT              // 'old way' of defining accesses
                                      // Sample access file :
                                      //  ***************************************
                                      // *  [ALIAS=TSTALIAS]  // aliasname        *
                                      // *    ACCESS=RWX      // access privileges*
                                      // *    LOGON_ASN=J:    // logon assignment *
                                      //  ***************************************
                                      // Note that even if you define an access for
                                      // update, it will be recreated so the full
                                      // specification has to appear here :
                                      // the  aliasname, the access rights and
                                      //  logon_asn.
  ACCESS=TSTALIAS,RWX,J               // 'new way' of defining accesses,  inlined.
                                      // Note that even if you define an access for
                                      // update, it will be recreated so the full
                                      // specification has to appear here :
                                      //  ACCESS=aliasname,accessrights,logon_asn.
  DESKTOP=EDYDESK.INI                 // Any standard SE component is allowed
  LAUNCHPAD=SELP.INI                  // here : DESKTOP,  PERS_DESKTOP,  LAUNCHPAD
  FLOPPY_DISK=FPY.INI                 //  SYSTEM_MENUS,  FLOPPY_DISK,  TREE_LOCK and
                                      //  SES_BEHAVIOUR
// Now define two users

[USER=PEPE]
  GROUP=SGTELLE                       // The group should exist when adding user
  PRIV_USER=USER                      // Only values : USER or  ADMIN
  FULL_NAME=Pepito de los palotes
  DESCRIPTION=A user
  USER_EXPIRE=NEVER                   // NEVER or  dd-mm-yyyy format
  CONNECTION=1                        // 1=>  Signon permitted 0=> Not permitted
  HOUR_START=8                        // These hours are for all days of the week.
  HOUR_END=20                         // set to 0-24 for  Signon permitted anytime.
  PASSWORD=password                   // 'password' is the default  signon password
```

```
  LAUNCHPAD=SELP.INI                // Any standard SE component is allowed
  FLOPPY_DISK=FPY.INI               // here : DESKTOP,  PERS_DESKTOP,  LAUNCHPAD
                                    //  SYSTEM_MENUS,  FLOPPY_DISK,  TREE_LOCK and
                                    //  SES_BEHAVIOUR

  // Follow keywords specific for  Lan Server

  HOME_DIR=X:\\c$\HOME\PEPE         // The  Lan Server home directory for this user
                                    // Use the syntax :
                                    //      \\MachineName\x$\path,  or
                                    //     x:\MachineName\x$\path
                                    //  MachineName  omitted means server machine
  MAX_STORAGE=2048000               // Defaults to -1 (no limit)
  SCRIPT_PATH=C:\SCRIPTS            //  Lan server script path
  ACCESS_FILE=ACCESS.DAT            // 'old way' of defining accesses
                                    // Sample access file :
                                    //  ***************************************
                                    // *  [ALIAS=TSTALIAS]   // aliasname        *
                                    // *    ACCESS=RWX       // access  privileges*
                                    // *    LOGON_ASN=J:     // logon assignment *
                                    //  ***************************************
                                    // Note that even if you define an access for
                                    // update, it will be recreated so the full
                                    // specification has to appear here :
                                    // the  aliasname,  the access rights and
                                    //  logon_asn.
  ACCESS=TSTALIAS,RWX,J:            // 'new way' of defining accesses,  inlined.
                                    // Format is  alias-name,access  privileges,
                                    //  logon-assignment
                                    // Privileges allowed :
                                    // D (Delete) A (Attributes) X (Execute)
                                    // R (Read) W (Write) P (Permissions) N (None)
[USER=MIGUEL]
  GROUP=SGTELLE
  PRIV_USER=USER
  FULL_NAME=Miguel  Indurain
  DESCRIPTION=A  power user
  USER_EXPIRE=NEVER
  CONNECTION=1
  HOUR_START=8
  HOUR_END=20

// Now a managers group and a manager definition

[GROUP=SGMANAG]
  DESCRIPTION=Managers  group       // The rest of possible keywords will take
                                    // the default values

[USER=ANDRES]
  GROUP=SGMANAG,OTHERGR             // LAN Server allows more than one group
                                    // Note that all groups should be specified
  PRIV_USER=USER                    // We could have used  GROUP_LS  instead of
  FULL_NAME=Andy  Garcia            //  PRIV_USER with the same meaning
  DESCRIPTION=A  manager
  USER_EXPIRE=1-1-1999
  CONNECTION=1
  HOUR_START=8
  HOUR_END=20

// Now update  Miguel adding a personal  launchpad,  and move it to another group

[ACTION=UPDATE]
[USER=MIGUEL]
```

```
  GROUP=SGMANAG                     // Move user to group managers
  LAUNCHPAD=SELP.INI                // And add/modify  launchpad

// Now delete the  launchpad component for user  Miguel

[ACTION=DELETE]
[USER=MIGUEL]                       // Not a user deletion, but only component
  LAUNCHPAD                         // deletion because a keyword follows.

// Now delete completely the created groups and users

[USER=MIGUEL]                       // Full user delete
[USER=ANDRES]
[USER=PEPE]
[GROUP=SGTELLE]                     // Group delete (must be empty)
[GROUP=SGMANAG]
[ALIAS=TSTALIAS]                    // Now delete the defined alias

// General notes :
//
// - Remember to specify all group names the user belongs to (except reserved ones)
//   when updating users. Reserved ones are  ADMINS, USERS and GUESTS
//
// - Remember that all Secure Entry group names have to start with 'SG'
//
// - Remember that update of accesses or components do only update the specified
//   ones through keywords, not erasing other components or accesses already
//   existent.
//
// - Also remember that deletion of main objects (groups, users, alias) is only
//   performed if no additional keywords are specified.
```

The definitions cleanup tool

The EDYERASE program allows you to easily do planned or unplanned cleanups of your users and groups database. The invocation syntax for this command is as follows :

```
EDYERASE [?|[/Help]|
         [[/Branch] [:branchname]] [[/]Groups]
         [[/]Ignoreerrors [[/]Local] [[/]NBbranch[:branchname]]
         [[/]Resources] [[/]Trace]  [[/]SGroups] [[/]Users]
         [[/]Usersgroup] [[/]Warning]
         ([[/]XUsers:mask]..) ([[/]XGroups:mask]..) ([[/XResources:mask]..)

  /Branch    : Erase all users belonging to a branch
  /Groups    : Erase all groups
  /Help      : Help
  /Ignore    : Ignore errors and continue
  /Local     : Local branch processing (ignored if no  UCM)
  /NBranch   : Erase all users not defined for the specified branch
  /Resources : Erase all resources except  SGMSHELL
  /Trace     : Trace on
  /SGroups   : Erase all  SecureEntry groups (all if standalone)
  /Users     : Erase all users except administrator ones
  /USERSGroup: Erase all users belonging to a group recursively
  /XGroups   : In any case, do not erase groups matching mask
  /XResources: In any case, do not erase resources matching mask
  /XUsers    : In any case, do not erase users matching mask
  /Warning   : Do not perform actions, just display

 Sample : EDYERASE /NB:BR03 /XU:9* /XU:THEBOSS
```

```
          Erases all users not belonging to branch  BR03 except
          users beginning with '9' and  THEBOSS user
```

Note that full wildcard support is provided for the **mask** type parameters.

The definitions dumper tool

The EDYDUMP utility allows you to extract resources, users and groups information from the users and groups repository, in a format which is suitable for later processing through the EDYDEFS utility. It will create the necessary batch file and download the appropriate security profiles as temporary files. The invocation syntax for this command is as follows:

```
EDYDUMP Help | [[/]File:f]  [[/]Password:p] [[/]Expire]
               [[/]Nofiles] [[/NOGroups] [[/]NOResources] [[/NOUsers]
              [[/]Trace]   [[/]Ignore]

          File    : Definitions file path and name to create
                    downloaded profiles will go to the same directory
          Password: Generate passwords for users. p may contain
                     %(.x)U for userid
                     %(.x)N for user number
                     %(.x)D for current day
                     %(.x)M for current month
                     %(.x)Y for current year
                    x indicates forced length. left padded with 0s
          Expire  : Add password expire keyword for all users
           Nofiles : Do not download profiles. Only  DEF data
           NOGroups: Do not extract groups information
       NOResources: Do not extract resources information (LS only)
          NOUsers : Do not extract users information
          Trace   : Trace calls to display
          Ignore  : Ignore errors and continue

     Example : EDYDUMP /P:U%.7N
```

Note that since the administration APIs do not return passwords, this is the only information you may find missing from the output batch file. To help overcome this problem, you can use the '/P' parameter, specifying a syntax for generating new passwords which are unique in a per user basis.

The generic administration tool command

The EDYADMIN program allows you to do any administration task through command line. Since this command is of a generic purpose, requires a good knowledge of the SecureEntry repository structure, and is not necessary for regular administration tasks. It is described in the following chapter under Programming your own administration tools.

Implementing SecureEntry solutions

In some occasions, what is provided with SecureEntry base may not be sufficient to implement the requirements that are necessary for a given installation. SecureEntry has an architecture which allows, with little effort, to achieve significant expandability in order to be adapted to a very heterogeneous set of needs.

You can basically expand SecureEntry in the following areas :

Programming to the SecureEntry user info API to obtain security information about the currently logged on user from within your application.

Programming user exits to perform required tasks in a per session event basis.

Programming your own naming filters to provide for special naming requirements within the signon process.

Adding your own components to be integrated as all SecureEntry base components are.

Programming your own logon procedures to grant access to your specific subsystems as part of the logon process.

Programming your own administration tools to do special tasks or processes in the administration area.

SecureEntry has a also a complete set of utilities to help you perform all of this and also aid you with your software integration effort. This includes several OS/2 native utilities,VDM utilities, plus a very sophisticated trace subsystem, which you can use to debug your solution, always with the handy help of the SecureEntry maintenance utilities.

The SecureEntry user info API

The SecureEntry kernel API allows you to obtain information about who is logged on, its privileges and other relevant session data. The include and library files are located in the SecureEntryPath\API\SOURCES\EDYAPI directory.

The provided functions are two :

```
unsigned long _System sgm_userinfo_getkey (char * key,
                                           char * data,
                                           unsigned long size,
                                           unsigned long reserved);
```

Allows you to get information in a per key basis.

```
unsigned long _System sgm_userinfo_get (char * buffer, long buflen,
                                        short level, unsigned long reserved);
```

Allows you to get user information in a structured way.

Note that return codes and valid structures are defined within the EDYAPI.H file.

Note also that other functions you may find here are maintained in the include file for compatibility reasons, but will not be honored by SecureEntry 3.0, since the profile management functions should go now through the new administration interfaces.

Programming user exits

When you want to do special processing in a per event basis, you can use SecureEntry user exits to do so. There are two ways to program your own user exits. Via REXX or a compiled language. In any case, you can find the source modules required within the SecureEntryPath\API\SOURCES\EDYCUST directory.

If you choose to program those in REXX, just fill in the provided EDYCUST.CMD file and place it within the SecureEntryPath\EXEC directory.

If coding the user exits in REXX, note that the performance degradation will be minimal, since the REXX user exits program is kept loaded as a function of the EDYSLA SecureEntry kernel. In this case, you have also limited support for global 'inter user exit' variables, as explained in the EDYCUST.CMD file.

If you code these in C, the filled and compiled file EDYCUST.DLL must be placed within the SecureEntryPath\DLL directory.

The user exits provided are :

User exit after startup:

Intended as a 'second opportunity' to the startup processing, allows you to start processes in a symmetrical way to the user exit before shutdown. Runs under superuser context.

User exit before logon dialog:

This one gives the opportunity to do some processing and optionally decide to continue with a guest logon before the SecureEntry logon dialog starts interacting with the user, i.e, a guest logon does not require to fill up the userid/password dialog. Runs under superuser context. You can optionally present your own logon dialog here, but if you do so, then your application dialog has to be system modal, and will have to call EDYUTIL with the appropriate NEXTLOGON parameters to avoid the SecureEntry LOGON dialog appearing afterwards. In any case, do not forget to return the system modality to the previous owner once your dialog is destroyed.

You can also decide to shutdown the system when exiting from this user exit.

User exit before logon:

This user exit can be used to decide on guest logon processing after the user has filled up the logon dialog. It can be used to implement 'service' logons with algorithmic passwords provided by a central site. Runs under superuser context.

You can also decide to shutdown the system when exiting from this user exit.

User exit before logon changing password:

Same as the user exit before logon, but receiving the new password also. Runs under superuser context.

User exit after logon changing password:

This user exit is intended to inform your applications that the password has been changed, if required. Runs in user context.

User exit before profiles activation:

This user exit is processed at logon, once the security profiles have been downloaded to the workstation's *WORK* directory. You can use it to modify/copy those profiles and do any preprocessing that may be necessary before they are activated.

User exit after logon:

Used to do specific processes after any logon. Note that presentation manager and Workplace shell are ready when this user exit is called. Also note that this user exit is called even in the guest logon case and after any shell (PMSHELL) restart. Runs under user context.

User exit before cancellable logoff:

This user exit gives the opportunity to verify and optionally deny the logoff event, if for any reason it is detected that it can not be processes due to application requirements. Runs under user context.

User exit before imminent logoff:

This user exit's purpose is to allow closing of critical applications, having granted the fact that the logoff event will be processed until the end. Runs under the logged on user context.

User exit after profiles deactivation:

This user exit is processed at logoff, once the security profiles have been deactivated at the workstation's. You can use it to modify/copy those profiles and do any postprocessing that may be necessary before they are uploaded.

User exit after logoff:

This user exit is intended to do any required cleanup necessary after a user logoff in superuser context.

User exit before shutdown:

This user exit allows for doing final cleanup of the machine before the shutdown event completes. Runs under superuser context.

User exit after pmshell:

This exit is given to inform whoever requires it that the Workplace Shell is active and running. Note that this one is special in the sense that nothing else is granted here about logged on users.

User exit after startup:

This exit is given to inform whoever requires it that the EDYSTART.CMD file has finished processing. You can use this user exit to start any process that you may want to somehow 'hide' from the EDYSTART.CMD file.

User exit before cancellable lockup:

This exit is given to inform whoever requires it that a lock event has been posted, and to give you the ability to programmatically cancel it, or prefill the parameter area with the userid/password to use.

User exit lockup:

This exit is given to inform whoever requires it that an lock event, which can not be cancelled, is about to be processed, in order to start or terminate any required task.

User exit before unlock dialog:

This exit is given to inform whoever requires it that an unlock event is about to be processed, to give the opportunity to display your own panel, or prefill/read/validate the password. If you do present your own dialog, then your dialog has to be system modal, and your application will be responsible for calling EDYUTIL with the appropriate NEXTUNLOCK parameters to avoid the UNLOCK dialog to be presented afterwards. In any case, do not forget to return the system modality to the previous owner once your dialog is destroyed.

You can also decide to logoff or shutdown the system when exiting from this user exit.

User exit before unlock:

This exit is given to inform whoever requires it that, once the unlock password has been obtained, it is going to be validated against the current SecureEntry validation password for the session. You can validate it yourself at this point.

You can also decide to logoff or shutdown the system when exiting from this user exit.

User exit after unsuccessful unlock:

This is the last chance before an invalid password processing follows its regular flow to inform the user at unlock time. You can use it to modify the behavior, accepting the password, blocking the machine, forcing a logoff or a shutdown.

User exit after unlock:

This exit is given to start/terminate any required process when an unlock event is finishing.

User exit before logoff from unlock dialog: This user exit allows for a programmed authorization of logoff from the unlock dialog, in case this option is enabled through the currently active SES behavior profile.

User exit before shutdown from unlock dialog: This user exit allows for a programmed authorization of shutdown from the unlock dialog, in case this option is enabled through the currently active SES behavior profile.

Signal user exit: This user exit allows to receive parameters from the application code within the user exit context, to perform specific actions under the user exit's code environment. The Session event launcher : EDYUTIL utility can be used to launch one of this user exits.

SupeUser signal user exit: This user exit allows to receive parameters from the application code within the user exit context, to perform specific actions knowing that they will run under super user context and within the user exit's code environment. Note that if you are not using the real Security Enabling Services, then all processes started within this user exit will run under regular user context, and not super user context as explained above. The Session event launcher : EDYUTIL utility can be used to launch one of this user exits.

LMP user exits: These user exits allow you to change the normal control flow of the logon procedures by examining the ret. code about to be returned, and changing it appropriately. With this user exits you can, for instance, avoid users to logon in emergency mode, or force an algorithmic password for specific users.

Folder open and close user exits:

This user exits are invoked whenever an object with a defined hook has been configured to call them, at open or close time. The user exit for the open object event may decide not to open the object by returning a specific return code. You can read more about hooked objects in The Hooked objects component chapter.

User exit hotkey:

If the active SES profile has defined hooks for systemwide hotkeys, then this user exit will be triggered each time one of the hooked key combination is detected. You can then do your logic, and choose to either process the hotkey yourself, or pass it along so that the default action is then performed.

User exit before reboot:

You can use this user exit as a signal that reboot is to be performed by SecureEntry, either from within the Ctl-Alt-Del options dialog, or as a Ctl-Alt-Del user request when the configured action for this key combination is to directly reboot the system.

For further information, please look at the User exits control flow, and read the appropriate fully commented source file.

A Sample user exit implementation is also available for review, as well as two sample logon and unlock dialogs.

User exits control flow

The following list shows the regular flow (calling sequence) for the different available user exits.

```
* Machine startup and  logon

UserExitAfterStartup
UserExitBeforeLogonDialog
UserExitBeforeLogon
UserExitBeforeLMPSignon              Note : one received per  LMP
UserExitAfterLMPSignon
UserExitAfterPmShell                 Note : Asynchronous (not granted timing)
UserExitBeforeProfilesActivation
UserExitAfterLogon

* Regular lock-unlock

UserExitBeforeCancellableLockup
UserExitLockup
UserExitBeforeUnlockDialog
UserExitBeforeUnlock
UserExitAfterUnlock

* Regular shutdown from desktop of logged on user

UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff
UserExitAfterProfilesDeactivation
UserExitBeforeLMPSignoff              Note : one received per  LMP
UserExitAfterLMPSignoff
UserExitAfterLogoff
UserExitBeforeShutdown

* Machine startup and regular  logon changing password

UserExitAfterStartup
UserExitBeforeLogonDialog
```

```
UserExitAfterPmShell                Note : Asynchronous (not granted timing)
UserExitBeforeLogonChangingPassword
UserExitBeforeLMPSignon             Note : one received per  LMP
UserExitAfterLMPSignon
UserExitBeforeProfilesActivation
UserExitAfterLogonChangingPassword

* Regular lockup -  logoff from unlock dialog

UserExitBeforeCancellableLockup
UserExitLockup
UserExitBeforeUnlockDialog
UserExitBeforeLogoffFromUnlock
UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff
UserExitAfterProfilesDeactivation
UserExitBeforeLMPSignoff             Note : one received per  LMP
UserExitAfterLMPSignoff
UserExitAfterLogoff

* Regular Shutdown from  logon panel

UserExitBeforeLogonDialog
UserExitBeforeShutdown

* Machine startup and  logon

UserExitAfterStartup
UserExitBeforeLogonDialog
UserExitBeforeLogon
UserExitAfterPmShell                Note : Asynchronous (not granted timing)
UserExitBeforeLMPSignon             Note : one received per  LMP
UserExitAfterLMPSignon
UserExitBeforeProfilesActivation
UserExitAfterLogon

* Lockup - unlock fail due to invalid password - Retry and successful unlock

UserExitBeforeCancellableLockup
UserExitLockup
UserExitBeforeUnlockDialog
UserExitBeforeUnlock
UserExitAfterUnsuccessfulUnlock
UserExitBeforeUnlock
UserExitAfterUnlock

* Regular  logoff from desktop of logged on user

UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff
UserExitAfterProfilesDeactivation
UserExitBeforeLMPSignoff            Note : one received per  LMP
UserExitAfterLMPSignoff
UserExitAfterLogoff

* Regular  logon changing password

UserExitBeforeLogonDialog
UserExitBeforeLogonChangingPassword
UserExitBeforeLMPSignon             Note : one received per  LMP
UserExitAfterLMPSignon
UserExitBeforeProfilesActivation
UserExitAfterLogonChangingPassword
```

```
* Regular Logoff from desktop, cancelled

UserExitBeforeCancellableLogoff

* Regular logoff from desktop

UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff
UserExitAfterProfilesDeactivation
UserExitBeforeLMPSignoff              Note : one received per  LMP
UserExitAfterLMPSignoff
UserExitAfterLogoff

* Forcing Guest  logon from User exit before  logon

UserExitBeforeLogonDialog
UserExitBeforeLogon                   Note : Returning  RC_GUEST_LOGON
UserExitAfterLogon

* Regular logoff from desktop of guest user

UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff

* Forcing Guest  logon from user exit before  logon dialog

UserExitBeforeLogonDialog             Note : Returning  RC_GUEST_LOGON
UserExitAfterLogon

* Regular logoff from desktop of guest user

UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff

* Regular logon

UserExitBeforeLogonDialog
UserExitBeforeLogon
UserExitBeforeLMPSignon               Note : one received per  LMP
UserExitAfterLMPSignon
UserExitBeforeProfilesActivation
UserExitAfterLogon

* Lock and then Shutdown from unlock dialog

UserExitBeforeCancellableLockup
UserExitLockup
UserExitBeforeUnlockDialog
UserExitBeforeShutdownFromUnlock
UserExitBeforeCancellableLogoff
UserExitBeforeImminentLogoff
UserExitAfterProfilesDeactivation
UserExitBeforeLMPSignoff              Note : one received per  LMP
UserExitAfterLMPSignoff
UserExitAfterLogoff
UserExitBeforeShutdown
```

Sample user exit implementation

To clarify things a bit, lets suppose the following scenario :

We have an already written application which does handle logon and logoff, and wish to integrate SecureEntry inside this environment, taking profit of the lockup, ctl-alt-del handling, and security profile activation features of SecureEntry. Lets assume also that the written application has the following logic :

```
Do
  Query UserID and Password
  Validate_user
  if not privileged then
     run teller application
Until privileged
```

We also want a 'backdoor' to be able to logon as a real SecureEntry administrator.

It seems clear now that we have three types of users :

Regular 'tellers' which a set of restricted security profiles

Service people who have limited access to system resources

SecureEntry administrators

One of the easiest ways to implement such a scenario, taking into account that userid/password query is normally processed by the application, is through user exits. Lets describe the proposal :

Normally, all logons will be as GUEST, except when logging as a SecureEntry administrator. This means that SecureEntry dialogs for logon will never be displayed except when the backdoor is called.

The default teller security profiles will be kept as the default ones, and stored in the NOUSER directory.

The privileged security profiles for the service people will be also in the NOUSER directory, but activated/deactivated in an as required basis (p*.ini).

The 'backdoor' real SecureEntry logon will be implemented as a signal passed parameter.

Just for the beauty of it, (and to show how to do it), we will start the system clock for real SecureEntry users.

Now, the required EDYCUST file changes are as follows :

1) Lets declare a global variable

```
Line 158 :  MyGlobalVars='aString'
```

2) UserExitAfterStartup:

```
/*
   We can do one time user exit initialization code here, since
   not counting the user exit after  PMSHELL, this one will be the
   first one
*/

/* To be able to open the clock  */
call RxFuncAdd 'SysLoadFuncs', 'REXXUTIL', 'SysLoadFuncs'
call SysLoadFuncs
```

```
      /* Make sure first  logon is regular */
      aString=''

   return RC_NONE

3) UserExitBeforeLogonDialog:

   /* If not using the backdoor, always user GUEST  LOGON */

   if strip(translate(aString))<>'SENTRY'  then
      return  RC_GUEST_LOGON

   return RC_CONTINUE_NORMAL_LOGON_FLOW

4) UserExitAfterLogon:

      UserID=Reserved1

      if strip(translate(aString))<>'SENTRY'  then
      do

        /* Regular logon....                    */
        /* This should be started or detached, since has to run
           within the user session, not inside a user exit       */

        'start /C  d:\sgmshell\exec\apprun.cmd'
      end
      else
      do
        /* Backdoor logon....                   */
        /* Following time will be regular  logon */
        /* But for this time as a sample lets open the clock object   */

        aString=''
        rc=SysOpenObject('<WP_CLOCK>','DEFAULT',1)
      end

   return RC_NONE



5) UserExitBeforeImminentLogoff:

      if strip(translate(aString))='PRIV'  then
      do
        /* Deactivate the privileged profiles by
           activating  NOUSER standard ones       */

        'd:\sgmshell\exec\edyrefr.exe  d:\sgmshell\nouser\edydesk.ini'
        'd:\sgmshell\exec\edyflpy.exe  /I:d:\sgmshell\nouser\edyflopp.ini'
        aString=''
      end
      /* Note that if sentry backdoor  aString value is maintained 'SENTRY'  */

   return RC_NONE

6) UserExitSignal:

      aString=AllParms

      if strip(translate(aString))='PRIV'  then
      do
        /* Activate the privileged profiles  */
```

```
          'd:\sgmshell\exec\edyrefr.exe  d:\sgmshell\nouser\pdesk.ini'
          'd:\sgmshell\exec\edyflpy.exe  /I:d:\sgmshell\nouser\pflopp.ini'
        end
        /* If backdoor to sentry then force  logoff */
        if strip(translate(aString))='SENTRY'  then
          'edyutil frlogoff'

    return RC_NONE
```

And the application code (APPRUN.CMD) would have to have the following logic change :

```
    Do
      Query UserID and Password
      Validate_user

      RUN EDYUTIL NEXTUNLOCK /U:UserID /V:password

      if not privileged then
         run teller application
    Until privileged

    RUN EDYUTIL SIGNAL PRIV
```

And at any time, calling EDYUTIL SIGNAL SENTRY would force the backdoor to run, making the next logon a real SecureEntry one.

Note how easy is it, what a huge behavior change achieved with just 20 lines of REXX code added!!.

Sample logon and unlock dialogs

The user exits allow to easily insert logon and unlock dialogs created by the user, as an alternative to the default dialogs that SecureEntry provides.

You can find two fully customable alternative logon and unlock dialogs in the EXEC directory of the SecureEntry path. The source files of both dialogs are available in the API\SOURCES\DIALOGS directory of the SecureEntry path. This are two ZIP files which contain the source tree structure of a VisproRexx project.

The logon dialog accepts the following syntax:

```
 LOGSAMP [/D:domain | /d:[domain]]
         [/U:userid /L:file.bmp /S:string /NG /NC /Ns /NS
          /NH  /P:px,py /T /F
```

/D:

to specify the default Domain for the next logon.

/d:

to specify the default Domain, and to show an entry field that allows for modifying this domain.

/U:

to specify the default Userid that must be shown in the userid entry field.

/L:

to specify the file having the logo that must be shown in the dialog. If not specified, it defaults to the IBM logo.

/S:

to specify the header string that must be shown in the logon dialog. If not specified, it defaults to the string "IBM Global Services".

/NG

to hide the Guest button.

/NC

to hide the Clear button.

/Ns

to hide the Shutdown button.

/NS

to hide the Forced Shutdown button.

/NH

to hide the Help button.

/P:

to specify the logon dialog position in the screen. Horizontal coordinates (px) and vertical coordinates (py) have their origin in the left bottom corner of the screen and their values range from 0 to 100, both included. The default position is at the center of the screen.

/T

to show the Clock.

/F

to show the Date.

The unlock dialog accepts the following syntax:

```
UNSAMP [/L:file.bmp /S:string /Nl /NL /Ns /NS
        /NH /P:px,py /T /F]
```

/L:

to specify the file having the logo that must be shown in the dialog. If not specified, it defaults to the IBM logo.

/S:

to specify the header string that must be shown in the logon dialog. If not specified, it defaults to the string "IBM Global Services".

/Nl

to hide the Logoff button.

/NL

to hide the Forced Logoff button.

/Ns

to hide the Shutdown button.

/NS

to hide the Forced Shutdown button.

/NH

to hide the Help button.

/P:

to specify the unlock dialog position in the screen. Horizontal coordinates (px) and vertical coordinates (py) have their origin in the left bottom corner of the screen and their values range from 0 to 100, both included. The default position is at the center of the screen.

/T

to show the Clock.

/F

to show the Date.

For example, if you want to use the alternative logon dialog having the following features:

with a modifiable default domain whose name is "domain"

without the Guest button

without the Forced Shutdown button

with Clock and Date

then you should call LOGSAMP in the user exit before the logon dialog in the EDYCUST.CMD file this way:

```
UserExitBeforeLogonDialog:
       /*  Call the alternative  logon dialog */
        'LOGSAMP /d:domain /NG/NS/F/T'
       /*  Return the  RC supplied by  LOGSAMP */
         return  RC
```

Similarly, you can call the alternative unlock dialog the same way in the user exit before unlock dialog.

Also, you may want to read about the **SGM_USER_DLGS** environment variable, to achieve maximum robustness in overriding the default logon/unlock dialogs.

Programming naming filters

The file EDYFILT.DLL can be used to provide for automatic translation of userid's, passwords and/or domain names. It is intended to allow for integrating different subsystem naming conventions.

The skeleton file to program a filter DLL can be found within the SecureEntryPath\API\SOURCES\EDYFILT path, and if filled and compiled and then placed within the DLL directory, it will be called by the different logon subsystems before using the passed arguments. For instance, you may decide to have all your passwords scrambled somehow to the Lan Server, or to preppend a given letter to all SecureEntry users, without requiring the user to type it in the logon dialog. Then this is the place to put the code.

Note however that any translation you do here is only used at signon time, and that the administration tools should use the 'real' defined names.

Adding your own components

The same way that SecureEntry provides with some default components (i.e, Launchpad, Treelock, ...), you can add to the SecureEntry database a given component of your own, and assign it to a user or group through the SecureEntry administration tools.

What is a Security component

The components description file and UPDATEDB command

IMPORTANT WARNINGS

What is a Security component

A security component is really an object (file) with some behavior expected and associated characteristics :

1. A component should have a component name. This one is evident.
2. A component has a download file name. This is the file name of a given instance of security profile for the component which will be used at signon time to activate it.
3. A component has an editor. This is the name and calling parameters of the program which is able to recognize and allow editing for a given profile of this component.
4. A component has an activation interface. Again, the this is the name and calling parameters of an executable module which is able to read a profile and set up the security restrictions for it within a given machine.
5. A component has also an icon which represents it, and
6. It has also some rules, i.e if it is static (the profiles only vary through an administrator tool), or if must be uploaded to the database after logoff, plus whether it is mandatory to have a given profile assigned in order to be able to logon to the workstation.

The components description file and UPDATEDB command

All of the components description characteristics are stored within the SecureEntry repository as a file which you can change or view through the UPDATEDB command. This command will normally take a description file in ASCII format and upload it to the repository, making it the active one (for the next machine reboot). The invocation syntax is :

```
UPDATEDB  descriptionfilename
```

If no descriptionfilename is used, the utility will just list the active component table contents (stored within the repository).

Note that the description file used at SecureEntry install time is named 'SENTRY.DSC', and you can find it within the first installation diskette. This file can serve you as a sample on how new components can be defined.

A sample of the description file follows :

```
[General]

MaxSignonAttempts   5       # Beware if you have only one administrator defined
MinPasswordLen      4
PasswordExpireDays  0       # 0 means do not expire passwords

[Components]

#CompName        DownloadName  Editor             Interface           Datatype Icon  Mand Chg Description
#-------------------------------------------------------------------------------------------------------------
---
DESKTOP          EDYDESK.INI   'EDYEDRES.EXE &'  $SENTRY             MISSING  @3003   N   A  Personal desktop
profile
PERS_DESKTOP     EDY_PER.INI   MISSING            $SENTRY             MISSING   MISSING N   U  Personal desktop
settings
SYSTEM_MENUS     EDYWIN.INI    'EDYWINE.EXE &'    'EDYWINR.EXE /F:&' MISSING  @3006   N   A  Window behavior
FLOPPY_DISK      EDYFLOPP.INI  'EDYFLINI.EXE /&' 'EDYFLPY.EXE /I:&' MISSING  @3005   N   A  Floppy disk behavior
TREE_LOCK        EDYDD32.INI   'E.EXE &'         'EDYDDUTL.EXE /F&' MISSING   MISSING N   A
SES_BEHAVIOUR    EDYSES.INI    'EDYSESE.EXE &'   'EDYBGINI.EXE /F&' MISSING  @3007   N   A  Lockup,  timeouts and
CAD
TLOCK_AUDIT      EDYDD32.AUD   'E.EXE &'         'EDYDDUTL.EXE /A&' MISSING   MISSING N   U  Tree Lock audit
LAUNCHPAD        EDYPAD.INI  '/I EDYLNEDT.EXE /F&''EDYLNREF.EXE /F&' MISSING  @3004   N   A  Personal  launchpad
SCENTER          EDYSC.INI   '/I EDYSCEDT.EXE &'  'EDYSCTST.EXE &'   MISSING  @3017   N   A  SmartCenter profile
WINDOW_LIST      EDYWDLST.INI    'EDYWDEDT.EXE &' 'EDYWDINI.EXE /F&' MISSING  @3011   N   A  WindowList behaviour
PERS_MODEL       EDYMPER.INI     'EDYEDPER.EXE &' $SENTRY             MISSING  @3013   N   A  Personal desktop
model
SHORTCUTS        EDYHOTK.INI     'EDYHOTKE.EXE &' 'EDYHOTKR.EXE /F &'MISSING  @3014   N   A  Shortcuts profile
HOOKED_OBJECTS   EDYCUSWP.INI 'EDYEDPER.EXE /P &'$SENTRY             MISSING  @3016   N   A  Hooked objects
ROAMING_DESKTOP  EDYROAM.INI     'EDYROAME.EXE &' 'EDYROAMT.EXE /F&' MISSING  @3018   N   A  Roaming desktops
EXECS_AUDITOR    EDYEXEC.INI     'EDYEXECE.EXE &' 'EDYEXECT.EXE /F&' MISSING  @3019   N   U  Processes auditor
```

It becomes evident by studying this file how the new components have to be added. Take into account the following :

The [General] section, even if present, will have only sense for standalone environment installations, because this values will be overridden by Lan Server in Lan Server installs.

Any line can have comments, which start by the '#' character

The reserved keyword 'MISSING' allows you to define component characteristics as not available.

The component name will be used by the administration utilities to identify new components through its API. i.e, once you define your component with a special name, this name becomes the index keyword to locate its profiles. It also becomes automatically a new keyword for the batch administration tools.

The DownloadName will be used as the name with which the profile will be downloaded within the workstation at logon time, within the SecureEntry directory path, WORK subdirectory.

The editor must be any executable program which 'Understands' your profiles and knows how to save them with the same name as when invoked. (i.e, you can use the system editor for ASCII profiles). The launch string, as placed in the description file describes how to invoke it (substituting '&' by the full path of the profile component file).

The interface must be any executable program that knows how to 'activate' a given profile with the configured startup string (substituting '&' by the full path of the profile component file). Note however, that SecureEntry will launch this startup string at logoff time without a filename, and it is the responsibility of the interface program to deactivate then the previously activated profile. The reserved keyword '$SENTRY' is reserved by SecureEntry.

Datatype is not used by SecureEntry right now.

Iconfile is the icon filename of an icon that represents the component. Note that you should place here the full path for the icon. What is in the default sample file (keywords start by '@') is a convention so that these are icons already linked with the administration (EDYSNADM) application.

The 'Mand' field indicates whether the component is mandatory for logon or not. It can only take the values 'Y' or 'N'.

The 'Chng' field indicates whether it is a component profile which the user can change or the administrator. Values are 'A' (Admin) or 'U' (User). If a component is defined as user modifiable, then it will be uploaded at logoff time to the LAN repository, or to the host if UCM is present, so that at the next logon (from this or another machine), the refreshed profile is downloaded.

Description is a free format description string for the component.

No two components can have the same component name or downloadname.

IMPORTANT WARNINGS

When designing for adding new components, take care of :

The UPDATEDB program will overwrite completely the active components description table. It is your responsibility to make sure that you do not delete components which have active profiles stored in the repository.

The component profiles may have a 'signature' added by the interactive administration tool. This signature is :

For text files, a first line added to the file with the ComponentName.

For INI (prf format) files, a new tuple Application=SENTRY, Key=COMPONENT, Value=Componentname.

This signature is added only by this tool to allow for drag and drop without being always prompting about which component is being dropped. Note that to avoid this behavior, you can always uncheck the 'save

component association checkbox' whenever dropping a component profile and being prompted for its association within the interactive administration tool.

Be careful when applying SERVICE to machines with customer defined components. Before using the diskettes, copy your components description file to an INSTALL directory within the last installation diskette, with file name 'SENTRY.DSC', so that the service procedure uses this one instead of the default one to update the database.

Programming logon procedures

A logon procedure is basically a DLL with a couple of standard entry points which you can add to a SecureEntry installation, providing for password synchronization and the basic identification functions. The idea is that in order to make SecureEntry logon to your own subsystems in a fully integrated way, you just have to write this quite simple code.

Writing and running your own LMP

API entry points

Rules to follow

Writing and running your own LMP

Within the SecureEntryPath\API\SOURCES\LMP directory, you have a skeleton of a logon procedure in order to do signon to your own subsystems.

Once that file is coded, compile it and generate a DLL named EDYxxLMP.DLL (do not forget to change the DEF file reflecting this archive name change), where xx are your sybsystem identifier letters, then place the code in the DLL directory, and modify the file SecureEntryPath\NOUSER\EDYSSLMP.DAT, so that it calls your DLL at signon time. Refer to Configuration files, where this configuration file is described.

API entry points

Follows a description of the entry points available for LMPs.

EdySignOn

In this function you are supposed to do the signon to your subsystem, activating any subsystem specific resource configured to be accessible at signon time (i.e, logon assignments for Lan Server), and take note of the userid, password and domain (if required) of the signed on user for further requests. Note that replying OK to this function implies that you validate and grant access to the passed on user with the passed on password. SecureEntry will not call you to do a second signon while a user is already in signon state. (The sign-off API will be called first).

This API is called in your order of appearance within the EDYSSLMP.DAT configuration file, with respect to other LMPs.

EdySignOff

In this function you have to issue a subsystem sign-off, reset your variables and take note that from now on your subsystem is in a signed off state.

EdyForcePassword

This API will only be called if you are being synchronized by other LMP, i.e, RACF or similar. It means that you have to force the password for the passed userid to the newly provided one. Note that if you use the supplied skeleton, it has implemented a security mechanism by which only SecureEntry in a safe state (id granted) can call this API.

EdyQueryUserid

Within this API you are requested to reply the signon parameters as they were passed to you, in order to know who is signed on for your subsystem.

EdyPwSyntaxValidation

In this entry point you are supposed to verify the syntax of a given password, that is, if the password would be accepted as valid for a user in a change password operation for your subsystem.

EdyIdSyntaxValidation

In this entry point you are supposed to verify the syntax of a given userid, that is, if the userid would be accepted as valid for a user in a change password operation for your subsystem.

EdyUserPwValidation

This API will be used by SecureEntry to verify the userid and password of a given user without issuing a real signon, just replying if the tuple is valid for your subsystem. This API may be called to you by SecureEntry in the name of end user applications.

Rules to follow

When writing LMPs, take note of the following :

All of the entry points are mandatory, except the force password one. This one is only used if your DLL is synchronized by another one. If yours is the synchronizer, you can leave this entry point empty since it will never be called.

Use the skeleton file EDYxxLMP.C. It provides with security measures by which only SecureEntry in a granted state can call the 'potentially risky' APIs.

Do not change the input parameters you are passed with by reference.

Always return with errors that are defined in the EDYERROR.TXT, or fill up the error string if providing new errors.

Do not store pointers to passed parameter across calls. I.e, the buffer you are passed with the userid may not be valid after you have replied to the signon call.

Programming your own administration tools

There are currently three different levels of administration equivalent API's to do your own development :

The Command line API (EDYADMIN program)

The REXX API

The 'C' API

All of these interfaces deal with the same objects, and you may wish to use the appropriate one depending on your coding experience and requirements.

As an introduction to the API, and at a generic level, there are four main functions (add, delete, update and view) defined over a set of 7 different object types :

User object which stores the characteristics (keywords) of a given user. These are the user logon data and security profiles.

Group object which stores the characteristics (keywords) of a given group. This are the group logon data and security profiles.

Resource object which stores the characteristics of a given resource. This one is only used under LAN SERVER environments, and refers to shareable resources with associated ALIAS names.

UserGroup object which stores a ownership relation between a user and a group, and has no other attributes.

GroupGroup object which stores a ownership relation between a group and another group, and has no other attributes. This object type is only used in standalone environments, where you can define one group as belonging to another one.

ResourceUser object which stores a 'uses' relation between a resource and a user. The associated keywords for this relation are the logon assignment and sharing mode.

ResourceGroup object which stores a 'uses' relation between a resource and a user. The associated keywords for this relation are the logon assignment and sharing mode.

So, for instance, if you wanted to add a new user to the database, you should do :

1. Execute a User object ADD call passing the necessary keywords to describe the security profiles and user data (LAN_DATA). The user data keyword describes characteristics such as user logon hours, description, password,...
2. Execute a UserGroup object ADD call to make the user part of all necessary groups (one call for each desired group relationship).
3. Execute a ResourceUser object ADD call to make the user use the required resources (one call for each desired resource relationship).

The EDYADMIN utility

This utility, located in the SecureEntryPath, EXEC directory can be used to call the API from batch files without any restriction, allowing full usage of the administration API.

The syntax for this command is :

```
EDYADMIN [S=subsystem] Command Objecttype=objectid[,objectid2]  [keyword]
```

Subsystem

Name of the agent to process it. You will normally not use this option if working with the default SecureEntry subsystem (this option is intended for developing of customer SecureEntry subsystem agents).

Command

A valid command name. These are :

Add

Delete

Update

View

Objecttype

A valid object type name. These are :

Group

GROUPGroup

Resource

RESOURCEGroup

RESOURCEUser

Subsystem

User

USERGroup

Objectid[,objectid2]

Valid object(s) name. Place them separated by a comma for object types that deal with more than one object id.

Keyword

Valid keywords string list (separated by commas). For instance,

```
LAUNCHPAD=@myfile1,LAN_DATA=DESCRIPTION=A  user
```

Note that the keyword id is always the name to the left of the first equal sign per each keyword string. You can indirect a value by using the @ sign.

Usage Samples :

```
EDYADMIN A U=JOHN
                  Adds user John with default values
EDYADMIN A USERG=JOHN,TELLERS
                  Adds link of user John to TELLERS group
EDYADMIN U U=JOHN DESKTOP=@EDYDESK.INI
                  Updates Johns desktop with file
EDYADMIN U U=JOHN LAN_DATA=FULL_NAME=John  Smith DESCRIPTION=A teller
                  Updates Johns name
EDYADMIN V USERG=*,*
```

```
                  View all  userids belonging to any group
EDYADMIN V USERG=*,TELLERS
                  View all  userids belonging to group TELLERS
EDYADMIN V USERG=*,
                  View all users not belonging to any group
```

The REXX API

The SecureEntry REXX API is built into the rexx dll named RXUCM. The batch administration tools are good samples of its usage, namely EDYERASE.CMD, EDYADMIN.CMD and EDYDEFS.CMD. Follow the description of the provided API functions :

RxEdyUcmLoadFuncs

This function is used to load the rest of the API, as in the following sample:

```
If RxFuncQuery('RxEdyUcmLoadFuncs')  Then
do
  call RxFuncAdd 'RxEdyUcmLoadFuncs',  'RXUCM', 'RxEdyUcmLoadFuncs'
  call RxEdyUcmLoadFuncs
end
```

RxEdyUcmDropFuncs

This function is used to unload the API functions, as in the following sample:

```
call RxEdyUcmDropFuncs
```

RxEdyUcmGetError

This function returns a descriptive error message concerning the last issued API function which ended up in error, as in the following sample :

```
say RxEdyUcmGetError()
```

Note that the reported error has always the format :

ERROR:*errorcode* SEVERITY:*severity* TYPE:*type* MESSAGE:*message*

Where:

Errorcode Numeric error code

Severity Severity level for the error. The following convention is used by the different SecureEntry components :

'E' for error

'W' for warning

'I' for informative

Type Type of error.

This intends to help locate the component which generated the error. It is a four char string, where the convention used has been that the first two represent the component who posted the error, and the second two represent the subcomponent. As a sample :

UCSQ Means that the UCM agent encountered a DB2 SQL error.

LSER Means that the Lan Server agent encountered a Lan Server API error.

SESR Means that the SecureEntry agent gave an error posted by the registry component.

RXUC Means that the error was produced within the RXUCM DLL itself.

Message Descriptive message for the error.

Note that the components are not to follow this conventions mandatory, but just advisory. Note also that not all of the functions will always return errors that can be retrieved through this function, but just the indicated ones.

RxEdyUcm_AddObj

This function adds a new object to the repository. The correct invocation parameters are shown by the following syntax diagram :

```
Rc=RxEdyUcm_AddObj(subsystem,objecttype,objectname1,objectname2,'keysstem')
```

Where:

Subsystem is the subsystem identification for the agent to process the call. Normally you should use always 'SENT' for SecureEntry administration.

objecttype Is the object type for the function.

Valid object types are :

'GROUP'

'GROUPGROUP'

'RESOURCE'

'RESOURCEGROUP'

'RESOURCEUSER'

'SUBSYSTEM'

'USER'

'USERGROUP'

objectname1 and *objectname2*

These are the id(s) for the object to be processed. Use the second one for complex objects which require two id's. If this field is not needed, a blank string (') must be placed.

keysstem

This is the name of a REXX stem where the keywords and associated values for the object are stored, following regular REXX standards, i.e the 0th element contains the number of elements, and the rest are of the form 'keyname=keyvalue'.

This function will return :

'OK' for correct operation

'WARNING' for warning return code

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_UpdateObj

This function updates an existing object in the repository. the correct invocation parameters are shown by the following syntax diagram :

```
Rc=RxEdyUcm_UpdateObj(subsystem,objecttype,objectname1,objectname2,'keysstem')
```

Where:

subsystem is the subsystem identification for the agent to process the call. Normally you should use always 'SENT' for SecureEntry administration.

objecttype Is the object type for the function. Valid object types are :

'GROUP'

'GROUPGROUP'

'RESOURCE'

'RESOURCEGROUP'

'RESOURCEUSER'

'SUBSYSTEM'

'USER'

'USERGROUP'

objectname1 and *objectname2*

These are the id(s) for the object to be processed. Use the second one for complex objects which require two id's. If this field is not needed, a blank string (') must be placed.

keysstem

This is the name of a REXX stem where the keywords and associated values for the object are stored, following regular REXX standards, i.e the 0th element contains the number of elements, and the rest are of the form 'keyname=keyvalue'.

This function will return :

'OK' for correct operation

'WARNING' for warning return code

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_ViewObj

This function retrieves an existing object from the repository. the correct invocation parameters are shown by the following syntax diagram :

```
Keysstem.0=0
Rc=RxEdyUcm_ViewObj(subsystem,objecttype,objectname1,objectname2,'keysstem')
```

Where:

Subsystem is the subsystem identification for the agent to process the call. Normally you should use always 'SENT' for SecureEntry administration.

objecttype Is the object type for the function. Valid object types are :

'GROUP'

'GROUPGROUP'

'RESOURCE'

'RESOURCEGROUP'

'RESOURCEUSER'

'SUBSYSTEM'

'USER'

'USERGROUP'

objectname1 and *objectname2*

These are the id(s) for the object to be processed. Use the second one for complex objects which require two id's. If this field is not needed, a blank string (') must be placed.

keysstem

This is the name of a REXX stem where the returned keywords and associated values for the object will be stored, following regular REXX standards, i.e the 0th element will contain the number of elements, and the rest are of the form 'keyname=keyvalue'.

Note that the 0'th element of the stem is initialized to 0, indicating that we request for all information about the object. Some agents will accept and serve requests taking keywordstem as an input parameter also for retrieving specific keywords values. This is why you are encouraged not to forget this initialization in order to avoid erratic program behavior.

This function will return :

'OK' for correct operation

'WARNING' for warning return code

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_DeleteObj

This function deletes an existing object from the repository. the correct invocation parameters are shown by the following syntax diagram :

```
Rc=RxEdyUcm_DeleteObj(subsystem,objecttype,objectname1,objectname2,'keysstem')
```

Where:

subsystem is the subsystem identification for the agent to process the call. Normally you should use always 'SENT' for SecureEntry administration.

objecttype Is the object type for the function. Valid object types are :

'GROUP'

'GROUPGROUP'

'RESOURCE'

'RESOURCEGROUP'

'RESOURCEUSER'

'SUBSYSTEM'

'USER'

'USERGROUP'

objectname1 and *objectname2*

These are the id(s) for the object to be processed. Use the second one for complex objects which require two id's. If this field is not needed, a blank string (") must be placed.

keysstem

This is the name of a REXX stem where the keywords and associated values for the object are stored, following regular REXX standards, i.e the 0th element contains the number of elements, and the rest are of the form 'keyname=keyvalue'. It is taken as an input parameter for selectively deleting only some keywords, and not the full object. This is why, when desiring to delete the complete object, you should set this parameter to the void string (").

This function will return :

'OK' for correct operation

'WARNING' for warning return code

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_SetAgent

This function can be used when developing specific agents for the API, in order to tell the Rexx API to redirect all requests directly to the agent, and do not go through the 'selector' agent. This way you can test in a non SecureEntry environment before the integration phase. The invocation command is :

```
rc=RxEdyUcm_SetAgent(agentid)
```

Where agentid is the two character string that identifies the agent DLL, which will have to be called : EDYxxAGT.DLL.

This function will return :

'OK' for correct operation

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_QueryUsersDB

This function allows you to query the location of the local users database. Takes no parameters and will return the component ID for the database. The invocation sample is :

```
locdb=RxEdyUcm_QueryUsersDB()
```

The return values can be:

'SR' For SecureEntry registry

'LS' For Lan server UPM

'ERROR' If any error. Use then the RxEdyUcmGetError function to obtain further information if required.

RxEdyUcm_Enable_Remote

This function will enable/disable remote access of the rest of the API when UCM is present, independent of the SGM_UCM_ENABLE environment variable setting and only valid for the current process. It takes one argument which can be 0 or 1. As a sample :

```
rc=RxEdyUcm_Enable_Remote(0)
```

This function will return :

'OK' for correct operation

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_Connect

Use this function to inform the administration subsystem that you are to start working with the API, and issue the required DB2 connect whenever host access is required.

The sample invocation code looks like :

```
Rc=RxEdyUcm_Connect()
```

This function will return :

'OK' for correct operation

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_Disconn

Use this function to inform the administration subsystem that you have ended working with the API, and issue the required DB2 disconnect if host access was issued.

The sample invocation code looks like :

```
Rc=RxEdyUcm_Disconn()
```

This function will return :

'OK' for correct operation

'ERROR' for error/not completed operation

If you receive an error or a warning, then the RxEdyUcmGetError function can be used to retrieve further information about it.

RxEdyUcm_GetUser

Use this function to obtain information about the currently logged on user. It will not normally return an error, and the invocation code is :

```
userinfo=RxEdyUcm_GetUser()
```

The returned string can easily be parsed since it is of the form :

`USER:`*`username`* `GROUP:`*`groupname`* `DOMAIN:`*`domainname`* `COMPUTER:`*`computername`* `ADMIN:`*`x`*

Where x can be 0 or 1.

The 'C' API

A mapping of the same API explained as the REXX API exists for 'C' programs. You can find the related include and library files within the SecureEntryPath, API\SOURCES\EDYUCM path, as well as a commented sample program to list the first level defined groups. What follows is a description of the API and differences with the already discussed, REXX API:

The following functions are available, and predefined in the EDYSLAGT.H file :

EdyUcm_Enable_Remote

This function is only useful when you grant access to the local database, even if users centralized management (UCM) is installed. The idea is to do the following :

```
...
Centralized processing calls
...
EdyUcm_Enable_Remote(0)
  ...
  Local administration calls
  ...
EdyUcm_Enable_Remote(1)
...
Centralized processing calls
...
```

So, the real function of this API is to inform the administration subsystem when the administration functions have to be redirected against the local database instead of the host DB2 UCM database. Note that the parameter sets a boolean flag, not a recursive counter.

EdyUcm_Connect

This function call is only mandatory when using the UCM centralized management option. Its mission is to do the DBM connect to the database, and takes no parameters. In the event that your code requires this call, it must be the first call before any other API call.

EdyUcm_Disconn

This is the opposite to the previous call. Only mandatory if UCM is in use, must then be the last call to the administration API before your program terminates. Takes one parameter of type long, that is only used when UCM is available, and indicates whether to commit or rollback the changes made before disconnecting. (set to 0 for commit, 1 for rollback). Note that at this point and since most UCM API calls are atomic, it does not really matter what parameter you code for this function, but it we suggest you to code commit or rollback appropriately just in case SecureEntry/2 in the future provides for transactional integrity across calls.

EdyUcm_AddObj

This function, like its REXX counterpart, can be used to add objects of a given type to the repository. See **Common parameters definition**

EdyUcm_DeleteObj

This function, like its REXX counterpart, is used to delete objects of a given type to the repository, or specific keywords. See **Common parameters definition**

EdyUcm_UpdateObj

This function, like its REXX counterpart, can be used to modify, add or delete keyword/value pairs from a given existing object. See **Common parameters definition**

EdyUcm_ViewObj

This function, like its REXX counterpart, is used to view a given object, that is, obtain the list of keyword/values associated to a given existing object. See **Common parameters definition**

EdyUcm_FreeMemUcm

This function is to be used after the EdyUcmViewObj, to free up the resources allocated by that call.

Common parameters definition: Parameters for the Add, Update, Delete and View functions :

Ucm_ObjectType Object type to process:

USR
User
GRP
Group
RES
Resource
USR_GRP
Link user-group
RES_USR
Link resource-user
GRP_GRP
Link group-group
RES_GRP
Link resource-group

void * Pointer to a structure that contains the data for each object type. The structure is Ucm_*XxxXxx*Data where *XxxXxx* is the object type used. This structure will always be an input parameter and has the following fields :

Subsystem

This is the subsystem destination ID. Set to the string 'SENT' for SecureEntry administration.

Object instance Ids

These are one or two fields, which are used to select the appropriate object instance (i.e to get the information about a user named john, the user id should be set to 'JOHN').

Condition

Leave this field blank if not using UCM (initialized to 0's). Otherwise, it can be set to the SQL select condition to run against the specified table.

unsigned short Number of keyword definitions to process. On view calls, set this field normally to 0, unless you want to obtain specifically some object keywords, and not all of them.

Ucm_KeyData * Pointer to a structure that is an array of Ucm_KeyData elements, containing the keyword definitions. For the View function, this parameter is Ucm_KeyData **, and as with the previous field, you can set this to point to a NULL initialized pointer, unless you want to obtain specifically some of the object keywords, and not all of them.

Ucm_Error * Pointer to a Ucm_Error structure, that contains all the necessary information to access the error.

Nerror This is the error code

Severity This is the error severity, as defined in the include file (i.e, 'I','W', 'E','C','S'),

Type Error type, and gives a clue on what type of error code the structure is returning :

UCSQ
UCM SQL error (Look for errorcode in the SQL reference)
UCAP

UCM API error (Look for errorcode in the EDYSLAGT.H file)
UCME
UCM memory error (not enough resources)
LSER
Lan Server error (Look into Lan server error documentation)
SESR
SecureEntry registry error (as defined in EDYERROR.H)

msg Descriptive message

asoc_struc Not used by now

Default allowable keywords

Keywords for object type USER

LAN_DATA the values for these keyword are pairs of (subkeyword=value) separated by binary 0 bytes, with a trailing final 0 (i.e, same as when dealing with a program environment). The type for this keyword must be 'A' (ASCII). The allowable subkeywords expected in the value field are :

HOUR_START start logon allowed hour

HOUR_END end logon allowed hour

PRIV_USER ADMIN,USER, or GUEST (Only for Lan Server environments)

FULL_NAME Full name

CONNECTION 1 (allowed), 0 (not allowed)

USER_EXPIRE NEVER, or day-month-year

DESCRIPTION Free text description

PASSWORD User password, only valid as input

PASSWD_EXPIRED 1 (expire password), 0 (do not expire password)

HOME_DIR \\MachineName\x$\path, or x:\MachineName\x$\path. Only valid in Lan Server environments

MAX_STORAGE As required by Lan Server. -1=no limit. Only valid in Lan Server environments

SCRIPT_PATH As required by Lan Server. Only valid in Lan Server environments

Other The keyword is assumed to be a security component identificator, as described in the 'SENTRY.DSC' file located within the SecureEntry path, INSTALL directory. The type is expected to be 'B' (binary), and values are buffers with the complete copy of the file data, with a leading LONG set equal to 4.

Keywords for object type GROUP

LAN_DATA the values for these keyword are pairs of (subkeyword=value) separated by binary 0 bytes, with a trailing final 0 (i.e, same as when dealing with a program environment). The type for this keyword must be 'A' (ASCII). The allowable subkeywords expected in the value field are :

DESCRIPTION Free text description

FULL_NAME Full name

Other The keyword is assumed to be a security component identificator, as described in the 'SENTRY.DSC' file located within the SecureEntry path, INSTALL directory. The type is expected to be 'B' (binary), and values are buffers with the complete copy of the file data, with a leading LONG set equal to 4.

Keywords for object type RESOURCE

This objects are only supported in Lan Server environment. All of the defined keywords are of type 'A' (ASCII). The id for this object is the alias name.

SERVER Server name

TYPE FILES, TREE, PRINTER or SERIAL

WHEN_SHARED STARTUP, BYADMIN, or DYNAMIC

RESNAME Resource name

DESCRIPTION Free text description

MAX_CONN Max number of supported connections

Keywords for object type RESOURCEGROUP and RESOURCEUSER

These are 'A' (ASCII) type keywords, only valid for Lan Server environments.

ACCESS Access type, i.e RWX

LOGON_ASN Logon assignment device

Keywords for object type USERGROUP and GROUPGROUP

No keywords defined.

Objects listing

If you want to list the objects existing of a given type, you should issue an EdyUcm_ViewObj call, but setting the appropriate object IDs as '*'. This will generate a list of all the objects found matching the criteria within the returned keyword structure. Note that the only exception to this rule is when listing users or groups. You can only access the list of users or groups in a way that works in all environments by viewing the appropriate USERGROUP or GROUPGROUP object list, as shown in the sample provided (LISTGRP.C).

Additional notes:

Error reporting : Errors are returned in a per function basis within the provided error structure (Ucm_Error) in the C API. The function call ret. code for the EdyUcm_xxxObj functions will be :

Rc=0 Function successful

Rc<0 Error. Function not performed. Look into the error structure for further information.

Rc>0 Warning. Function performed, but an unexpected condition was found. Look into the error structure for further information.

Remember to initialize all structures whenever calling the API. Unless otherwise noted, all fields should be set to 0. This includes the error (Ucm_Error) and keywords structure (Ucm_KeyData).

Remember to free up the memory for the returned KeyData buffers using the function EdyUcm_FreeMemUcm(). This is necessary only for API returned structures (i.e, only in EdyUcm_ViewObj calls).

Indirection through the '@' character is not supported within the 'C' API. All component data is passed by value at this level, as explained above under **default allowable keywords**

Uppercasing and blank stripping of ID's is not done for you at the 'C' API level.

You should link and redirect calls to the 'EDYSLAGT.DLL' when working with the 'C' API.

SecureEntry OS/2 utilities

SecureEntry provides with several tools to do all the necessary coding for a final security solution. This chapter describes those that run under pure OS/2 environment.

Session event launcher : EDYUTIL

User Information display : EDYUSINF

Lan server classes override : WPSLAN

Switch list utilities : EDYSWL2

Semaphore management : EDYSEM2

Selective application closer : EDYCLOSE

CM/2 emulators manager : EDYE3270

Master boot record saver : EDYRWMBR

Session event launcher : EDYUTIL

This command line utility allows you to launch a given session event.

The command line invocation syntax is :

```
EDYUTIL TOLOCKUP | TOLOGOFF | FRLOGOFF | REBOOT
        TOSHTDWN [REBOOT] |
        FRSHTDWN [REBOOT] |
        NEXTLOGON [/CLEAR | /ASK /U:userid /P:password
                   /N:newpass /D:domain] |
        NEXTUNLOCK [/CLEAR | /ASK /U:userid|NO
                   /V:validationpassword|NO /P:password] |
        SIGNAL "aString" |
        SUPERSIGNAL "aString"
```

TOLOCKUP

To launch a lockup event

TOLOGOFF

To launch a logoff event

FRLOGOFF

To launch a forced logoff event (not cancellable)

REBOOT

To reboot the machine immediately

TOSHTDWN

To launch a shutdown event. Use the REBOOT parameter to force a reboot once shutdown is completed

FRSHTDWN

To launch a forced shutdown event (not cancellable). Use the REBOOT parameter to force a reboot once shutdown is completed

NEXTLOGON

To prefill the next logon parms data area. When using this parameter you must specify either

/CLEAR To erase the previous logon parms data area, or

One or more of :

/U: To set up the userid for the next logon

/P: To set up the user password for the next logon

/D: To set up the domain for the next logon

/N: To set up the new password field for the next logon

/ASK To make the default SecureEntry logon panel to prompt for entries. Otherwise, if the logon parms data area is found filled at logon time, then an automatic logon attempt will be made with the preconfigured parameters.

NEXTUNLOCK

To setup the validation password/userid to use for the current session unlock panels, and also to optionally prefill the password field during the next unlock event. When using this parameter you must specify either

/CLEAR To erase the previous unlock parms data area, or

One or more of :

/U: To set up the informational userid for the next unlock, or NO for none

/V: To set up the user password used to validate the input password for the next unlock

/P: To prefill the user password used for the next unlock

/ASK To make the default SecureEntry unlock panel to prompt for entries. Otherwise, if the unlock password (/P parameter) data area is found filled at unlock time, then an automatic unlock attempt will be made with that parameters.

SIGNAL

To invoke the SIGNAL user exit passing it the aString parameter.

SUPERSIGNAL

To invoke the SUPERSIGNAL user exit passing it the aString parameter.

If the Nextlogon/Nextunlock features are used, they can be used as late as within the user exit before the logon/unlock dialog, thus effectively providing for an easy way to substitute the default SecureEntry logon/unlock panels with a custom one. Remember, however, that for a dialog to be shown within a user exit, it has to be system modal. Besides, the responsibility to maintain the focus on your customized panels becomes yours.

Note that this features are very powerful, since they also allows for a way to enter user data directly from a device which does not necessarily has to be the keyboard.

This program is provided as a tool, but also as a sample C program, with the appropriate API files for if you want to launch your own events from within your applications, all of this located in the SecureEntryPath\API\SOURCES\EDYFLOW directory.

Note that this program executes the events in an asynchronous way, so that it will normally return control long before the indicated event is processed.

This utility is located in the SecureEntryPath\EXEC directory of the workstation.

IMPORTANT NOTES

It is not allowed to use this utility during machine startup, since the SecureEntry kernel is not yet fully active at this time. Should you need to use it at every boot, you can do it during user exit after start up processing, the earliest.

SES events are serialized by architecture. This means that you will not normally be able to use this utility to launch SES events while another event is being processed. A clear example of this is when you want to shutdown or logoff from within a given user exit. Use instead the appropriate ret. code to return from the user exit. An exception to this rule is when the logon and unlock panels are being displayed. SecureEntry allows in this case you to launch EDYUTIL from a background process to force a shutdown or logoff event.

User Information display : EDYUSINF

This utility displays the currently logged on user, group, domain, server, computer and whether it is administrator or not.

The command line invocation sentence is :

```
EDYUSINF
```

It will display the information as in the following sample :

```
USER:USERID
GROUP:
DOMAIN:SEDOMAIN
SERVER:\\SESRV01
ADMIN:YES
COMPUTER:SESRV01
```

This utility is located in the SecureEntryPath\TOOLS directory of the workstation.

Lan Server classes override : WPSLAN

If you are running under a Lan Server environment, then the Lan Server adds Lan entries to most objects system menus, which you may want not to display. In that case you can use the WPSLAN program to install or uninstall those entries.

The command line invocation sentence is :

```
WPSLAN /U to deregister the LAN server classes
WPSLAN /I to register and replace the LAN server classes
```

This utility is located in the SecureEntryPath\TOOLS directory of the workstation.

Switch list utilities : EDYSWL2

The files that compose the Window List utility are located in the SecureEntryPath\TOOLS directory on the workstation.

If you want to display *on-line information* about this procedure, enter:

```
EDYSWL2
```

This utility provides access to the task list and to the system sessions (Window List), even to those that are hidden. It allows for session management.

The syntax for this command is:

```
EDYSWL2 [action:session]
```

Action

Action to be performed with the specified session. The parameter value can be:

HIDE

Removes the session from the Window List.

KILL

Ends the process running in the session.

SHOW

Restores the session in the Window List.

SWITCHTO

Gives control to the session.

LIST

Displays the list of sessions. No session must be specified for this action.

MINIWIN

Minimizes all windows for the specified session.

HIDEWIN

Hides all windows for the specified session.

RESTWIN

Restores all windows for the specified session.

MAXIWIN

Maximizes all windows for the specified session.

ACTIWIN

Activates all windows for the specified session.

DEACWIN

Deactivates all windows for the specified session.

CLOSWIN

Closes all windows for the specified session.

Note that all the *WIN actions are performed against the windows, and not the switch list entries, in an asynchronous mode (rc=0 only means that the message has been sent).

Session

Session that will be affected by the action. The parameter value can be specified in different formats:

HSWL:*handle*

Where *handle* is the Window List handle in hexadecimal.

PID:*pid*

Where *pid* is the process identifier in hexadecimal.

SESSID:*sessionid*

Where *sessionid* is the session identifier in hexadecimal.

SLINDEX:*index*

Where *index* is the Window List index in decimal.

TITLE:*title*

Where *title* is the session title in ASCII.

Blank characters and control keys must be replaced by underscores.Wildcards are allowed. All task list sessions matching the specifications will have the intended action applied. Note that the carriage return character (CR) plus the line feed character (LF) require two underscores.

Semaphore management : EDYSEM2

If you want to display *online information* about this utility, enter:

EDYSEM2

This utility provides access to OS/2 event semaphores from a command line. It is intended to ease process synchronization between working sessions.

The syntax for this command is :

```
EDYSEM2 [Post|Wait] Semname
```

Post

Opens the OS/2 event semaphore \SEM32\Semname, provided it is already created, and posts an event. If the semaphore does not exist, it waits for the semaphore being created.

Wait

Opens the OS/2 event semaphore \SEM32\semaphore_name, provided it is already created, or creates the semaphore, if it does not exist. Then, it waits for an event to be posted.

Semname

Name of the semaphore. The parameter value is not case-sensitive.

SecureEntry also provides the EDYSEMV utility, with the same functionality and parameters, which runs under a VDM.

The EDYSEM2 utility can be stopped by pressing Ctrl-C. The EDYSEMV utility does not support it.

Selective application closer : EDYCLOSE

The selective application closer utility enables to close applications after a given event.

The files that compose this utility are located in the SecureEntryPath\TOOLS directory on the workstation.

Those applications that will remain active must be specified in an INI file.

When the EDYCLOSE procedure runs, the INI file is read and only the applications that are **not** specified are closed. To close the sessions, the procedure uses a DosKillProcess. If the file can not be found, no application is closed.

The applications that do not appear in the Window List are not closed, no matter whether they are specified in the INI file or not. The session where the EDYCLOSE procedure runs is not closed, either.

To start the procedure, enter the following:

```
EDYCLOSE [Filepath]Filename[.INI]
```

Filepath

Full path of the INI file with the specifications. The default is the current directory.

Filename

Filename of the INI file with the specifications. The default is **EDYCLOSE**.

For example, you can specify to kill all the user applications present in the window list except those specified in the list file by starting this procedure in any appropriate user exit provided by the SecureEntry user exits API, or

your own applications. Note that it is not really necessary to use this tool to kill user applications at logoff time, since this is done automatically by SecureEntry, but it may be needed to be used by your application in some instances.

Editing the selective application closer file

To produce the input file for the selective application closer utility, you can use a settings notebook that is opened by entering:

```
EDYEDTCL [Filepath]Filename[.INI]
```

Filepath

Full path of the INI file with the specifications. The default is the current directory.

Filename

Filename of the INI file with the specifications. The default is **EDYCLOSE**.

The following information can be specified:

Name

Identifies the application that will not be closed and must match the corresponding entry in the Window List.

To specify a task that corresponds to an object title of more than one line, use the \n string. For example, Minimized\nWindow Viewer.

After typing the task name, click on the Add push button, so that the name is added to the List. Specify as many task names as applications will not be killed.

List

Contains the task names of all those applications that will remain active. To remove a task name from the list, select it and click the Remove push button.

CM/2 emulators manager tool : EDYE3270

This command line utility allows controlling of the start and stop of the CM/2 emulator sessions. It is intended to be added as part of the SecureEntry user exits to have the sessions started as desired under a per signon basis.

The command line invocation sentence is :

```
EDYE3270 START  to start the configured emulator sessions
EDYE3270 STOP   to stop the configured emulator sessions
EDYE3270 SWITCH to switch status of the configured emulator sessions
```

This utility is located in the SecureEntryPath\TOOLS directory of the workstation.

If you don't use CM/2 support for your emulators, but the Access Feature (AF) through Personal Communications (PCOM), then the utility EDYE3270 is useless and you will have to use the proper PCOM tools as follows:

```
PCSWS    name.WS      to start the session configured in the file  name.WS
PCSBAT   name.BCH /R  to start the set of sessions configured in the file  name.BCH
```

```
PCSSTOP              to stop the up and running emulator sessions
```

SecureEntry will automatically stop the up and running emulator sessions if they are identified in the file EDYKILL.NOT, whose support is currently available.

To get more information about the file EDYKILL.NOT refer to Configuration files and its own comments section. To get more information about PCOM emulators handling refer to *Personal Communications v4.2 Quick Beginnings*.

Master boot record saver : EDYRWMBR

All systems are exposed to virus or other events which can make them non bootable. SecureEntry provides now a means of restoring the original master boot records in case an external event destroys them, no matter whether or not the Software Boot Protection is installed.

When using this utility, you are encouraged to generate a copy of the master boot records of the system in the last diskette of the OS/2 Utility Diskettes, so that you can restore them after booting from the Utility Diskettes if the system cannot boot from hard disk.

Setup process

Make a back-up copy of the original master boot records:

```
EDYRWMBR.EXE /R [/Fpath_name]
Where:
/R          reads the original  MBRs from the physical disks.
/Fpath_name writes the files containing the  MBRs in the specified path_name. If not
present, it defaults to current directory.
```

There exists one saved file for each physical disk in the system. The name of these files is EDYBOOT and their extension is 001 for the first physical disk, 002 for the second one, and so on. For example, in a system with two hard drives, the command:

```
EDYRWMBR /R /Fa:
```

will read the original master boot records of both disks and write them to the files EDYBOOT.001 and EDYBOOT.002 in the A: drive.

If the Software Boot Protection is installed, the original master boot records will be written to the external files. If the Software Boot Protection is not installed, the current master boot records will.

If these files already exist, they will be overwritten.

If the master boot records are invalid, this utility will refuse to generate the external files.

Restore a back-up copy of the original master boot records:

```
EDYRWMBR.EXE /W [/Ppassword] [/Fpath_name]
Where:
/W          writes the saved  MBRs to the physical disks.
```

```
/Ppassword  this parameter is mandatory if the Software
            Boot Protection is enabled.
/Fpath_name reads the files containing the  MBRs from
            the specified  path_name. If not present, it
             defaults to current directory.
```

Having a system with the Software Boot Protection installed and the saved files in the current directory, you can restore the original MBRs using the command:

```
EDYRWMBR /W /Pxxxx
```

Having a system without the Software Boot Protection installed and the saved files in the A: drive, you can restore the original MBRs using the command:

```
EDYRWMBR /W /Fa:
```

Since this command will be used when the system cannot start, it will apply the strategy "do my best", hence very few checkings will be carried out on the current master boot records of the physical disks.

Virtual DOS Machine utilities

Migrating DOS programs to use them in the OS/2 Virtual DOS Machines (VDMs) requires changes in the processing environment.

OS/2 is a multiprocessing platform from which the DOS programs that you migrate can benefit. That is, several DOS programs can run within their own environments in different VDMs.

SecureEntry provides the following programs that meet the synchronization requirements involved in this multiprocessing:

VDM programs starter : EDYSTRTV

To start OS/2 programs from a VDM.

VDM NET commands launcher : EDYNETV

To run network commands from a VDM.

VDM switch list utility : EDYSWLV

To access the OS/2 task list from a VDM.

VDM session switcher : EDYBRNGV

To bring to foreground a OS/2 program from a VDM.

VDM semaphores : EDYSEMV

To work with OS/2 semaphores from a VDM.

The files corresponding to these programs are located in the SecureEntryPath\TOOLS directory on the workstation.

VDM programs starter : EDYSTRTV

If you want to display *on-line information* about this utility, enter:

```
EDYSTRTV
```

This utility starts DOS programs in VDMs. The CMD.EXE path must be specified in the PATH statement of the OS/2 CONFIG.SYS file.

The syntax of this command is:

```
EDYSTRTV [/S:Sessiontype] [/F:Sessionmode] [/T:Sessiontitle]
         [/D:DOSsettings] [/X:Winposx] [/Y:Winposy] [/W:Width]
         [/H:Height] Programname Programparameters
```

Sessiontype

The following types of session are supported, which should be specified by these parameter values:

1 OS/2 full screen

2 OS/2 window

3 OS/2 Presentation Manager

4 Virtual DOS machine (either DOS VDM or WINOS2 VDM)

7 VDM window

The default is **1**.

Sessionmode

The following modes of session are supported, which should be specified by these parameter values:

0 Foreground

1 Background

The default is **1**.

Sessiontitle

The title of the session can be any string.

Blank characters and control keys must be replaced by underscores. Note that the carriage return character (CR) plus the line feed character (LF) require two underscores.

DOSsettings

The DOS settings must be separated by the ; character. A file can also be specified by @file_name.

Winposx

Horizontal coordinate of the window position.

Winposy

Vertical coordinate of the window position.

Width

Width of the window.

Height

Height of the window.

Programname

Program to be started.

Blank characters and control keys must be replaced by underscores. Note that the carriage return character (CR) plus the line feed character (LF) require two underscores.

Programparameters

Parameters for the program to be started.

VDM NET commands launcher : EDYNETV

If you want to display *online information* about this utility, enter:

```
EDYNETV
```

This utility runs the OS/2 NET command from VDMs.

For information about the OS/2 NET command, enter:

```
EDYNETV HELP
```

Interactive commands are not supported. Redirected input (for example, when using the SEND command) or absolute paths (for example, when using the COPY command) are not supported, either.

The following is required:

The Virtual DOS LAN API Support of LAN Server must be installed.

The LAN Requester must be active.

The SecureEntry user must be logged-on.

The NET.EXE and EDYSEM2.EXE paths must be specified in the PATH statement of the OS/2 CONFIG.SYS file.

The syntax for this command is:

```
EDYNETV [ MACH | USE redirectionspecs ]
```

MACH

Returns the machine ID.

USE

Returns the list of all redirected devices and drives, if no redirection specifications are provided.

The following can be specified:

dev \\server\netname [password]

Starts a device redirection.

d: \\server\netname [password]

Starts a drive redirection.

dev /D

Ends a device redirection.

Other specifications are directly passed to the OS/2 session where the NET.EXE program is running.

VDM switch list utility : EDYSWLV

If you want to display *online information* about this utility, enter:

```
EDYSWLV
```

This utility provides access to the task list and to the system sessions (OS/2 Window List), even to those that are hidden in the switch list, from VDMs. It allows for session management.

The CMD.EXE and EDYSWL2.EXE paths must be specified in the PATH statement of the OS/2 CONFIG.SYS file.

The syntax for this command is:

```
EDYSWLV  [action:session]
```

action

Action to be performed with the specified session. The parameter value can be:

HIDE

Removes the session from the Window List.

KILL

Ends the process running in the session.

SHOW

Restores the session in the Window List.

SWITCHTO

Gives control to the session.

LIST

Displays the list of sessions. No session must be specified for this action.

MINIWIN

Minimizes all windows for the specified session.

HIDEWIN

Hides all windows for the specified session.

RESTWIN

Restores all windows for the specified session.

MAXIWIN

Maximizes all windows for the specified session.

ACTIWIN

Activates all windows for the specified session.

DEACWIN

Deactivates all windows for the specified session.

CLOSWIN

Closes all windows for the specified session.

session

Session that will be affected by the action. The parameter value can be specified in different formats:

HSWL:*handle*

Where *handle* is the Window List handle in hexadecimal.

PID:*pid*

Where *pid* is the process identifier in hexadecimal.

SESSID:*sessionid*

Where *sessionid* is the session identifier in hexadecimal.

SLINDEX:*index*

Where *index* is the Window List index in decimal.

TITLE:*title*

Where *title* is the session title in ASCII.

Blank characters and control keys must be replaced by underscores. Note that the carriage return character (CR) plus the line feed character (LF) require two underscores.

Note : In order for this utility to work properly, the companion EDYSWL2 utility must be located within any of the environment PATH directories.

VDM session switcher : EDYBRNGV

If you want to display *online information* about this utility, enter:

```
EDYBRNGV
```

This utility starts a program, provided it is not started, or switches to a program, if already started, and brings the program to foreground. The CMD.EXE and EDYSWL2.EXE paths must be specified in the PATH statement of the OS/2 CONFIG.SYS file.

Both DOS and OS/2 programs can be located either in a path specified in the PATH statement or in any other path, which must be specified by entering the full path of the program. DOS programs can also be located in the current path.

The syntax for this command is:

```
EDYBRNGV [/S:Sessiontype] [/F:Sessionmode] [/T:Sessiontitle]
         [/D:DOSsettings] [/X:Winposx] [/Y:Winposy] [/W:Width]
         [/H:Height] Programname Programparameters
```

Sessiontype

The following types of session are supported, which should be specified by these parameter values:

1 OS/2 full screen

2 OS/2 window

3 OS/2 Presentation Manager

4 Virtual DOS machine (either DOS VDM or WINOS2 VDM)

7 VDM window

The default is **1**.

Sessionmode

The following modes of session are supported, which should be specified by these parameter values:

0 Foreground

1 Background

The default is **0**.

Sessiontitle

The title of the session can be any string.

Blank characters and control keys must be replaced by underscores; note that the carriage return character (CR) plus the line feed character (LF) require two underscores.

DOSsettings

The DOS settings must be separated by the ; character. A file can also be specified by @file_name.

Winposx

Horizontal coordinate of the window position.

Winposy

Vertical coordinate of the window position.

Width

Width of the window.

Height

Height of the window.

Programname

Program to be started.

To specify a CMD program, the parameter value must be:

```
CMD.EXE /C filename.CMD
```

Programparameters

Parameters for the program to be started.

VDM semaphores : EDYSEMV

If you want to display *online information* about this utility, enter:

```
EDYSEMV
```

This utility provides access to OS/2 event semaphores from VDMs. It is intended to ease process synchronization either between VDMs or between VDMs and OS/2 sessions.

The syntax for this command is :

```
EDYSEMV [Post|Wait] Semname
```

Post

Opens the OS/2 event semaphore \SEM32\Semname, provided it is already created, and posts an event. If the semaphore does not exist, it waits for the semaphore being created.

Wait

Opens the OS/2 event semaphore \SEM32\semaphore_name, provided it is already created, or creates the semaphore, if it does not exist. Then, it waits for an event to be posted.

Semname

Name of the semaphore. The parameter value is not case-sensitive.

SecureEntry also provides the EDYSEM2 utility, with the same functionality and parameters, which runs under OS/2.

The EDYSEM2 utility can be stopped by pressing Ctrl-C. The EDYSEMV utility does not support it.

SecureEntry maintenance utilities

SecureEntry provides also with a set of utilities for its own maintenance, plus general system maintenance. Some of these are used by SecureEntry itself during its different processes, but may be of interest to solve problems or defects in specific situations.

Deleting directories with EDYDD

Unpacking installation files using UNPACK32

Updating the SecureEntry database definitions with UPDATEDB

Recreating SecureEntry database with CREADB

Registering WP classes with EDYCLASS

Fixing OS/2 INI files by means of EDYCLINI

Installing PM DLLs with EDYWINI

Recreating the SecureEntry workbench using EDYCRWRK

Establishing SecureEntry Lan Links with EDYSRV and EDYFREE

Migrating SecureEntry 2.0 definitions using MIGRADB

Controlling the size of the log files using EDYLOGFS

Browsing the SecureEntry audit files with EDYLOGBR

Obtaining and viewing configuration image photo files with EDYPHOTO

WARNING!The incorrect use of some of this utilities may leave your system in a unstable/unusable state.

EDYDD

The EDYDD program is a generic 'delete directory' utility. It erases the contents of the specified directory and all of its descendants, no matter if any file is marked as readonly or not. This program is located in the SecureEntryPath\INSTALL directory.

The format for this utility is :

```
EDYDD directoryname [/N]
```

directoryname

The full path of the directory to delete

/N

Use this parameter to process without confirmation questions.

UNPACK32

The UNPACK32.EXE is a program used to obtain from a SecureEntry installation diskette bundle file a given component. It is used internally by SecureEntry during the installation or service process. Its syntax is exactly that of the UNPACK2.EXE provided with OS/2 base. Refer to the OS/2 documentation for more details. This program is located in the SecureEntryPath\INSTALL directory.

UPDATEDB

This program is used to update the SecureEntry database for static characteristics, such as passwords expiration days or adding new components to the table of recognized ones. Refer to the chapter named 'Adding your own components' for instructions on its usage. This program is located in the SecureEntryPath\INSTALL directory.

CREADB

This program is used during installation to create an empty SecureEntry database of components and users. This program takes as input a subsystem name and a components description file. It resides in the SecureEntryPath\INSTALL directory.

The syntax for this command is :

```
CREADB subsystemname configurationfile
```

subsystemname

Name of the subsystem to create. Only supported value is : SENTRY

configurationfile

Path and name for the configuration file. The one used by the SecureEntry installation is SecureEntryPath\INSTALL\SENTRY.DSC

EDYCLASS

This utility is used by SecureEntry to register/deregister the SecureEntry SOM required classes. It is not normally necessary to use this command manually, since the product takes care of having those correctly registered at startup. The program is located in the SecureEntryPath\INSTALL directory.

The syntax for this command is :

```
EDYCLASS /I | /U [binaryprofile]
```

/I

To register the SecureEntry classes.

/U

To deregister the SecureEntry classes. Optionally, specify a desktop restrictions profile which species the desired object settings to set as default ones.

EDYCLINI

This program can be used to maintain and cleanup the OS/2 system initialization files OS2.INI and OS2SYS.INI. Note that this program has to be used with extreme care, since it can easily destroy your system settings or cause system malfunctions.

WARNING : YOU ABSOLUTELY MUST KNOW WHAT YOU ARE DOING!!!!!

```
EDYCLINI [Operation mode] [Actions]

Operation mode:

  /B[+|-]   Batch mode: enable (-) or disable (+) user interaction.
  /D[+|-]   Deferred mode. /D+ will start the program after next boot.
            /D- will cancel the effect of a previous call with /D+.
  /T[+|-]   Test mode. /T+ will not write corrections to disk.
            /T- will update the  *.INI files.
  /V[0|1|2] Verbose mode: quiet (0) or verbose (2) operation.

Actions:

  /N[+|-]   Nowhere shadows: delete (+) or ignore (-) shadows
            in the  <WP_NOWHERE> folder.
  /H[+|-]   Handles: delete (+) or ignore (-) the handles of
            objects that have no matching file or directory.
  /I[+|-]   Identifiers: delete (+) or ignore (-) the object identifiers
            that have no matching handle.
  /Rdrives  Remove all the handles of objects that are defined
            in the specified drives. More than one drive can be
            specified using ';' as a separator.
  /U[i|d|c] UNC dives: ignore  (i), delete (d) or, if available, check (c)
            the drives specified with  UNC names.

Default parameters:

  EdyClini /B- /T- /V2 /N- /H- /I-  /Ui
```

Note that if you use the program to cleanup the <WP_NOWHERE> folder, then if the system malfunctions afterwards, the only way (not granted) method to fix it could be :

Reboot the machine to a command line through config.sys

Remove the <WP_NOWHERE> physical directory

Continue the boot process

Run the following little code :

```
/* */
call RxFuncAdd 'SysLoadFuncs', 'REXXUTIL', 'SysLoadFuncs'
call SysLoadFuncs
call SysSetObjectData '<WP_NOWHERE>','OBJECTID=<WP_NOWHERE>'
```

Now use any .INI maintenance tool to remove from the OS2.INI file the data corresponding to the application PM_ABSTRACT:FolderContents within the key corresponding to the object handle for <WP_NOWHERE>, which can be located scanning the application PM_WorkPlace:Location for <WP_NOWHERE> key.

Finally, reboot again

EDYWINI

This program is used to install or deinstall the PM DLLs required by some components. It resides in the SecureEntryPath\EXEC directory, and its syntax is :

```
EDYWINI /I[inifile] dllname [..dllname] |
        /U[inifile] dllname [..dllname] |
        /L[inifile]
```

/I

To register a set of DLLs (dllname(s)) to the system.

/U

To deregister a set of DLLs (dllname(s)) from the system.

/L

To list the registered DLL names.

Note that you can specify in all three commands an optional path and filename to the OS2.INI file to work with (inifile). By default, the one taken will be the one pointed to by the USER_INI *config.sys* variable.

The DLLs required by SecureEntry to be installed are named : *EDYWIN*, *EDYSESNO*, and *EDYLKSTR*. So, if at any time you recreate your *OS2.INI* file and intend to use it in a SecureEntry machine, you must issue the following command before the machine will become operative :

```
EDYWINI /I EDYWIN EDYSESNO EDYLKSTR
```

Note that this utility can be run in a non-PM environment, such as an Alt-F1 command line.

EDYCRWRK

This REXX command can be used to recreate the SecureEntry workbench if necessary. It should be used by an administrator, and its invocation syntax is :

```
EDYCRWRK
```

Note that this command is located in the SecureEntryPath\INSTALL directory.

EDYSRV and EDYFREE

The EDYSRV program is a daemon that runs in the background of network environment installed SecureEntry servers, and communicates with clients through netbios. If this program is necessary, the appropriate sentence to launch it will have been added to your EDYSTART.CMD (boot) command. Both programs reside in the SecureEntryPath\EXEC directory.

The invocation syntax for EDYSRV is :

```
(detach) EDYSRV [/N:netbiosname] [/S:netbiossessions]
                [/R:update_policy]
```

netbiosname

Netbios name to add to the network. Note that clients use, to communicate with this module, the name 'EDYdomainname', where domainname is the name of your Lan Server domain (if any).

netbiossessions

Number of parallel netbios sessions to use, where the default value is 4. You are encouraged to increase this number if you see big contention in the server machine (i.e, many users logging on at the same time with RACF or UCM).

update_policy

The update policy to use for this server. This parameter can take the following values:

```
/R:H -> EDYSRV will use the corporate policy
        stored in the central repository  (UCM) to apply the
        refresh branch process.
/R:I -> EDYSRV will override the update corporate policy
        and run the refresh branch process
        at IPL time.
/R:L -> EDYSRV will override the update corporate policy
        and run the refresh branch process during each user's  signon.
/R:N -> EDYSRV will override the corporate policy
        and will not run any branch refresh update.
/R:T<hhmm> -> EDYSRV will override the corporate
        policy and run the branch update process at the specified time.
/R:E<mmmmm>-> EDYSRV will override the corporate
        policy and run the branch update process at  IPL time, and
        every mmmmm minutes subsequently.
```

EDYFREE is the command to unload a running EDYSRV. Its invocation syntax is :

```
EDYFREE netbiosname
```

MIGRADB

This REXX command can be used to migrate a SecureEntry 2.0 users and components database to the new SecureEntry 3.0 library. It must be run in the Domain controller (Where the SecureEntry 2.0 components were stored). To use it, you should have installed SecureEntry 3.0 over the old SecureEntry 2.0, and make sure that the environment variable SGM_LS is still defined and pointing to the directory structure for the users database of the old installation.

The syntax for this command is :

```
MIGRADB  SECP|BOTH
```

SECP

Use this option just to migrate the security components, assuming you are still using the groups and users definitions of the Lan Server UPM. (Working under Lan Server environment).

BOTH

Use this option if you have installed SecureEntry 3.0 in standalone mode, and want to redefine users and groups, or if you have erased the Lan Server users and groups that you had defined.

EDYLOGFS

The EDYLOGFS program is an utility that allows you to control the size of SecureEntry Log Files. This program is located in the SecureEntryPath\EXEC directory.

It gets all its input from an ASCII file called EDYLOGS.STR located in the SecureEntryPath\nouser directory. Each line of this file may specify a way to locate a SecureEntry log file and a size control policy that will be applied to this log file. As a first approach, a line of this file is interpreted like this:

```
where_to_locate_the_log_file    which_size_control_policy_to_apply
```

Locating SecureEntry Log Files

All non-blank nor comment lines in the EDYLOGS.STR file must begin with a pathname to a SecureEntry log file that ends with the base SecureEntry log file name. This pathname can be interpreted in three different ways depending on the pathname's header:

`x$...`

Absolute pathname starting from the x drive's root

`boot...`

Absolute pathname starting from the boot drive's root

`\...`

Relative pathname to SecureEntryPath

Optionally, an environment variable can also be stated after the pathname. If so, this environment variable is used first (and if successfully last) to try to locate the SecureEntry log file. To do so it is interpreted as a pointer to the complete pathname of a directory where the SecureEntry log file should be found.

Size Control Policies

All non-blank nor comment lines in the EDYLOGS.STR file must end stating which size control policy should be applied to the SecureEntry log file specified at the begin of the same line.

EDYLOGFS utility offers two ways for restricting the SecureEntry log files size:

By Size (in Kb)

To set a size control policy based upon file size in Kbs on a particular file, the user states two numbers optionally preceded by an 'S' at the end of the EDYLOGS.STR line that references this particular file. Like this:

```
c$\myfile.log          MY_ENV_VAR          S64,32
```

These two numbers are interpreted (from left to right) as:

The Base File (or Trunc File) Size in Kb (**BFS**)

The maximum allowed Increment in Kb over **BFS**. (**IBFS**)

Note: the second number,
IBFS, is not mandatory. Its default value is **BFS**/2

From them it is derived the maximum allowed size for the specified log file which is **BFS**+**IBFS** Kb. If EDYLOGFS finds the log file to have a size greater than this value it will trunc the log file to have a size as near as possible to **BFS** Kb assuring no text line in the file gets trunked, and it will do so in a FIFO way, that is, deleting the data that was first written to the file.

To conclude here are some examples that set size restrictions based upon file's size in Kb:

```
c$\myfile.log          MY_ENV_VAR          64
c$\myfile.log          MY_ENV_VAR          s64,32
boot\myfile.log                           S23
\nouser\myfile.log                        56,10
```

By number of lines

To set a size control policy based upon the number of lines on a particular file, the user states two numbers preceded by an 'L' at the end of the EDYLOGS.STR line that references this particular file. Like this:

```
boot\myfile.log                                L100,10
```

These two numbers are interpreted (from left to right) as:

The Base (or Trunc) Number of Lines for the file (**BNL**)

The maximum allowed Increment in lines over **BNL** for the file (**IBNL**)

Note: the second number,
IBNL, is not mandatory. It's default value is **BNL**/2

From them it is derived the maximum number of lines the specified log file can have, which is **BNL**+**IBNL** . If EDYLOGFS finds the log file to have a number of lines greater than this value it will trunc the log file to have exactly **BNL** lines, and it will do so in a FIFO way, that is, deleting the lines that were first written to the file.

The Input File for EDYLOGFS utility.

EDYLOGFS utility reads its input from the ASCII file EDYLOGS.STR located in the SecureEntryPath\nouser directory. Each non-blank nor comment line of this file states a file size control policy that is to be applied to a SecureEntry log file.

EDYLOGS.STR Syntax Specification

What follows is a grammar specifying the syntax of the EDYLOGS.STR file:

(**<LINE>** | **<COMMENT>**) ('\n')*

<COMMENT>

= ';' (any string)

<LINE>

= (<blanks>|<null>) **<PATHNAME>** ((<blanks> **<ENV_VAR>**) | <null>) <blanks> **<SIZE_CTRL_POLICY>** (<blanks>|<null>|**<COMMENT>**)

<null>

= the null string

<blanks>

= ('_'|'\t')+

<PATHNAME>

= (**<BOOT>** | **<DRIVE>** | <null>) <os/2 pathname> .br /*Specifies a pathname to a SecureEntry log file*/

<ENV_VAR>

= any valid string for an environment variable .br /*Specifies an environment variable which points to a pathname of a SecureEntry log file*/

<SIZE_CTRL_POLICY>

= (**<POL_CHAR>** | <null>) **<TRUNC_SIZE>** (<null> | (',' **<OFFSET>**))

<BOOT>

= any upper/lower case of the word 'BOOT'.

<DRIVE>

= <char> '$'

<os/2 pathname>

= any valid os/2 pathname with no drive specification and terminated by a base file name

<POL_CHAR>

= 'S'|'s'|'L'|'l'

<TRUNC_SIZE>

= <num>

<OFFSET>

= <num>

<char>

= ('a'-'z') U ('A'-'Z')

<num>

= any integer number

Note: <num>,
<ENV_VAR> and <os/2 pathname> tokens have size restrictions. If any of this tokens has more characters than allowed, a syntax error will be returned.

A sample EDYLOGS.STR file

This is an EDYLOGS.STR file example:

```
;SENTRY LOG FILES SIZE CONTROLLER EXAMPLE FILE

;These blank lines are ignored.

\install\myfile.log       64
;   Maximum size 128 Kb. Would be  trunked to 64 Kb.
;   File "myfile.log" would be searched in the  SecureEntryPath\install  directory

c$\install\myfile.log    L10
;  Maximum number of lines: 15. Would be  trunked to 10 lines.
;  File "myfile.log" would be searched in the  c:\install directory.

c$colon.myfile.log       MY_ENV_VAR    l100,50
;  Maximum number of lines: 150. Would be  trunked to 100 lines.
;  File "myfile.log" would be searched first in the directory pointed by
;  MY_ENV_VAR environment variable, and, only if not found, would be
;  searched in the c drive root directory.
```

The default EDYLOGS.STR file shipped with SecureEntry.

The default EDYLOGFS.STR file shipped with SecureEntry is as follows:

```
;SENTRY LOG FILES SIZE CONTROLLER CONFIGURATION FILE

\install\sentry.log                                    64,64 ;SENTRY INSTALLATION LOG FILE
boot\edylkini.log                    SGM_INI_LOGPATH   64,64 ;STARTUP MESSAGES LOG FILE
\nouser\edyadmin.log                 SGM_SL_LOGPATH     64,64 ;SENTRY ADMINISTRATION LOG FILE
boot\os2\security\sesdb\edysla.log   SGM_SES_LOGPATH   64,64 ;SENTRY SESSION ACTIVITY LOG
FILE
```

After applying a service to your SecureEntry version the new default file EDYLOGS.STR will be copied into the SecureEntryPath\EXEC directory. If the file SecureEntryPath\NOUSER\EDYLOGS.STR has not been modified for your installation, it will be then automatically updated; otherwise you will have to manually update it in order to make the changes effective.

Command Line Syntax

The syntax to call EDYLOGFS utility from the command line is:

```
EDYLOGFS [(/V|/?)]
```

/V	Verbose mode.
/?	Display usage help

Installing EDYLOGFS

There are two possible places for a call to EDYLOGFS :

Within the config.sys file. This is the default place and as is setup in a regular SecureEntry installation.

To automatically control log file sizes at boot time it is recommended to call EDYLOGFS.EXE from the CONFIG.SYS file. A way to do this is to add the following line to your CONFIG.SYS file:

```
CALL=SecureEntryPath\exec\edylogfs.exe
```

A second place where you could place a call to this utility is within any user exit. SecureEntry grants that its log files are closed at this time and thus the files can be truncated.

Error Return Codes

EDYLOGFS utility returns a zero on successful execution, otherwise something went very wrong and a negative integer related to the cause of the problem is returned.

What follows is a list of the error codes that EDYLOGFS utility might return:

-1

The SecureEntryPath could not be found. Either the environment variable SGM_SHELL was not defined or pointed to an invalid directory

-2

The input file (EDYLOGS.STR) could not be opened or didn't exist in the SecureEntryPath\nouser directory

-3

I/O error on the input file (EDYLOGS.STR)

-4

The input file (EDYLOGS.STR) could not be closed

-5

The process EDYLOGFS.EXE could not allocate memory

-6

Program flow reached some point presumed to be unreachable

EDYLOGBR

The **EDYLOGBR** program can be used to browse interactively the following SecureEntry log files, conveniently merged by event timestamp :

The session event log file

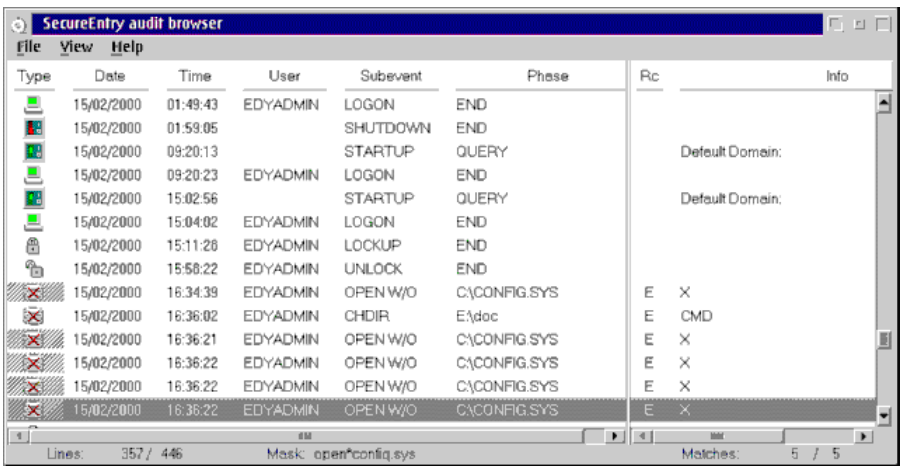
The administration log file

The Treelock audit file, if one is present in the machine

It can be run from a command line, by typing

```
EDYLOGBR
```

or by launching the appropriate object in the *SecureEntry workbench*, *installation tools* folder :



Once started, it will display the selected log files records, for your review, as choosed from the *View* menu :

Errors/denials Will show only records logged for events that terminated in an error condition or a denied access.

Simplified view Will show all of the log lines, but skip event start records as well as Lan Server administration events, which are redundant since the same record is also logged normally as a SecureEntry subsystem administration event.

Full view Will show all the current logs information.

Event log Use this menu selection to include/exclude those records coming from the SecureEntry session event log file. This log tracks session flow events such as logon, lockup,...

Administration log Use this menu selection to include/exclude those records coming from the SecureEntry administration log file. This log tracks all of the activity issued against the SecureEntry administration system.

Audit log Use this menu selection to include/exclude those records coming from the SecureEntry Treelock audit file, if one is present in the *NOUSER* directory as a machine default profile. This log tracks access attempts to directories/files without the required permission.

Show icons Can be used to switch on/off the icon view from the result container.

Refresh now can be used to force re-reading of the selected log files.

The *File* menu has the following options :

Export Allows you to save, in ASCII format, the displayed records.

Find You can filter off any records which do not include the chosen *mask* with this option. Note that wildcards are allowed and you should assume that the scanned string for each record is the concatenation of all the column values for the intended record. So, for instance issuing a find request with the mask : *smith*add* will find all records that refer to an administration *add* operation issued by userid *smith*.

Find Next Will find the next matching record, given the current find *mask* starting at the current selected record in the container.

Find Previous Will find the previous matching record, given the current find *mask* starting at the current selected record in the container.

Quit Quits the application.

Now, some important notes about this utility :

First, take into account that log files are normally purged (old lines removed) by the log files size controller utility at machine startup, so it may be the case, when searching for very old events, that those are no longer present in the log files.

The utility uses date and time formats as specified in the country object of the OS/2 configuration folder, so it may render uncorrectly sorted records if this configuration setting is changed for previously existing log records.

For last, remember that the scope for the event and audit logs is local, that is, you will only find events there that originated in the current machine. However, and for administration events, the scope (for LAN installations) is at a domain level, since there is a single edministration log file in the domain server.

EDYPHOTO

The **EDYPHOTO** programs allow you to create and view a binary file with the necessary installation and configuration files for problem determination and solving. Both the photo taker and the photo display programs can be accessed from the **SecureEntry Installation tools** folder, or started manually.

Double click over the **photo generator** icon to get a file named **MMDDhhmm.BIN** which contains a copy of the machine configuration files. Alternatively, you can generate a configuration photo file by typing :

```
EDYPHOTO
```

By typing the parameter '?' you can obtain a brief description of the complete syntax and parameters for this utility.

Drop the previously generated file into the **Photo viewer icon** to view its contents. Alternatively, you can invoke the photo displayer by typing :

```
EDYPHDSP  [filename]
```

Note that photo archives are normally left with read-only attribute, so that they can not be mistakenly erased.

The trace server

The trace server is provided so that applications can record trace data.

When applications run, information is gathered that can be used either for debugging purposes at development stage or to diagnose and fix problems at production time.

Operation modes

Changing operation mode

Loading the trace server

Operation modes

The trace server supports these modes of operation:

Tracing to memory (memory status)

Tracing to file (file status)

In addition, the trace server can be loaded but not operative (stop status).

To minimize the trace overhead, it is recommended to load the trace server in not operative mode and change the operation mode dynamically.

Tracing to memory

The EDYTRDSP.EXE program is provided to display trace data on the screen.

The contents of the trace panel are updated only upon request. To refresh the contents, select the appropriate action from the View pull-down menu or press the F5 function key.

The font of the panel contents can be modified by dropping a new font from the font palette. You should select a non-proportional font type; for example, system monospaced, system vio, or courier.

Tracing to file

The EDYTRCS.EXE program is provided to save trace data in a file.

You can display the records in the trace file by using the EDYTRDSP.EXE program.

Changing operation mode

The EDYTRSET.EXE program is provided to change the operation mode of the trace server.

To start this program, enter:

```
EDYTRSET  [/S:status]
```

Where status is the mode of operation to which the trace server will be set:

M : Trace to memory

F : Trace to file

S : Not operative

The F parameter value applies only if the EDYTRCS.EXE program is loaded.

Loading the trace server

This section explains the trace server loading statement and the load time return codes.

If you specify that the trace server stores trace data to a file and this trace file already exists, the loading parameters will not take effect. In addition, trace data will be appended to the existing trace file.

Loading statement

The following diagram shows the loading statement of the EDYTRCS program. To catch errors corresponding to this program that may happen at load time **or** at run-time, you can redirect the output to a file.

```
DETACH EDYTRCS [/S:status] [/M:buffersize]
               [/T:filesize] [/PT:drive:path\filename]
               [/PC:drive:path] [/N:tracenumber]
```

where :

Status is the mode of operation (MEMORY, FILE or STOP). FILE is the default.

Buffersize is the Size of the memory buffer, in KB, where trace data is stored.

The parameter applies only if the EDYTRCS.EXE program is loaded before any application that records trace data. The parameter value ranges from 4 to 512. The default is **64**.

Filesize is the maximum size of the file, in KB, where trace data is stored. The file header should not be included.

The parameter value ranges from 64 to the available disk space. The default is **150**.

/PT and /PC Specifies the full path name of the trace file (optional). Default name is EDYTRC.DAT and located where the EDYTRACE.INI file is located.

Tracenumber is the number of valid traces to be kept after a stop condition is fulfilled.

The parameter value ranges from 0 to 512. The default is **10**.

Loading return codes

The following list shows the codes that can be returned at load time:

```
Value     meaning
----------------
x'5A'     Error creating or opening trace file.
x'5C'     Invalid parameter specified.
x'5D'     Insufficient disk space for trace file.
x'5F'     Trace file damaged.
x'61'     Error initializing trace server.
x'62'     Trace server already loaded
```

SecureEntry technical information

This chapter contains a description of SecureEntry 3.0 used resources, environment variables, logs and configuration files.

SecureEntry directory structure

Environment variables

Configuration files

Log files

About processes and running contexts

About Object IDs

Session process description

Fine tuning SecureEntry

Error codes and messages

SecureEntry directory structure

The SecureEntry directory includes the following subdirectories :

The API directory, where the source code for implementing API calls as well as NLS translation is located.

The DLL directory, where all SecureEntry DLLs reside.

The EXEC directory, where the command line interfaces and executables are located.

The TOOLS directory, where the SecureEntry tools (integration programs) are located.

The NOUSER directory, where the machine configuration data as well as the security profiles repository is placed.

The WORK directory, where the assigned security profiles are downloaded at logon time from the repository server.

The HELP directory, which holds the required help files.

The TEMPLATE directory, where the basic profiles templates as well as component samples are located.

You may have also a TEMP directory, used by the administration tools to do temporary copies of the repository files.

And by last, with a NLS dependant name, there is the workbench directory, where the workbench objects are located.

Environment variables

The following environment variables are used by SecureEntry :

General purpose

Lan Server behavior specific

Related to session control

UCM specific

Administration specific

Related to Workplace Shell

Related to other components

General purpose

SGM_SHELL This variable is mandatory and must be defined in your config.sys file. It should be defined as a path to where the SecureEntry files have been installed. By default, and unless otherwise specified, it points to the C:SGMSHELL directory.

SGM_DB This environment variable is also mandatory, and must be defined in your config.sys file. It points to the directory where the SecureEntry profiles database (file EDYREGDB.VLB) is located. In network environments, it can be a UNC path.

Lan Server behavior specific

SGM_LS This variable is obsolete, and not generally used, but it must be set to its original value if the MIGRADB utility is to be run in order to be able to locate the original (SecureEntry 2.0) security profiles.

SGM_LS_IFLOGGED This environment variable sets the action to do for Lan Server environments, when at logon time a domain logon is found in the machine. The options are :

```
SET SGM_LS_IFLOGGED=INFORM    :  Do not logon. Display error message (default)
SET SGM_LS_IFLOGGED=FORCE     :  Force a logoff and retry logon automatically
SET SGM_LS_IFLOGGED=USE       :  Use the logged on session if the  userid and
                                 domain found already logged match the
                                 requested ones. Inform otherwise.
SET SGM_LS_IFLOGGED=FORCEUSE  :  Use the logged on session if the  userid and
                                 domain found already logged match the
                                 requested ones. Force a  logoff and retry
                                 otherwise.
```

Note that this variable can prove very useful for integrating SecureEntry with other security products, so that the logon responsibility can be shared.

SGM_SGMSHELL_GROUP SecureEntry/2 does internally provide shared access to its repository for Lan Server clients through the hidden resource *SGMSHELL*. Since its normal behavior is to grant such access for all SecureEntry groups (those that start with letters 'SG'), it may be the case, if you have or plan on having more than 64 such groups that the ACL (Access Control List) for the SGMSHELL resource becomes too small, since its limit is 64 entries.

To overcome such limit, this environment variable can be used. By specifying it, you are effectively defining the name of the group which should have access to the SGMSHELL resource, and that way, telling SecureEntry not to update the ACL for this resource with accesses for each group beginning with letters 'SG'.

So, if this variable has value 'USERS', all the users defined in the Lan Server domain will be able to issue a SecureEntry signon. If it has another value, then you are responsible to make sure that all SecureEntry users belong to such group AND to the SecureEntry group ('SG'), in order to be able to signon.

Note that the access to the SGMSHELL resource will be granted only after at least one SecureEntry group has been redefined (added) or modified (updated) with this environment variable in place.

Related to session control

SGM_EDYLK_SHOW This variable can be used to force the startup dialog to be kept hidden even if a EDYSTART.CMD file can be found within the root directory of the boot drive, by assigning it a value of **NO** or **0**.

SGM_HIDE_WAIT_DLGS Use this environment variable set to **YES** to avoid showing of the 'xxxx in progress, please wait...' dialogs. If this variable is not defined or set to other value, these dialogs will be displayed.

SGM_INI_LOGPATH This environment variable sets up the path for the EDYLKINI.LOG file, which keeps information about the startup process activity. It is not set by the installation process. By default, if this variable is not set, then this log file will be placed within the root directory of the boot drive, that is, the same directory where the EDYSTART.CMD file resides.

SGM_SES_LOGPATH This environment variable sets up the path for the EDYSLA.LOG file, which keeps information about all of the machine's session activity. It is not set by the installation process. By default, if this variable is not set, then this log file will be placed within the path pointed by the SESDBPATH variable (normally being the OS2\SECURITY\SESDB path of the boot partition). If SES is not installed, then the file will be stored in the SGMSHELL\NOUSER directory, unless otherwise specified through the appropriate environment variable.

SGM_WPS_FASTLOAD This variable is not mandatory. It determines the activation sequence for Desktop restrictions at WPS load time. The WPS is loaded when the workstation is started, after an error causes it to terminate, and after logging on when the SES environment variable 'RESTARTUSERSHELL' is set to **YES**.

If fast loading is disabled, the Desktop restrictions defined in NOUSER are applied when the WPS is being loaded. Once the WPS is fully initialized, the restrictions for all the other SecureEntry components are applied; then, the user's Desktop restrictions are applied.

If fast loading is enabled, the user Desktop restrictions are applied as soon as they are available (user identified); then, the restrictions for all the other SecureEntry components are applied. This means that the user can access the Desktop before additional restrictions are applied. Fast loading can affect security if the user's Desktop restrictions allow the user to manipulate desktop objects that should only be available when all the other restrictions have been applied (e.g. starting an editor before the File System restrictions have been applied).

A second effect of this variable is to specify 'when to rise the curtain' at logon time, i.e, when will the user be able to use his desktop. If enabled, the background bitmap will be closed as soon as the user is

identified. If disabled, this will be delayed until all security profiles are active, thus not allowing user interaction with the machine until that moment.

By default, fast loading is disabled. To enable it, set this variable to **YES** or to any other value that starts with **Y**.

SGM_ALLOW_CAD Use this variable set to **YES** to allow one-touch Ctl-alt-del to reboot the system, as any non SecureEntry OS/2 system does. By default, the value of **NO** will enable SecureEntry specific support for this key combination. The third allowed value is **TWICE**, which will force twice-touch of the key combination to effectively reboot the machine. This is useful for obtaining system dumps, since then Ctl-alt-del followed by Ctl-alt-NumLock-NumLock will force the dump.

SGM_PM_WAIT_B4_KILL Use this environment variable to specify a wait interval (in seconds) before SecureEntry decides to kill user context PM running applications at logoff time after sending them the close message. This becomes handy if you want to grant the logoff function to terminate even if applications request for any user input when receiving the WM_CLOSE PM message. So, for instance, if you are editing a file, and then decide to logoff, the typical editor behavior would be to prompt about saving or discarding changes at this time. Once the specified timeout expires without further user interaction, the editor will be killed and the logoff task resumed.

The default value is 20 seconds. Use a value of -1 to indicate an indefinite timeout. Use a value of 0 to indicate no timeout, i.e, kill applications immediately. In any case, and to avoid problems, it is advisable to have this variable set to a value different from 0, since some processes do not die 'elegantly' if they do not receive the WM_CLOSE message and/or some data could be lost if they do not have an opportunity to save it.

SET SGM_WAIT_B4_FLUSH Set this variable to xxx, where xxx is the number of seconds to wait before flushing system buffers to disk at shutdown time in case of a shutdown hang.

This means that a system power off will not imply a chkdsk during the next IPL if xxx seconds have passed from the beginning of WinShutdownSystem.

If "edyutil { toshtdwn | frshtdwn } reboot" was specified, then the reboot will take place almost immediately if it succeeds, or after xxx seconds have passed since the beginning of WinShutdownSystem.

If not specified, xxx defaults to 300 seconds (5 minutes).

If xxx==0, then 300 seconds (5 minutes) will be set.

```
Format:

o If xxx begins with 0, then the value will be assumed to be octal
o If xxx begins with 0x or 0X, then the value will be assumed to be hexadecimal.
o Otherwise the value will be assumed to be decimal.
```

SGM_SS_ALLOW_IU This environment variable does only make sense when using RACF (or the RACF emulator) to synchronize passwords. It allows you to customize the behavior of the signon process when *Unknown user* is received from such logon procedure. The valid values are :

> **0** (default) Abort the signon process presenting to the user the appropriate error message dialog box.
>
> **1** Gate to emergency logon *without* synchronizing passwords, so that if the userid/password are valid at the branch, signon will succeed. Note that password changes will **not** be allowed.

2 Continue signon process *without* synchronizing passwords, so that if the used userid/password are valid at the branch, signon will succeed. Password changes are allowed.

3 Continue signon process *without* synchronizing passwords, so that if the used userid/password are valid at the branch, signon will succeed. Password changes are allowed. Additionally, present the user with a dialog box message informing about the event (message number 106).

Note that using values different than 0 may cause userid/password desynchronization problems, but may be a requirement for local administration at your corporation.

SGM_SS_ALLOW_IP This environment variable does only make sense when using RACF (or the RACF emulator) to synchronize passwords. It allows you to customize the behavior of the signon process when *Invalid password* is received from such logon procedure. The valid values are :

0 (default) Abort the signon process presenting to the user the appropriate error message dialog box.

1 Gate to emergency logon *without* synchronizing passwords, so that if the password is valid in the branch, signon will succeed. Note that password changes will **not** be allowed.

2 Continue signon process *without* synchronizing passwords, so that if the used password is valid at the branch, signon will succeed. Password changes are allowed.

3 Continue signon process *without* synchronizing passwords, so that if the used password is valid at the branch, signon will succeed. Password changes are allowed. Additionally, present the user with a dialog box message informing about the event (message number 106).

Note that using values different than 0 may cause password desynchronization problems, but may be a requirement for local administration at your corporation.

SGM_SS_IF_NO_AUTOLOCKUP Set this variable to **NO** if you want the screen saver function disabled when there is no autolockup defined through the active SES restrictions profile. i.e, you only want the screen saving function to activate over a lockup screen, and not over an unprotected user desktop.

SGM_SS_USEREXIT Set this variable to **NO** if you want to avoid the screen saving function to be active while processing user exits. Note that in this case, and if you provide your own logon or unlock dialogs, they will be responsible for avoiding losing the focus and should provide their own screen saving function.

SGM_SS_WHEN_LOCKUP Set this variable to **YES** if you want the screen saving function to activate immediately when a lockup event is processed, because of explicit user request or inactivity timeout. The customized (security profile) screen saver inactivity timer will still be used for :

Popping up the screen saving function while not in lockup state (if automatic lockup is not being used), or

Popping up the screen saver function while in lockup state after the user forces the lockup panel to appear by issuing some keystrokes.

SGM_BACK_BITMAP Set this variable to **NO** if you want to disable displaying of the background bitmap during the logon, logoff and lockup processes. This is a feature not normally needed, but which may prove useful in those installations where SecureEntry has to be integrated with existing applications in a seamless way.

If you require more granularity, this same variable accepts an alternate syntax, as follows:

```
SET SGM_BACK_BITMAP=xyz
```

Where:

x specifies where to display the background bitmap at startup.

y specifies where to display the background bitmap during logon/logoff.

z specifies where to display the background bitmap during lockup.

Specify all values x, y and z as character '0' (disable) or '1' (enable).

SGM_DISABLE_SYSTEM_KEYS Set this variable to a number in order to specify a given systemwide key to disable. the number can be any combination (addition) of :

```
Ctrl-Alt-Del ..............  1
 Ctrl-Alt-NumlockNumlock  ... 2
Ctrl-Esc ..................  4
Alt-Esc ...................  8
Alt-Tab ................... 16
Pause ..................... 32
Print-Screen .............. 64
Left Windows key ........ 128
Right Windows key ....... 256
Windows select key ...... 512
```

You can specify octal notation, by beginning the value with '0', hexadecimal, beginning with '0x', or decimal (otherwise). Note that for this feature to work correctly, you need to have a fully compatible IBM-PS2 type keyboard.

If you configure a given key combination to be disabled through this environment variable, and also to be enabled by the active EDYSES profile, then the key will be disabled since a bitwise OR is done among both masks to decide what to disable or not

The default value is 0 (do not disable system keys).

SGM_HOOK_SYSTEM_KEYS

This environment variable allows you to configure which of the supported systemwide keys are to be hooked through a user exit. Allowed values are the same as those of SGM_DISABLE_SYSTEM_KEYS. Note that if a given key combination is disabled, then even if it is also configured to be hooked, it will not be seen by the hook (user exit). If you configure a given key combination to be hooked through this environment variable, and also to not be hooked by the active EDYSES profile, then the hook will be active since a bitwise OR is done among both masks to decide what to hook or not

The default value is 0 (do not hook system keys).

SGM_SES_CAD

Use this variable set to **YES** to force the system to handle Ctrl-Alt-Del Requests through the SES API, instead of using SecureEntry's own device driver, which is the default behavior. Using SecureEntry device driver to check for CAD combination has the advantage that Ctrl-Alt-Del is correctly distinguished from Ctrl-Alt-Numlock, but could not work if you are working with non-standard keyboard devices.

SGM_SES_INACTIVITY

Use this variable set to **YES** to force the system to handle inactivity timeouts through the SES API, instead of using SecureEntry's own device driver, which is the default behavior. Using SecureEntry device driver to check for inactivity has the advantage of increased accuracy, but could not work if you apply a OS/2 base fixpack for which this feature has not been tested.

SGM_HOOK_CANN_KEY

Use this environment variable to force SecureEntry *not* to internally hook the Ctrl-Alt-NumLockNumLock key combination, by setting it to a *NO* value. Note that SecureEntry does always by default handle this key combination internally, to avoid potential problems when trying to obtain a memory dump over a FAT partition in a disk where SecureEntry boot protection is installed, since the memory dump process needs to have direct BIOS access to the partition.

The drawback of this approach is that then and since the key combination is intercepted by SecureEntry, such obtained system dumps do always point CS:EIP to SecureEntry's EDYSLA.EXE kernel code, masking off the real responsible module which caused the necessity of a dump in the first place.

Summarizing, set this variable temporarily to *no*, if you want to obtain a clean memory dump, but in this case you should either uninstall the boot protection feature first, or be careful not to generate the dump into a FAT partition.

SGM_USER_DLGS

If you are Overriding the default SecureEntry logon/lockup dialogs, then you can use this environment variable to specify the name of your executable modules or dialog titles which implement such overridden windows. These will be used to identify which windows are allowed to steal system modality over the lockup background bitmap when the appropriate user exit is serviced. If this environment variable is not used, then SecureEntry will assume that the first system modal window that appears during user exit processing is the intended one, thus allowing for a few seconds hole where an asynchronous system modal window could be assumed to be the expected logon/lockup one.

Specify as many process names or window titles (enclosed between double quotes) as you need, separated by commas, case insensitive and with wildcards allowed. For instance, and if you are using the provided Sample logon/unlock dialogs, you could specify:

```
SET SGM_USER_DLGS=*LOGSAMP.EXE,*UNSAMP.EXE
```

SGM_HIDE_EXIT_AFTER_LOGON

Use this environment variable to specify whether SecureEntry should keep the logon background bitmap in place while the user exit after logon is being processed (value **YES**), or leave this user exit to be processed while the desktop is accessible (value **NO**). Note that the installation process sets this variable to **YES**, to grant workstation security, since treelock profiles enforcement is not active while this user exit is running.

SGM_NC This environment variable is used to specify the **NSC** subsystems that will act as password synchronizers when using the provided NSC/2 LMP. It will accept the following values:

> *ALL*. All systems defined in the **NSC** profile will act as password synchronizer subsystems. This is the default value.

The indexes, separated by commas, of the subsystems (LOCAL, SERVER, LANSERVER, HOST) that are defined in the **NSC** configuration file that you want to act as synchronizer subsystems. For example:

SGM_NC=2,3,6

In this case the subsystems defined in second, third and sixth position within the **NSC** configuration file will act as synchronizer subsystems.

SGM_SHUTDOWN_AT_LOGON_PANEL

Use this environment variable set to **NO** if you want to disable the *Shutdown* button in the standard SecureEntry logon dialog, for instance, because the machine is a server and you want to avoid casual shutdowns initiated by unauthorized users. The default value if unspecified is **YES**.

UCM specific

SGM_UCM_ENABLE This variable is to be defined only when UCM is being used, in the centralized administration workstation. When set to **YES**, all administration API requests are rerouted to the host through the DB2/DDCS UCM agent. Note that you can set this variable in a per administration session basis (i.e, not config.sys mandatory).

SGM_UCM_BRANCH This variable is to be defined only when UCM is being used at the central UCM administrator workstation. It defines the default branch name to be used by the administration utilities when adding a new user to the corporate repository.

SGM_UCM_THIS_BRANCH This environment variable defines the branch name by which this actual branch will be known at the host site. This is only for informative purposes, so you can relate a given branch ID, which is given automatically by the host, with a full sense identification string. Note that this environment variable will only be really used once, whenever a new branch is known to the host for the first time, so if you intend to use it, you will have to set it up during the installation process of your new branch servers. Once installed, the utility **EDYBRNVW.EXE**, can be used at any workstation to view the branch synchronization and identification data.

EDY_UCM_MAXROWS This variable can be defined to customize the maximum number of rows that the UCM API will be able to handle as a query result before returning the MANY_ROWS_FETCHED error. The default value not to impose any limit.

SGM_REFRESH_PARMS This environment variable its not mandatory. This variable should be defined in the config.sys file. Sets up the number of retries for the branch refresh task in case of error, the time interval between this repeat operations and the minutes interval to be used to calculate the init hour of refresh branch. This variable can take the following allowable values: SGM_REFRESH_PARMS=a,b,c with this ranges: 0 to 9, 0 to 99 and 0 to 180 for a, b, c respectively. The default values if SGM_REFRESH_PARMS not specified is 1,15,30.

SGM_UCM_LOGPATH This environment variable sets up the path for the EDYDIS.LOG file, which keeps information about the refresh activity of a branch. It is not set by the installation process. By default, if this variable is not set, then this log will be placed within then path pointed by the SGM_DB variable (normally the server machine SecureEntryPath\NOUSER directory).

SGM_NETBIOS_ADAPTER_NUM If your desired netbios protocol is not configured over logical adapter 0 through MPTS, you can use this variable to indicate which logical adapter to use for

communicating between the EDYSRV and the client machines. This communication is specially relevant when using UCM or RACF validation. Valid values are 0,1,2 and 3.

SGM_FORCE_LUALIAS Use this environment variable to specify the independent LU 6.2 alias to use for EDYSRV/EDYTP communications. If not specified, (which is the default case), the default configured independent LU 6.2 in communications manager will be used.

SGM_UCM_DBDFT Only for the UCM's administrator workstation. If you catalog the UCM Database in the administrator workstation with an alias different than 'UCM', then you must define this environment environment variable with value: the UCM choosen alias.

SGM_UCM_CORPORATE_NAME Use this environment variable in the UCM administrator workstation only. If you install UCM in a multi-enterprise corporation with a separate UCM Database for each company and you wish to be able issue all UCM administrator's tasks from **a single workstation**, you must customize this variable as explained below:

Before you proceed with SecureEntry and UCM installation, you must decide the institution name to use for each company. This name will be required by SecureEntry and UCM installation processes. If you use the same name for each company then you will not need to use this environment variable.

After installing SecureEntry in the UCM administrator's workstation, make as many object copies as needed of the *UCM, Users and groups administration tool* (as many as different existing UCM databases you wish to administer). Make the same number of copies of the file UCMADM.CMD located in **\SGMSHELL\EXEC** (i.e, UCMADM**1**.CMD..UCMADM**n**.CMD ...).

Then set this environment variable for each *UCMADM**x**.CMD* file with the appropriate institution name you want to administer. Make sure also that each command file connects to the appropriate DB2 database alias.

Now assign each command file to each of the copied *UCM, Users and groups administration tool* object by directly editing the object's settings.

SGM_FORCE_MODE Only for the domain controller workstations in your branches. If you want to define another APPCMODE for the UCM communications channel, you must define the APPCMODE's name in this environment variable. This can be very useful in RACF and UCM installations. If you use the RACF emulator provided by SecureEntry, this environment variable is not needed.

SGM_SNA_TIMEOUT Use this environment variable to specify, in seconds, an increased maximum timeout for APPC host responses. The default and minimum allowable value is 180 seconds.

Administration specific

SGM_SL_LOGMODE This variable is not mandatory, and can be used to set up the logging mode for administration utilities. Its possible values are :

```
SET SGM_SL_LOGMODE=TEST     : Log all activity, including returned keyword
                              values in view operations. Useful for testing
                              your own developed agents
SET SGM_SL_LOGMODE=ALL      : Log all activity
SET SGM_SL_LOGMODE=UPDATES  : Log only activity which modifies the database.
                              This is the default setting
SET SGM_SL_LOGMODE=NONE     : Log no activity
```

SGM_SL_LOGPATH This environment variable sets up the path for the EDYADMIN.LOG file, which keeps information about all administration activity. It is not set by the installation process. By default, if this variable is not set, then this log file will be placed within the path pointed by the SGM_DB variable (normally the server machine SecureEntryPath\NOUSER directory.

SGM_ADM_PRIV This environment variable can be used to filter which administration functions are allowed by the agents selector for administrator users. You can specify it by setting this variable to an integer value where each bit specifies a given operation, either in decimal, hex (0x...) or binary (0b...) :

```
EDYUCM_PRIVILEGE_SUB_VIEW          0x00000001
EDYUCM_PRIVILEGE_SUB_ADD           0x00000002
EDYUCM_PRIVILEGE_SUB_UPDATE        0x00000004
EDYUCM_PRIVILEGE_SUB_DELETE        0x00000008
EDYUCM_PRIVILEGE_RESOURCE_VIEW     0x00000010
EDYUCM_PRIVILEGE_RESOURCE_ADD      0x00000020
EDYUCM_PRIVILEGE_RESOURCE_UPDATE   0x00000040
EDYUCM_PRIVILEGE_RESOURCE_DELETE   0x00000080
EDYUCM_PRIVILEGE_GROUP_VIEW        0x00000100
EDYUCM_PRIVILEGE_GROUP_ADD         0x00000200
EDYUCM_PRIVILEGE_GROUP_UPDATE      0x00000400
EDYUCM_PRIVILEGE_GROUP_DELETE      0x00000800
EDYUCM_PRIVILEGE_USER_VIEW         0x00001000
EDYUCM_PRIVILEGE_USER_ADD          0x00002000
EDYUCM_PRIVILEGE_USER_UPDATE       0x00004000
EDYUCM_PRIVILEGE_USER_DELETE       0x00008000
EDYUCM_PRIVILEGE_USER_GRP_VIEW     0x00010000
EDYUCM_PRIVILEGE_USER_GRP_ADD      0x00020000
EDYUCM_PRIVILEGE_USER_GRP_UPDATE   0x00040000
EDYUCM_PRIVILEGE_USER_GRP_DELETE   0x00080000
EDYUCM_PRIVILEGE_RES_USER_VIEW     0x00100000
EDYUCM_PRIVILEGE_RES_USER_ADD      0x00200000
EDYUCM_PRIVILEGE_RES_USER_UPDATE   0x00400000
EDYUCM_PRIVILEGE_RES_USER_DELETE   0x00800000
EDYUCM_PRIVILEGE_GRP_GRP_VIEW      0x01000000
EDYUCM_PRIVILEGE_GRP_GRP_ADD       0x02000000
EDYUCM_PRIVILEGE_GRP_GRP_UPDATE    0x04000000
EDYUCM_PRIVILEGE_GRP_GRP_DELETE    0x08000000
EDYUCM_PRIVILEGE_RES_GRP_VIEW      0x10000000
EDYUCM_PRIVILEGE_RES_GRP_ADD       0x20000000
EDYUCM_PRIVILEGE_RES_GRP_UPDATE    0x40000000
EDYUCM_PRIVILEGE_RES_GRP_DELETE    0x80000000
```

So, for instance :

```
SET SGM_ADM_PRIV=0x11111111
```

Would allow only view access through SecureEntry administration tools, and

```
SET SGM_ADM_PRIV=0x1111F111
```

Would allow view access to all of SecureEntry repository, plus add/update/and delete access for user objects.

Note however, that using this variable only makes sense if you are providing a 'closed' environment to your local branch administrators, i.e, they do not have access to a command line. If this variable is not specified, unlimited access is granted to administrator users.

Related to Workplace Shell

SGM_WPS_LOADCLASS This variable is not mandatory. It can be used to specify WPS classes that need to be loaded before the SecureEntry profiles are activated during the WPS loading. It should only be defined if custom components are defined that include WPS code that needs to be initialized before activating the user profiles. The value of the variable should be a list of class names separated with comas, with no blanks allowed.

SGM_WPS_DISABLE This variable is not mandatory, and should only be used during problem determination. When it is set to **YES**, or to any other value that starts with **Y**, the SecureEntry WPS classes that replace the standard classes are disabled.

SGM_WPS_TRACE This variable is not mandatory, and is only useful during problem determination. It allows to define the level of trace detail of different subsystems in the WPS code. The value must be a list of subsystems with their corresponding level of detail. The defined subsystems and their default level of detail are: RES-3: Object restrictions CLS-3: Class loading OBJ-3: Object loading POS-3: Object position management MGR-3: Profile and event management DRG-3: Drag and Drop SHD-3: Shadow folders MTX-5: Mutex semaphore management ASY-5: Asynchronous events management Examples:

```
If the variable is not defined and the  SNTOBJ trace level is set to 1,
none of this subsystems will trace its actions. If the trace level is
changed to 3, all the subsystems except  MTX and ASY will trace their
Actions.

If the variable is set to  SGM_WPS_TRACE=RES-1,DRG-1,  and the SNTOBJ
trace level is set to 2, the RES and  DRG subsystems will trace their
Actions.
```

SGM_WPS_PRINTJOBS This variable is not mandatory. It defines how restrictions should be applied on print jobs. Its possible values are: RESTRICT_NONE, RESTRICT_BY_TITLE, RESTRICT_BY_DEFAULT or RESTRICT_PRINTJOBS. When it is set to RESTRICT_NONE, no restrictions are applied on print jobs, no matter their title. When it is set to RESTRICT_BY_TITLE, print jobs are restricted with the standard mechanism: if a matching title is found in the Desktop Restrictions profile, the specified restrictions are used; if no matching title is found, the restrictions specified for the DEFAULT entry are used; if there is no DEFAULT entry, no restrictions are applied. When it is set to RESTRICT_BY_DEFAULT, print jobs are restricted with the restrictions specified for the DEFAULT entry in the Desktop Restrictions profile, no matter their title; if there is no DEFAULT entry, no restrictions are applied. When it is set to RESTRICT_PRINTJOBS, all print jobs are treated as if their title were RESTRICT_PRINTJOBS. If the Desktop Restrictions profile contains a RESTRICT_PRINTJOBS entry, its restrictions are used for all print jobs, no matter their title; if there is no RESTRICT_PRINTJOBS , the restrictions specified for the DEFAULT entry are used; if there is no DEFAULT entry, no restrictions are applied. The default value is RESTRICT_BY_TITLE.

SGM_WPS_NONDESKTOP This variable is not mandatory. It defines how restrictions should be applied on objects that are not located on the current Desktop or Nowhere folders or any of their subfolders. Its possible values are: RESTRICT_NONE, RESTRICT_BY_TITLE, RESTRICT_BY_DEFAULT or RESTRICT_NONDESKTOP. To avoid a severe performance degradation, the restrictions applied to these objects only affect their menus and setting pages: no change is done regarding their styles (visible or not, deleteable or not...). If their styles need to be restricted, the standard OS/2 mechanism can be used, either on installation or by means of a user exit if they need to be

changed dynamically (e.g. SysSetObjectData(ObjectPath, "NOTVISIBLE=YES;NODELETE=YES")). When it is set to RESTRICT_NONE, no restrictions are applied on these objects, no matter their title. When it is set to RESTRICT_BY_TITLE, these objects are restricted with the standard mechanism: if a matching title is found in the Desktop Restrictions profile, the specified restrictions are used; if no matching title is found, the restrictions specified for the DEFAULT entry are used; if there is no DEFAULT entry, no restrictions are applied. The use of this value can cause a noticeable delay when opening these objects' settings notebook or when displaying their menu. When it is set to RESTRICT_BY_DEFAULT, these objects are restricted with the restrictions specified for the DEFAULT entry in the Desktop Restrictions profile, no matter their title; if there is no DEFAULT entry, no restrictions are applied. When it is set to RESTRICT_NONDESKTOP, these objects are treated as if their title were RESTRICT_NONDESKTOP. If the Desktop Restrictions profile contains a RESTRICT_NONDESKTOP entry, its restrictions are used for all these objects, no matter their title; if there is no RESTRICT_NONDESKTOP , the restrictions specified for the DEFAULT entry are used; if there is no DEFAULT entry, no restrictions are applied. The default value is RESTRICT_NONE.

SGM_WPS_BEEP This variable is also not mandatory, and can be used to make SecureEntry beep through the speaker at certain times for debugging purposes. The supported values are :

REFRESH : A beep will sound at the beginning and end of a desktop refresh.

EXIT : A beep will sound whenever the WPS dies.

TRAP : Will sound a beep whenever a trap is processed by SecureEntry WPS code (default).

SGM_WPS_SKIP_PREPOPULATE This variable controls whether SecureEntry should inhibit the default folders population done by the OS/2 system at startup. A standard OS/2 WARP system WorkPlace prepopulates the following folders at startup :

<WP_TOOLS>
<WP_GAMES>
<WP_DRIVES>
<WP_CONFIG>
<WP_OS2SYS>
<WP_INFO>
<WP_PROMPTS>
<WP_SERVER>

Doing this task takes several seconds, and it may be very well the case that this set of folders do not belong to the 'most used' ones on your system, in which case it is just spending time without purpose. This together with the fact that up to recent fix levels on OS/2 WARP fix code this 'prepopulate thread' could potentially aggravate an internal timing dependant deadlock problem of workplace at startup, have been the reasons for adding the present environment variable.

The variable can take the following values :

ALL Skip all folders populate at startup

NONE Skip no folders populate at startup (default if not present)

<ObjectId>..<ObjectId> Skip listed folders populate at startup

SecureEntry installs itself setting the **ALL** value in config.sys file.

SGM_EDYSC_DISABLE This variable can be used to disable all the WarpCenter's SENTRY features leaving the SmartCenter Workplace Shell class as original defined. Allowed values are **YES** or **NO**

SGM_WPS_IGNORE_ADMIN You can override SecureEntry specific workplace shell functionality done for administrators by setting this variable to **YES**. Specifically, this refers to the visibility state of the SecureEntry workbench folder and the added folder's popup menu entry 'Save positions'. Note that you can cancel this setting temporarily by executing the provided *EDYWPADM.CMD* command, as supplied within the SecureEntry path, TOOLS directory.

Related to other components

SGM_WIN_EXPLICITMENUS Set this variable to **NO** if you want the windows behavior component to enforce the <DEFAULT> system menu settings for all windows of the system. Note that the default value is set to **YES**, so that system menu settings will only be enforced for explicitly configured windows. Setting the **NO** value will cause system menu options that are managed by the applications to become active/inactive as specified, with unpredictable results unless tested.

SGM_OVER_MK This variable can be used to control what information is used as a seed for floppy encryption with the institution dependant algorithm.

If not specified or has the value **NONE**, then no override is made to the seed calculation algorithm, so that the institution name, configuration type, server name and UCM/RACF flags are used to build up the seed. The net effect is that diskettes encrypted this way will end up being branch specific, as long as the domain server name is unique across branches.

The value **ALL** (default as installed) ensures that only the institution name will be used to generate the encryption seed.

Any other value implies to use the installation fields institution name, configuration type, server name and UCM/RACF flags to generate the master key for floppy encryption, but assuming that the server name is fixed to the one specified through the environment variable.

SecureEntry will set this variable to the value **ALL** at installation time.

WarpCenter Environment Variables.

SGM_ROAM_LOGPATH This environment variable sets up the path for the EDYROAM.LOG file, which keeps information about the Public Applications component in LAN Server environments. It is not set by the installation process. By default, if this variable is not set, this log file will be placed within the subdirectory NOUSER of the path pointed to by the environment variable SGM_SHELL.

Configuration files

SecureEntry 3.0 has some control files which define its configuration and installed version. These are :

SENTRY.SIG This file contains information about the SecureEntry driver that is installed. A sample follows :

```
PRODUCT=Secure Entry
VERSION=3.0
```

```
BUILD=69
BUNDLES=8
DATE=23 Sep 1996 20:46:05
```

This file is located in the SecureEntryPath, INSTALL subdirectory.

SENTRY.CNF This file contains information about how SecureEntry has been configured. It is almost self-explanatory, with the output coming directly from the installation configuration panel. A sample of this file follows :

```
INSTITUTION=Jone's  Bank
INSTALLPATH=C:\SGMSHELL
INSTALLFROM=A:\
CONFIGURATION=1
USERACF=0
USEUCM=0
INSTALLSOURCES=1
```

The only not so obvious field here is the 'CONFIGURATION' one, which can have the following values :

1. Standalone workstation
2. IBM Lan Server networked
3. Network installation. Not Lan Server.

This file is located in the SecureEntryPath, INSTALL subdirectory.

EDYSSLMP.DAT This file describes the configuration of the LMP's (Logon procedures) to be called, as well as the location of the users and security profiles databases. Although its format is internal to SecureEntry, it is an ASCII file and thus, can be seen or edited. You should never touch this file unless otherwise specified by the documentation or SecureEntry development team. This file is located in the SecureEntryPath, NOUSER subdirectory.

In order to read this file, you can use any ASCII editor, since this is an editable file, which must match the following rules :

The file is splitted into two blocks :

The LMPs description list, where all composing logon procedures are described by their unique two-letter specifier :

1. **SS** For the SecureEntry synchronizer LMP. This is mandatory and should always be the first one.
2. **RF** For the RACF validation LMP. This one, if present, has to be the second one, since it is mandatory that this LMP synchronizes passwords to all of the other ones.
3. **UF** For the RACF emulator LMP. This one, if present, has to be the second one, since it is mandatory that this LMP synchronizes passwords to all of the other ones. You can install the RACF emulator instead the RACF validation.
4. **UC** For the UCM channel LMP. This one, if present, has to be immediately after the RF or UF LMP, and its mission is to update the LAN configuration at logon time before the Lan Server (or SecureEntry registry) LMPs do signon, so that they find the requesting user already present in the Lan.

5. **LS** For the Lan Server LMP, in lan server environments. This is the LMP which controls signon to Lan Server.
6. **SR** For the SecureEntry Registry LMP, in standalone environments.
7. **xx** Any other combination represents a user written LMP, which must coded in the DLL named EDYxxLMP.DLL.

The database description specifiers, contains two lines, which describes where the location of the users database is and where that one for the security profiles. The only supported values are :

SR or LS For USERS_DATABASE.

SR For SCPRF_DATABASE.

The optional parameters that can be placed by the side of the LMPs list describe whether the LMP will require a DOMAIN name and which default value to present at signon time.

Follows a sample EDYSSLMP.DAT file :

```
SS 1 SRVMAR
LS 1 SRVMAR
USERS_DATABASE  LS
SCPRF_DATABASE  SR
```

EDYREGDB.VLB This is a ciphered library file which contains the security profiles and/or users and group information. You can not list or modify this file, and it is completely managed by SecureEntry. The CREADB utility can be used to regenerate a new, empty one by administrators. This file is located in the SecureEntryPath, NOUSER subdirectory of all standalone workstations or server machines of networked environments.

EDYKILL.NOT This is an ASCII file which contains a list of processes not to be killed at any time. Since in regular OS/2 installations it is normal for daemons and detached processes to never expect a DosKillProcess call to kill them, it is frequent that many of this kind of processes are not even tested for this event, and may abend or cause unexpected behavior if they are killed. In the other hand, SecureEntry can not afford not to kill all superuser/user context running processes at shutdown/logoff time unless explicitly said so, since any PM program could theoretically resume user interaction at this point, jeopardizing system security.

So, the net idea is that if you find shutdown traps by a specific process, add its name to the list and it will not be killed. This file is located in the SecureEntryPath, NOUSER directory, and accepts wildcards in the process names specification.

A second purpose of this file is to specify 'orderly close' commands for whenever SecureEntry has to kill a given process. This is useful either for performance reasons, or as a last resort when a given application does not end correctly because of interprocess dependencies.

After applying a service to your SecureEntry version the new default file EDYKILL.NOT will be copied into the SecureEntryPath\EXEC directory. If the file SecureEntryPath\NOUSER\EDYKILL.NOT has not been customized for your installation, then it will be automatically updated; otherwise you will have to manually update it in order to make the changes effective.

Read the comments section of the same file for more information on how to alter it.

EDYSTART.CMD

This file is the SecureEntry version of the STARTUP.CMD file and has the same purpose. The reason why it is different is that it must be started by SecureEntry in order to grant a proper secure machine startup. Read the chapter The startup process in order to know more on how to use it.

EdyStart object

This object, located within the SecureEntry NOUSER directory, acts as a folder and can contain a list of shadows which substitute the traditional OS/2 startup folder. The difference being that, by placing your startup objects here, SecureEntry grants that these programs will be started in superuser context by waiting for those before proceeding with logon (assuming the objects represent synchronous tasks).

EDYUCFPW.DSC

This is an ASCII file which contains the list of valid characters and the maximum and minimum allowable length for password validation as needed by the UF RACF SecureEntry emulator. By default this file has maximum length for passwords set to 14 and the minimum one set to 4 characters. The default allowable character set is as follows:

```
A..Z, 0..9, a..z and @, # and $.
```

Remember that if you want to update this file, you will have to update the same definitions at the Host site.

SecureEntry Log files

If at any time you want to check out SecureEntry activity, there are several files which can be used for tracing or auditing purposes :

The installation log file

This file reflects all the installation/service activity due to the machine. It is named 'SENTRY.LOG', and placed in your SecureEntryPath, INSTALL directory. A sample of this file follows :

```
SecureEntry start install : 23  Sep 1996 21:26:56
Applying: Secure Entry Version: 3.0 Build level: 69
Secure Entry start setup : 23  Sep 1996 21:34:03
Relocating base system files..
Creating SENTRY database..
Creating SENTRY  logon configuration file..
Creating SENTRY STARTUP file..
Setting up default profiles
Modifying CONFIG.SYS file..
Modifying SES configuration file..
Registering SYS_DLLs..
SecureEntry start crwrk : 23 Sep 1996 21:34:18
The folder 'SecureEntry ^Workbench' has been created successfully.
The shadow of  'SecureEntry ^Workbench' has been created successfully.
```

```
Saving current launchpad..
SENTRY setup completed correctly..
You must shutdown now to activate changes and start Secure Entry.
Workstation setup completed OK
```

The startup process file

This file has all messages posted to the startup process during its execution. it is named 'EDYLKINI.LOG', and placed in the same directory where the 'EDYSTART.CMD' file must reside, that is, the root directory of the boot drive (unless otherwise specified with the environment variable SGM_INI_LOGPATH). This file is useful in order to know what went wrong in a machine's startup if it was attempted in an unattended environment. See the startup process description for more detailed information.

The administration log file

This file reflects all the activity of the administration API. It is named 'EDYADMIN.LOG', and placed in your SecureEntryPath, NOUSER directory of all the standalone workstations, or of the Network environment SecureEntry servers. You can change this location by setting up a 'SGM_SL_LOGPATH' environment variable within the config.sys file. A sample of this file follows :

```
Date       Time   Computer User    Typ Process      Agent    Rc  Actn Subs ObjType Obj1 Obj2 Keys
-------------------------------------------------------------------------------------------------
23/09/1996 22:50:14         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] ADD  SENT Group  GRUPO
LAN_DATA
23/09/1996 22:50:27         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] ADD  SENT User  LUIS
LAN_DATA
23/09/1996 22:50:28         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] ADD  SENT UserGroup LUIS,GRUPO
23/09/1996 22:51:57         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] UPDT SENT Group  GRUPO
FLOPPY_DISK
23/09/1996 23:01:34         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] ADD  SENT User   ADMIN2
LAN_DATA
23/09/1996 23:01:58         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] UPDT SENT User   ADMIN2
LAN_DATA
23/09/1996 23:02:26         EDYADMIN A EDYSNADM.EXE EDYSEAGT[ ] UPDT SENT User   EDYADMIN
LAUNCHPAD
```

The session event log file

This file reflects all the session activity for a given machine. It is named 'EDYSLA.LOG', and placed in the OS2\SECURITY\SESDB path of the machine's boot partition. You can change this location by setting up a 'SGM_SES_LOGPATH' environment variable within the config.sys file. A sample of this file follows :

```
  Date     Time      Event    Phase  Result   UserId              Additional info
-------- -------- ----------- ------ ------ --------
----------------------------------------
11/21/96 11:43:13 DOMAIN      QUERY   [ ]            Default Domain:   SEDOMAIN
11/21/96 11:44:58 SHUTDOWN    START   [ ]
11/21/96 12:19:57 SHUTDOWN    END     [C]
11/21/96 11:45:05 LOGON       START   [ ]   U0000007 Domain:  SEDOMAIN
11/21/96 11:46:37 LOGON       END     [ ]   U0000007
11/21/96 11:47:33 LOCKUP      START   [ ]   U0000007
11/21/96 11:47:33 LOCKUP      END     [ ]   U0000007
11/21/96 11:47:38 UNLOCK      START   [ ]   U0000007
11/21/96 11:47:46 UNLOCK      END     [E]   U0000007 You typed and invalid Password. Try
again.
11/21/96 11:47:51 UNLOCK      END     [ ]   U0000007
11/21/96 11:48:21 LOGOFF      START   [ ]   U0000007
11/21/96 11:48:30 LOGOFF      END     [C]   U0000007
11/21/96 11:48:43 SHUTDOWN    START   [ ]   U0000007
11/21/96 11:48:49 SHUTDOWN    END     [C]   U0000007
```

The treelock log file

In order to audit the treelock component, it can be configured to log its accesses to a text file. Since this file is treated as another component, you can place it in the NOUSER directory, or assign one in a per user basis. See the description of the Treelock component for a more detailed description of this file.

Public applications log file

This file contains the errors and warnings found by the public applications component during its execution.

The Distribution data log file

This file keeps the information related with the refresh branch and refresh profiles activity. In this file you can find whether the refresh operations (branch or profiles) went ok. It is named 'EDYDIS.LOG' and it is placed in the SecureEntry\NOUSER path of the server machine. You can change the location of this file by setting up the SGM_UCM_LOGPATH environment variable within the config.sys file.

A sample of this file follows:

```
10/28/98 12:16:48 < REFRESH BRANCH ONLINE BEGINS >

> REFRESH STATISTICS
    Buffer Length  : 34738
    Final Register found
>   Read Blocks    : 4
>   Correct Blocks : 4
>   Error Blocks   : 0
>   Number of Add Object Operations    : 1
>   Number of Update Object Operations : 2
>   Number of Delete Object Operations : 1

10/28/98 12:16:57 < REFRESH BRANCH ONLINE ENDED  >
```

Note that you can control the size of the SecureEntry log files, as well as any other ASCII file by using the EDYLOGFS utility.

About processes and running contexts

With SecureEntry installed, each running process in the machine has a context of execution associated for security. This basically means that the system keeps track of who started the process and what privileges did it have. Basically, there are two kinds of execution contexts :

1. **Superuser context** All processes started with an associated superuser context have no file system restrictions (treelock does not check accesses for them), plus are never killed or terminated at user session logoff time.
2. **User context** All processes started with an associated user context will basically have the currently active treelock profile restrictions enforced for file accesses, plus they will be killed/terminated at logoff time unless otherwise specified through the EDYKILL.NOT file.

The obvious following question is what processes run under one or the other context :

Processes started through CONFIG.SYS have superuser context.

Processes started through EDYSTART.CMD have superuser context.

Processes started through the EDYSTART object in the NOUSER folder have superuser context.

Processes started during a user session (between logon and logoff) have User context.

Processes started through a user exit have the context related with the specified user exit :

User exit after startup : Superuser context
User exit before logon dialog : Superuser context
User exit before logon : Superuser context
User exit before LMP signon : Superuser context
User exit after LMP signon : Superuser context
User exit before profiles activation : User context
User exit after logon : User context
User exit after PMSHELL : Undefined context (depends on timing)
User exit before cancellable lockup : User context
User exit before lockup : User context
User exit before unlock dialog : User context
User exit before unlock : User context
User exit after unsuccessful unlock : User context
User exit after unlock : User context
User exit signal : Undefined context (depends on timing)
User exit before cancellable logoff : User context
User exit before imminent logoff : User context
User exit after profiles deactivation : User context
User exit after logoff : Superuser context
User exit before shutdown : SuperUser context
User exit before logoff from unlock : User context
User exit before shutdown from unlock : User context
User exit before LMP signoff : User context
User exit after LMP signoff : User context
User exit open folder : Undefined context (depends on timing)

About Object IDs

When administering SecureEntry, you will find several components that accept only objects with an assigned object ID as suitable for certain operations. This is because the object ID is the only way to uniquely identify an object within the desktop that will retain this characteristic in any machine of a SecureEntry workgroup.

Many objects have already an object ID, so you do not have to worry about this until you find a problem. In the event that you do want to deal with an object for which an object ID is required, but does not have one, the following piece of REXX code can be used to assign one to it :

```
/* */
Call rxfuncadd sysloadfuncs, rexxutil, sysloadfuncs
Call sysloadfuncs

/* assign to objectpath the fullpath of the object */

result=SysSetObjectData(objectpath,  'OBJECTID=<'||NewId||'>')
```

Note that object IDs have to be unique, and you will have to repeat this command in all machines where this object will be used by SecureEntry.

Refer to Public Applications object IDs to get more information about this kind of LAN Server objects.

SecureEntry session process description

Up to now, we have been describing the SecureEntry process as separate components, but in order to have a clear view on what happens 'under the covers', in terms of events timing, we will describe now all the processes.

After power on, the sequence of events is the following :

1. Machine power on
2. OS/2 bootstrap and config.sys processing
3. SecureEntry device drivers load and SES activation
4. Second read of config.sys by OS/2, honoring 'call=' statements
5. First shell (PM) is loaded. SecureEntry PM DLL hooks loaded
6. Asynchronously to this point, if configured, OS/2 launches the daemons which will end up loading the second shell (Workplace). Whenever the workplace will be loaded and in a stable state, SecureEntry will serve the user exit after pmshell. Note that this happens normally not before several of the following steps are completed
7. SecureEntry processes the EDYSTART.CMD file (Startup process)
8. SecureEntry serves the user exit after startup
9. SecureEntry forces a SES logon event
10. Control is passed to SecureEntry from SES in order to process the logon event
11. **(LOGON)** SES Logon event processing : SecureEntry serves the user exit before logon dialog. If the user exit decides to do a guest logon, then go to **GUEST**
12. SecureEntry inspects the logon parms area and decides whether to present the logon panel or skip the next point
13. The logon panel is displayed. When the user presses 'OK', go to next step, or if shutdown has been pressed, then go to **SHUTDOWN**

14. SecureEntry serves the user exit before logon. If the user exit decides to do a guest logon, then go to **GUEST**
15. SecureEntry serves the user exit before logon changing password if a password change request has been made for this logon
16. SecureEntry issues signon to the LMP (Logon procedures) chain in the order defined in the EDYSSLMP file, synchronizing passwords if necessary. If there is any error, then go to **LOGON**
17. SecureEntry downloads the security profiles for the signed on user into the SecureEntryPath\WORK directory, overriding if necessary the group profiles with the per user profiles. If there is any error, then go to **LOGON**
18. SecureEntry activates each component for which a profile was downloaded with the profile data, following the interface as defined in the repository database. If there is any error, then go to **LOGON**
19. **(GUEST)** SecureEntry serves the user exit after logon
20. SecureEntry serves the user exit after logon changing password if a password change request has been made for this logon
21. The user session begins. Only a request for logoff or shutdown will exit from this state, and proceed to the next step. Note that any new process started within the user session will be considered by SecureEntry a user privileged process, and killed at logoff time
22. A request for logoff or shutdown has been made. The appropriate event (logoff or shutdown) is posted to SES.
23. SecureEntry serves the user exit before cancellable logoff. If cancel has been returned we go back to the user session (previous step). Otherwise SecureEntry displays if configured to do so, the confirmation dialog as a last opportunity to keep on the user session.
24. SecureEntry serves the user exit before imminent logoff
25. SecureEntry kills all of the remaining user session processes
26. SecureEntry Deactivates all of the components downloaded in the logon process by passing them a call indicating them to reset to the profiles as stored in the NOUSER directory (machine configuration profiles)
27. SecureEntry Uploads any component profile which may have been changed by the user to the repository, and erases them from the workstation's WORK directory
28. SecureEntry issues sign-off to the LMP chain, calling only those LMPs which were in a signed on state
29. SecureEntry serves the user exit after logoff
30. If the event in process was logoff, then post a SES logon event which will cause a jump to **LOGON**, and terminate the logoff event.
31. **(SHUTDOWN)** SecureEntry serves the user exit before shutdown
32. SecureEntry kills foreground processes which are still alive
33. SecureEntry gives control to SES to shutdown the system, which will in turn call OS/2 shutdown API

Fine tuning SecureEntry. Performance considerations

There are several factors affecting SecureEntry operation performance. Follow this chapter for a clue on how to tune your system.

Note that performance tuning usually yields the best results if you look first after the 'bottlenecks' of the system. Symptoms of such bottlenecks are :

General system sluggish response time, and possibly too much disk activity. This is normally an indication of low memory condition, and is accompanied by very big swapper files (when compared to the amount of memory installed). It deserves a general system tuning session and may possibly end up by adding more memory to the system.

General operating system performance tuning

Specific performance problems at certain moments. i.e, at logon time. This one is normally an indication of another sort of problem, and if happening within SecureEntry, may be cured by following the indications under 'SecureEntry specific performance tuning'.

SecureEntry specific performance tuning

General operating system performance tuning

The following things can be tuned in order to improve performance on a OS/2 running machine :

Software tuning

1. Swapper : The swapper file should be positioned into the more used partition of the least used physical drive. Also, it usually helps to preallocate the swap file to a given size so that it does not have to be dynamically shrunken or expanded. You can do this by using the SWAPPATH sentence of your CONFIG.SYS file, as follows :
```
SWAPPATH=D:\ 8192 16384
Set the swapper file to reside in the root directory of the D drive,
with an initial size of 16 Mb, posting a user alert if the remaining
free space is less than 8 Mb
```

2. Disk cache : Usually a big enough disk cache helps performance also. Use the DISKCACHE sentence for FAT drives, and/or the CACHE / IFS sentences for HPFS drives. Do not forget the lazy write flags for maximum performance!!.
3. Remove/do not install unused software/drivers. I.e, multimedia extensions, or HPFS drivers if this file system is not being used.
4. Remember to switch off traces and / or verify switches in end user runtime environments. Also gadgets such as 'desktop animation' can make the system feel slower.
5. Try to keep your disk defragmented, and if possible use HPFS instead of FAT partitions, specially for big disks.
6. Fine tune your PATH, LIBPATH and DPATH config.sys statements, placing first those directories which are accessed more.

7. In fast machines and/or server machines, it may be a good idea to reduce the time slicing parameters : **TIMESLICE** and **MAXWAIT**, to improve the responsiveness of the system. (i.e TIMESLICE=32,512 and MAXWAIT=1).
8. If using windows applications, it may be a good idea to switch on the WINOS2 fastload setting (through the proper page of the settings notebook of the WINOS2 configuration object), and run all windows applications from the same session.
9. For startup performance tuning, you can do the following :

   Keep optimized system .INI files (OS2.INI and OS2SYS.INI). There are several utilities around that can help you with this. SecureEntry provides you also with one : EDYCLINI.

   Remember to turn off archiving of your critical files at each power on. You should do this only when major updates to your system configuration have been made. You can do this from within the desktop page of the desktop settings notebook.

Hardware tuning

1. Adding memory is many times the most efficient way to improve performance. Look at the server machines first.
2. Consider also adding L2 cache memory to your system if this is an option for your hardware.
3. Of course, faster processor/disks help a lot also.
4. Think about improving your host communications link speeds, specially if using UCM.
5. For Lan based installations, keep an eye on Lan performance and contention problems, which may be solved by lan splitting or hardware changes.

SecureEntry specific performance tuning

Although SecureEntry is installed with most options already tuned for performance, the following rules could also be observed to help improving the response time even more :

Although SecureEntry does not have a minimum physical memory requirement, you are encouraged to consider that the approx working set size of the code is of around 3Mb, being this the amount of physical memory that you should add to the system if your objective is to achieve the same level of performance that you have without it in the same hardware. Note also that SecureEntry has constantly a minimum of 15 threads active, so to be safe you should increase your CONFIG.SYS THREADS statement by that amount.

Within Lan Server environments, take into account that the 'big chunk' of signon time is taken by Lan Server itself, so generally tuning Lan Server performance will result in a substantial SecureEntry operation improvement.

Also for Lan environments, take a close look at the protocol parameters of your PROTOCOL.INI file, located in the IBMCOM directory. The base network software normally installs itself with too much conservative default parameters for simple environments such as the typical banking branch network. This is why simply by changing the NETBIOSTIMEOUT parameter from 2000 to 500 msec you can easily cut the signon time by about 20 seconds or more. Another parameter to reduce for fast not bridged lans would be the NETBIOSRETRIES one on the same file. Reducing it to 2 or 3 will improve response time also.

If you are having problems at signon time, measure the complete logon time with a stopwatch (not the first logon), and verify if the time is mostly spent :

1. When the logon panel is displayed but the window 'signon in progress.. please wait' has not yet appeared. In this case, look for your user exit code or at other activity being performed in the machine at the same time.
2. When the logon panel is displayed and the window 'signon in progress.. please wait' is over it. In this case, the time is spent on RACF validation, UCM download and Lan Server validation and signon. These are the components to be checked.
3. When the logon panel has disappeared. Check in this case after the individual components activation time.

Avoid using too many user exit code when possible. User exits are a very powerful mechanism, but quite expensive. An exception to this rule can be found when it is necessary to unload software. It will always be more efficient to do it using the proper command than leaving the task to SecureEntry, which does not know the internals of the processes composing any particular software package. You can use for this purpose the user exit before shutdown and/or the user exit before imminent logoff, or a customized *EDYKILL.NOT*file.

Try to use machine-assigned components (in the NOUSER directory) rather than group or user assigned ones, since this will save time, not having to activate/deactivate the same profiles at each signon.

Consider the possibility of enabling the environment variable 'SGM_WPS_FASTLOAD' in your config.sys. Take into account the risks exposed under Environment variables. Note that this setting only modifies the 'perception' of faster signons, since the population of the desktop will be anticipated a few seconds.

Do not change the config.sys variable RESTARTUSERSHELL default setting of 'NO', except when required by your installation. Restarting the Workplace Shell at each signon will easily delay it by several seconds.

Place the SecureEntry path as closer to the beginning of the PATH and LIBPATH statements as possible within config.sys.

Also for best power on performance, leave alone the 'SGM_WPS_SKIP_PREPOPULATE' environment variable set to 'ALL', as installed by SecureEntry.

Suppress all unneeded logging/traces. For instance, you can set the trace level to 0 by adding to your EDYSTART.CMD file :

```
EDYTRSET /A:0
```

For the launchpad component, two considerations :

1. Try to set up launchpads which contain only objects with object ID. Adding objects to a launchpad (when it is created), is much more time expensive if the objects to be added do not have an object ID.
2. If installing within an old machine (i.e, OS/2 has been installed there for a long time), it may be a good idea to 'clean' the nowhere directory with a tool like the supplied EDYCLINI. Old launchpad shadows (not SecureEntry ones) end up here and are never deleted by the system itself, making access to objects residing in this location slower.

WARNING : Do this only if you find lots of objects within the NOWHERE folder.

For your screensaver or background bitmaps, take into account that big bitmaps take a lot of disk space, which in turn takes a long time to be read, and may cause quite some swapping activity to be processed, so it is a matter of finding the desired middlepoint between image quality and response time. If you decide to use 'big' bitmaps anyways, then if possible try to use bitmaps with the same sizing in pixels/colors of your desktop resolution, as this will avoid expensive color remapping and reseizing functions.

Error codes and messages

Session control errors and messages

The following errors may appear during regular runtime and session control functions, i.e logon, logoff, lockup... Some of these can be reported also through the administration APIs.

Message number	Message	Description
EDY0001E	Invalid password.	The entered password is not valid for this user during a logon or unlock operation.
EDY0002E	Operating System error.	A generic operating system error happened. The message can contain additional data. Check traces or log files to know more about it.
EDY0003E	Subsystem %s not available.	A required subsystem was not available at logon time. (SR: Registry, LS: Lan Server, UC: UCM, RF: RACF..).
EDY0004E	Control file wrong configuration.	Check the integrity of the *EDYSSLMP.DAT* file. It probably has been altered.
EDY0005E	Logoff state: no user logged on this workstation.	An operation pre requiring a logon found that no user has logged into the machine (typically can happen during logoff if a lan server logoff was issued during user session).
EDY0006E	Unknown user.	The requested userid is not known to the logon subsystem.
EDY0007E	Another user is logged on this workstation.	The logon subsystem found a user already logged on during logon processing.
EDY0008E	Password expired.	The password has expired. You must change the password to proceed.
EDY0009E	New password invalid.	The new password is not valid. Either too short, too long, contains invalid characters, or is equal to one of the last used passwords.
EDY0010E	Bad input parameter.	This is an internal error. Contact service support.
EDY0011E	Connection denied, user cannot log on.	The user's account has been revoked. Contact your system administrator.

EDY0012E	Impossible to load DLL %s.	During logon subsystem initialization, the specified DLL could not be found.
EDY0080E	Security Profile manipulation general error.	Internal error. Contact service support.
EDY0081E	Error while getting access to the shared memory.	Check machine resources and/or contact service support.
EDY0082E	Shared memory is full.	Internal error. Contact service support.
EDY0083E	Information not available in the shared memory.	Internal error. Contact service support.
EDY0084E	Querying for shared memory information: input buffer is too small.	Internal error. Contact service support.
EDY0085E	Error reading INI file.	Internal error. Contact service support.
EDY0086E	Reading INI: label not found.	Internal error. Contact service support.
EDY0087E	File not found.	The requested file does not exist.
EDY0088E	File copy error.	The requested file could not be copied.
EDY0100W	Emergency logon (basic subsystems are available but %s has failed). Password change not allowed.	The basic logon subsystems are available, but the director one failed (typically RACF). You are logged on in emergency mode. Password change will not be allowed for this reason.
EDY0101W	Control File: found end of file.	Check the integrity of the *EDYSSLMP.DAT* file. It probably has been altered.
EDY0102W	Control File: found an empty line.	Check the integrity of the *EDYSSLMP.DAT* file. It probably has been altered.
EDY0103W	Pending request.	Internal error. Contact service support.
EDY0104W	Control File: found the users DB line.	Check the integrity of the *EDYSSLMP.DAT* file. It probably has been altered.
EDY0105W	Control File: found the profiles DB line.	Check the integrity of the *EDYSSLMP.DAT* file. It probably has been altered.
EDY0110E	Lan Server fail.	Lan Server was available but failed. Possible causes are Lan Server internal error, Too many names in access control file, error accessing the NET.ACC file, and others.
EDY0111E	Lan Server configuration error.	An error related to Lan Server configuration or code level has been found. Either net path not found, not supported request or API, computername not valid, or wrongly configured server.
EDY0112E	Some Lan Server service has not been started and was required.	This error should not appear in any case. EDY0003 would be expected.
EDY0113E	Lan Server domain fail.	Could not obtain the domain server name. Contact administrator.
EDY0114E	This operation needs Lan Server administrator privilege.	Unauthorized request reported by Lan Server.

EDY0115E	UserId and/or password are invalid and cannot be validated. Try again.	The EDYSRV program is probably not running in the domain controller machine. Thus, could not refine further the error code and this more generic one is displayed.
EDY0116E	Lan Server: This user is already logged in the LAN.	Your Lan Server is not configured for multilogon, and you are attempting to log on while are logged on already in another machine.
EDY0117E	Lan Server: Logon/logoff in process, or Lan Requester tool is active.	Another logon or logoff is in progress at this time.
EDY0118E	Lan Server: User account expired.	The user account has expired. Contact your system administrator.
EDY0119E	Lan Server: new password is too short.	Self explanatory.
EDY0120E	Lan Server: new password is too recent.	Self explanatory.
EDY0121E	Lan Server: The password cannot be changed.	Self explanatory.
EDY0122E	Bad connection conditions: invalid logon hours.	Self explanatory.
EDY0123W	Domain controller unavailable. Using backup controller.	Informational : The main domain controller was not found in the domain, A backup controller has taken its role.
EDY0201E:	Too many subsystem logons.	Internal error. Contact service support.
EDY0202E:	Invalid system id %s.	Internal error. Contact service support.
EDY0203E:	Unauthorized request. Access denied.	Unauthorized request reported by SecureEntry Registry.
EDY0204E:	A user %s already exists.	Could not add user/group because it is already defined as a user in the SecureEntry registry.
EDY0205E:	A group %s already exists.	Could not add user/group because it is already defined as a group in the SecureEntry registry.
EDY0206E:	A component %s already exists.	Could not add a component profile because it already exists.
EDY0207E:	The group %s does not exist.	The requested group does not exist in the registry.
EDY0208E:	The group %s is not empty.	Can not delete a group from the registry while users belong to it.
EDY0209E:	Can not delete current logged user.	The logged on user can not be deleted.
EDY0210E:	The component name %s is invalid.	The component has not been registered as a valid component to the registry, or the syntax is invalid for the name.
EDY0211E:	The component %s does not exist.	Could not update/delete the component profile because it does not exist related to the specified user / group.
EDY0212E:	Could not load DLL %s.	The registry could not load the required DLL.

		Make sure it exists and is accessible through the libpath config.sys setting.
EDY0213E:	Could not find address of %s.	Internal error. Contact service support.
EDY0214E:	Users Database not EdyReg defined.	The requested operation is only valid for standalone configurations.
EDY0215E:	Profiles Database not EdyReg defined.	Check EDYSSLMP.DAT file syntax and integrity.
EDY0216E:	EDYAPI Error %s.	The registry got an error while querying EDYAPI for configuration information. Check the reported error in this same table.
EDY0217E:	Sentry subsystem not found.	There was probably an error during installation. Contact service support.
EDY0218E:	Component list empty.	Internal error. Contact service support.
EDY0219E:	Syntax error in component list : %s .	The registered components list is in error. Check SENTRY.DSC file and reregister it through UPDATEDB.
EDY0220E:	The component %s is unknown.	The component has not been registered as a valid component to the registry.
EDY0221E:	The mandatory component %s is not present.	Mandatory component missing. The user can not logon.
EDY0222E:	Unknown EdyReg error %d.	An unknown error was reported to the registry. Check the error message.
EDY0223E:	Unknown EdyReg error class %d.	Internal error. Contact service support.
EDY0224W:	Component to be put matches previous version. Not stored again.	Informative.
EDY0330E:	Internal error.	Internal error. Contact service support.
EDY0331E:	A security error due to integrity violation has occurred. The system will shutdown.	Internal error. Contact service support.
EDY0332E:	Incorrect Userid length. Try again.	Error reported during logon/unlock panels syntax checking.
EDY0333E:	Incorrect Password length. Try again.	Error reported during logon/unlock panels syntax checking.
EDY0334E:	Incorrect Domain length. Try again.	Error reported during logon/unlock panels syntax checking.
EDY0335E:	New and verification Passwords don't match.	Error reported during logon/unlock panels syntax checking.
EDY0337E:	You typed and invalid Password. Try again.	Error reported during logon/unlock panels syntax checking.
EDY0338E:	Invalid Password change. Contact your System Administrator.	Error encountered during the syntax validation phase of the logon and/or unlock events.
EDY0341E:	You are not authorized to logon. Contact your System Administrator.	Error encountered during the syntax validation phase of the logon and/or unlock events.

EDY0347E:	A Severe Initialization Error has been produced. Module: %s. Function: %s. Line: %d. System Error: %#X. The system will shutdown.	Take note of the provided information and contact SecureEntry support.
EDY0349E:	The OS/2 initialization file contains an invalid application. The program to fix it could not be executed. The system will shutdown. Contact your System Administrator.	The initialization file specified by the environment variable USER_INI, normally being the file OS2.INI located in the subdirectory OS2 of the boot partition, contains the pair *SYS_DLLS*/*Load* having an invalid key value. This error may be caused by overwriting the system initialization files with other ones not having the needed information. The SecureEntry program that solves this program could not be launched. Contact service support.
EDY0350E:	The OS/2 initialization file contains an invalid application. The program to fix it terminated unexpectedly. The system will shutdown. Contact your System Administrator.	The initialization file specified by the environment variable USER_INI, normally being the file OS2.INI located in the subdirectory OS2 of the boot partition, contains the pair *SYS_DLLS*/*Load* having an invalid key value. This error may be caused by overwriting the system initialization files with other ones not having the needed information. The SecureEntry program that solves this program abnormally ended (ABENDed). Contact service support.
EDY0351E:	The OS/2 initialization file contains an invalid application. The program to fix it could not actually fix it. The system will shutdown. Contact your System Administrator.	The initialization file specified by the environment variable USER_INI, normally being the file OS2.INI located in the subdirectory OS2 of the boot partition, contains the pair *SYS_DLLS*/*Load* having an invalid key value. This error may be caused by overwriting the system initialization files with other ones not having the needed information. The SecureEntry program that solves this program could not correct the problem. Contact service support.
EDY0352E:	The OS/2 initialization file contains an invalid application. The problem has been fixed. The system will shutdown. Contact your System Administrator if you continue having this problem.	The initialization file specified by the environment variable USER_INI, normally being the file OS2.INI located in the subdirectory OS2 of the boot partition, contains the pair *SYS_DLLS*/*Load* having an invalid key value. This error may be caused by overwriting the system initialization files with other ones not having the needed information. SecureEntry fixed the problem in the initialization file. An IPL is needed in order for the new valid values to be effective. Another forthcoming of this error after IPLing the system indicates an automatic replacement of operating system initialization files at some stage of the boot process.

Administration API errors

Although you will normally encounter these errors when working directly with the administration API, it is possible also to get one of these when using any of the administration tools. The key is to look at the error type and then find the appropriate error list depending on the ultimate component causing the error condition, as follows :

Error types table

Error code	Error type	Description
xxxx	RXUC	The REXX layer DLL found an error. Look into the associated message to find more about it.
xxxx	SLAG	SecureEntry selector agent error. This means that the SecureEntry selector agent, that is, the common entry point to the administration API, found an error. Refer to the agents error table to find more about xxxx.
xxxx	SEAG	SecureEntry administration agent error. This means that the SecureEntry subsystem administration agent found an error. Refer to the agents error table to find more about xxxx.
xxxx	SESR	SecureEntry registry component error. The SecureEntry/UCM subsystem administration agents received an error reported from the SecureEntry registry. Look into the associated message to find more about it. Refer to the Session control errors and messages table to find specific information about code xxxx.
xxxx	LSAP	Lan Server agent error xxxx. This means that the SecureEntry Lan Server agent found an error. Refer to the agents error table to find more about xxxx.
xxxx	LSER	Lan Server API error xxxx. This means that the SecureEntry Lan Server agent got an error passed up from the Lan Server API. Refer to the Lan server documentation to find about error code xxxx. If this is a common error, chances are that the associated error message contains some meaningful information. TIP: Most Lan Server API errors are also OS/2 base errors, so HELP SYSxxxx may give you a clue.
xxxx	REGT	Registry API error xxxx. This means that the SecureEntry Lan Server agent got an error passed up from the SecureEntry registry component. Look into the associated message to find more about it. Refer to the Session control errors and messages table to find specific information about code xxxx. TIP: The Lan Server agent does only use the SecureEntry Registry to store per group logon assignment data, so the error must be related to an operation of this kind.
xxxx	UCAG	UCM administration agent error. This means that the UCM subsystem administration agent found an error. Refer to the agents error table to find more about xxxx.
xxxx	UCSQ	A SQL API error has been reported to the UCM subsystem agent. Refer to the SQL error code table to find about error xxxx.
xxxx	UCAP	A UCM API error has been reported to the UCM subsystem agent. Refer to the agents error table to find more about error code xxxx.
xxxx	UCME	This is a memory error reported by the UCM API to the UCM administration

		agent. An allocation request failed either in the host or workstation site.

Agents error table

Error number	Error name	Description
001	INFOOK	Informational message. No real error happened, the associated message contains event information that had to be reported.
002	INVALID_KEYWORD_SIZE	One of the passed keywords (object attributes) was either too short or too long.
003	INVALID_OBJTYPE	The agent received an invalid object type through the API.
004	INVALID_OPERATION	The operation is not valid at this time (add, update, delete..).
005	INVALID_PARAMETERS	Generic error to report invalid parameters passed to the agent through an API call.
006	EXISTS_OBJ	The object already exists.
007	NEXISTS_OBJ	The required object does NOT exist.
008	EXPECTED_KWD	A mandatory keyword was expected for this object.
009	NEXPECTED_KWD	An unexpected keyword was received for this object.
010	NEXIST_KWD	The keyword does not exist.
011	EXIST_KWD	The keyword already exists.
012	INCORRECT_KWD	Generic. The keyword is incorrect.
013	NOT_ALL_KWD	Not all keywords could be returned.
014	MANY_ROWS_FETCHED	Too many rows fetched. The UCM API is configured with a maximum of tuples to handle as a response to a query. Use the EDY_UCM_MAXROWS to customize this maximum for your needs.
015	NROWS_FETCHED	No rows fetched. The result is empty since there are no rows matching the search criteria.
016	DEL_USR_UCM	Could not delete user from the UCM subsystem. It is still defined in other subsystems.
017	DEL_SUB_UCM	Could not delete the UCM subsystem. It must be the last one to delete.
018	NEXIST_DEFKWD	Default required keyword 'Keyword Id' not found for the UCM subsystem.
960	PRVERR	Not enough privilege for the requested operation. Check the environment variable SGM_ADM_PRIV setting.
961	LOGERR	Could not open the log file for writing. The returned operating system return code is also supplied.
962	DYNERR	Could not dynamic link the required module. The returned operating system ret. code is also supplied.
963	MAXERR	The maximum number of subsystems has already been

		reached. Contact SecureEntry service support if this error happens.

Processes Auditor Editor Errors and Messages

The following messages may appear during regular editor runtime.

Error Number	Error Message	Error Description
EDY1000	Initialization failed.	An initialization task could not be carried out. This may be produced as a consequence of lack of resources in the system or of the editor corruption. Verify that other programs in the system are running correctly and that the image of the editor executable in the disk is not corrupted.
EDY1001	Failed to create main window.	The editor main dialog could not be created. This may be produced as a consequence of lack of resources in the system or of the editor corruption. Verify that other programs in the system are running correctly and that the image of the editor executable in the disk is not corrupted.
EDY1002	Cannot open input file.	The specified file is R/O, or is held by another process, or does not exist, or the full path name does not exist, or is actually a directory, or the disk is full, or the system has ran out of handles. In the latter, wait until another program has ended and retry the operation.
EDY1003	Cannot open output file.	The specified file is R/O, or is held by another process, or the full path name does not exist, or is actually a directory, or the disk is full, or the system has ran out of handles. In the latter, wait until another program has ended and retry the operation.
EDY1004	Failed to load help manager.	The standard OS/2 help manager could not locate the file EDYEXECE.HLP. Verify that this file resides in the SecureEntry installation path, HELP subdirectory.
EDY1005	Failed to load string.	The editor could not locate a string. Verify that the file EDYERROR.MSG resides in the SecureEntry installation path, EXEC subdirectory, and that the image of the editor executable in the disk is not corrupted.
EDY1006	Failed to display help panel.	The editor could not locate a help panel. Contact SecureEntry support team.
EDY1007	Cannot load Exit List processor.	The termination routine of the editor could not be loaded. This may be produced as a consequence of lack of resources in the system. Verify that other programs in the system are running correctly.
EDY1008	Error getting file information.	A call to the spooler validation or file system seek routines resulted in error. Contact SecureEntry support team.
EDY1009	Not enough memory.	Lack of memory in the system. Reduce the number of running programs, or the values of the BUFFERS=, TRACEBUF=, DISKCACHE=, THREADS=, RMSIZE=, or DEVICE=VDISK.SYS statements in the CONFIG.SYS file and then restart the system, or remove unwanted files from the swap file disk and restart the system.

EDY1010	Error reading file.	A call to the spooler validation or file system read routines resulted in error. Contact SecureEntry support team.
EDY1011	Error writing file.	A call to the spooler validation or file system write routines resulted in error, or the disk is full. If not the latter, contact SecureEntry support team. For an export operation, this error indicates that not all the information has been correctly exported, if any.
EDY1012	Invalid input file format.	The input file is not a SecureEntry Processes Auditor profile. Retry the operation with a real SecureEntry Processes Auditor profile.
EDY1013	The process name to audit is missing.	You did not provide the process name to audit when trying to add a new process name to the list of processes to audit. Provide a process name and retry the operation.
EDY1014	The process name to audit is invalid.	The process name to audit you tried to add contains one or some of the following invalid characters: 01-1F hex " + , / ; < = > [ ] | Remove the invalid characters and retry the operation.
EDY1015	The process name to audit already exists.	You tried to add a process name that already exists. Provide a different process name and retry the operation.
EDY1016	The number of invocations is missing.	You specified to control the number of invocations of the selected process, but you did not specify the upper limit. Specify it and retry the operation.
EDY1017	The number of invocations must be a number.	You specified to control the number of invocations of the selected process, but the upper limit you specified is not a number. Specify it as a number and retry the operation.
EDY1018	The number of invocations must be in the range 1 to 65535.	You specified to control the number of invocations of the selected process, but the upper limit you specified is out of range. Specify it to be in the range 1 to 65535 and retry the operation.
EDY1019	Could not test the file. Returned error code: %u	The activation or deactivation of a SecureEntry Processes Auditor profile did not succeed. Note down the enclosed return code and contact SecureEntry support team.

The following messages may appear during import operations. Notice that any error will abort the whole import operation.

Error Number	Error Message	Error Description
EDY1040	Unexpected double quote found at line %u of input file. Import process aborted.	Double quotes must enclose only process names containing blank spaces. Remove such a symbol from the specified line in the import file and retry the operation.
EDY1041	Expected double quote not found in line %u of input file. Import process aborted.	Double quotes must enclose only process names containing blank spaces. Add such a symbol to the specified line in the import file and retry the operation.
EDY1042	Unexpected end of line in line %u of input file. Import process aborted.	The specified line in the import file was not complete. Either remove or comment the line, or add the missing information, and retry the operation.

EDY1043	The number of invocations in line %u of input file must be a number in the range 1 to 65535 or the word 'All'. Import process aborted.	Change the number of invocations to audit in the specified line of the import file to a number in the range 1 to 65535 or to the word 'All' (without single quotes). Then retry the operation.
EDY1044	Unexpected value for Invocations Are Working Sessions in line %u of input file. Import process aborted.	Change the value for Invocations Are Working Sessions in the specified line of the import file to 'Yes' or 'No' (without single quotes). Then retry the operation.
EDY1045	Unexpected value for Discard CPU Usages having Value Zero in line %u of input file. Import process aborted.	Change the value for Discard CPU Usages having Value Zero in the specified line of the import file to 'Yes' or 'No' (without single quotes). Then retry the operation.
EDY1046	The start date in line %u of input file is not valid. Import process aborted.	Change the format of the string conforming the start date in the specified line to a valid one and retry the operation.
EDY1047	The start time in line %u of input file is not valid. Import process aborted.	Change the format of the string conforming the start time in the specified line to a valid one and retry the operation.
EDY1048	The ending date in line %u of input file is not valid. Import process aborted.	Change the format of the string conforming the ending date in the specified line to a valid one and retry the operation.
EDY1049	The ending time in line %u of input file is not valid. Import process aborted.	Change the format of the string conforming the ending time in the specified line to a valid one and retry the operation.
EDY1050	The ending timestamp in line %u of input file is less recent than or equal to its related starting timestamp. Import process aborted.	Either change the ending timestamp to a more recent one, or the starting timestamp to a less recent one, and retry the operation.
EDY1051	The start date and time and the process ID at line %u of input file has already been processed in the file. Import process aborted.	You are trying to import a detail line whose key, constructed by the start date and time and the process ID, already existed in the same import file. Slightly modify the key in the specified line in the import file by changing the hundredths of second and retry the operation, or comment the specified line and retry the operation.
EDY1052	The process ID specified in line %u of input file is not valid. Import process aborted.	The process ID could not be understood as an hexadecimal value. Change it in the specified line to a valid one and retry the operation.
EDY1053	Not enough memory for import process starting at line %u of input file. Import process aborted.	Lack of memory in the system. Reduce the number of running programs, or the values of the BUFFERS=, TRACEBUF=, DISKCACHE=, THREADS=, RMSIZE=, or DEVICE=VDISK.SYS statements in the CONFIG.SYS file and then restart the system, or remove unwanted files from the swap file disk and restart the system.

EDY1054	Too many items starting at line %u of input file. Import process aborted.	The maximum number of items in the list of processes to audit is 32767. Either try to break the input file into smaller ones, or try to split the current profile into profiles having less items.
EDY1055	The start date and time and the process ID at line %u of input file already exist in current profile. Import process aborted.	You are trying to import a detail line whose key, constructed by the start date and time and the process ID, already exists in the current profile. Slightly modify the key in the specified line in the import file by changing the hundredths of second and retry the operation, or comment the specified line and retry the operation.

UCM specifics

Working with UCM allows SecureEntry to provide you with centralized, corporate level administration. In order to set up the centralized management workstation, read the Setting up the User Centralized Management workstation section.

Under this environment, all of what has been explained until now is absolutely valid, with very few differences :

You have to use the administration tools through the UCMADM command, which will setup the system to reroute all administration requests to the host system. So, for instance, to call the interactive administration tool the command becomes :

```
UCMADM EDYSNADM
```

If you have installed the RACF validation feature, you will not be able to change user passwords through the SecureEntry administration tools. You will have to use RACF to do so.

If you have installed the RACF emulator, you will be able to change user passwords through the SecureEntry administration tools.

All changes you make to the central repository will become effective in the LAN at logon time for the affected users. Changes made to group or resource definitions will have to be processed through the **EDYUCDIS** program before they are effective in the LAN environment.

You can define that all changes made to groups and resource definitions in the central repository become effective in the LAN environment through an asynchronous process (Refresh branch online). This is an alternative method to **EDYUCDIS** program. The utility **EDYBRNVW.EXE**, can be used in that case at any workstation to view the branch synchronization data. To activate this procedure you must use the UCM administration tools. With this tools you can define the update method for all branches of the corporation.

You can allow parallel administration (allow regular LAN administration), but then the central repository will not know about changes you made at a LAN level, so you are encouraged not to do so.

Refresh branch online

Refresh Branch Online is a procedure to update the definitions of groups and resources of every branch of the corporation. This procedure has been integrated in EDYSRV.EXE process for updating LAN definitions dynamically.

The refresh process is an alternate way to the Host batch processes, which gets information of UCM database changes (tables of changes), and to the Lan batch process (EDYUCDIS), which updates this information in the SecureEntry repository server.

To activate the refresh branch online process, the EDYSRV process must be loaded with a specific parameter (policy). This parameter is explained in the **EDYSRV and EDYFREE** at SecureEntry maintenance utilities.

The EDYDIS.LOG file keeps information about all refresh branch update activity. This file is stored in the path pointed to by the SGM_UCM_LOGPATH environment variable, explained in **UCM specific** at Environment Variables.

SecureEntry provides the way to allow for easily update of the definitions of groups and resources. This chapter describes how do you do this. The following pages give more information about the process.

Update policy methods

Purge change tables

Update policy override

Update policy methods

The different methods to update this definitions are named update policy.

When this procedure is started, the actual update level of every branch is established by checking for it in the corporate repository. Then the necessary changes are downloaded to the branch and applied, registering the branch as updated to the latest known level.

The update policy can be modified according to needs of every corporation, and stored at the UCM repository. You can do this operation through to UCM administration tools. At installation time, this procedure will be off by default.

Once a day, every branch will query the Host for the update policy to synchronize the Lan policy for if it has been changed.

The update policy can take the following values:

Logon The branch information changes, if any, are downloaded from the Host at LOGON time.

Ipl The branch information changes, if any, are downloaded from the Host at IPL time.

Never The branch information changes are never downloaded from the Host.

Time The branch information changes are downloaded from the Host at the hour and minute specified.

Purge change tables

When the system administrator defines the update policy branch through the administration tools, he also can define the number of branches of the corporation to be stored in the corporate repository. If this number is not zero, then the purge process will be activated. Then, when changes are downloaded to all branches of the company, the information will be removed from the tables of pending changes within the corporate repository.

Update policy override

You can customize EDYSRV to override the update policy stored in the repository of the company. To do this, you must load EDYSRV with one specific parameter which will override de update host update policy.

This parameter can take one of the following values at load time:

Logon The branch information changes, if any, are downloaded from the Host at LOGON time.

Ipl The branch information changes, if any, are downloaded from the Host at IPL time.

Never The branch information changes are never downloaded from the Host.

Time The branch information changes are downloaded from the Host at the hour and minute specified.

Periodically Specify the wait number of minutes before activating the refresh process and then repeat it every specified number of minutes. At start up time, the branch information changes, if any, are downloaded from the Host.

UCM Logging Facility

Through of the UCM administrator utilities you can activate de Logging facility for UCM operations. All the operations will be logged at Host site and later can be managed by the EDYEXLOG process at Host environment. For further information about this process refer to the EDYEXLOG batch process in the UCM administrator's guide.

UCM Recovery Facility

If the Refresh branch online process fails because a wrong definition has been made, you can correct the definition through the UCM administrator utility and afterwards run the EDYRVUCM process in your UCM administrator workstation to correct and compress the change tables involved in the error.

This tool checks the change tables of the UCM database compressing and correcting the unnecessary rows as described in EDYRVUCM recovery tool.

RACF emulator

The RACF emulator lets you validate users passwords against the UCM database instead of the security subsystem that you may have installed at Host site.

You can specify the valid character set and maximum and minimum length of the passwords to be used for the users of the corporation.

Deinstalling SecureEntry

If, after installing, you ever want to deinstall SecureEntry 3.0, you can use the command 'UNINSTAL.CMD' that will be placed under your x:\SGMSHELL\INSTALL directory. It will completely remove SecureEntry from your machine. The command syntax is as follows:

```
UNINSTAL [BATCH] [SHUTDOWN]
```

Use the *BATCH* parameter to avoid program prompts, and the *SHUTDOWN* one to force machine shutdown after process completion.

Note that before uninstalling, you must remove manually the boot protection mechanism if installed.

Note also that after the deinstallation process is finished, you will be prompted to reboot the machine, and the immediate reboot may take longer than usual, since it is then when all SecureEntry files will be physically erased.

Differences from SecureEntry 2.0

This version of SecureEntry has lots of improvements over previous versions, being basically a much improved superset of that version :

1. SES (Security enabling services) architecture based, Which improves robustness and security, besides being a follow on for IBM OS/2 security based strategy.
2. Seamless integration of new components :

   The windows behavior (system_menus) component, allows you to set up the initial position of any application window, plus deactivating system menu entries.

   The windows list component, allows you to control and setup the aspect of the switch list, as well as its behavior.

   The Floppy restrictions component, allows you to restrict access to the floppy disk drive to any user/group, besides allowing simple encipherment of the diskette drive contents.

   The SES behavior component, allows you to choose on various system level functions in a per user/group basis, including the action to take on Ctrl-Alt-Del, What bitmaps to use, and/or how to configure the screen saver function.

   The File access component (Tree-Lock), allows you to restrict access at a file system level to files and/or directories.

   The SecureEntry Launchpad component allows you to configure launchpad models for your different groups/users.
3. Much broader scope. Now SecureEntry supports three types of basic installations, from standalone to centrally administered installs, not requiring Lan Server if not desired. (Thus, incorporating its own user's registry).
4. New Logon Modular Procedures architecture, which makes it possible to build a custom-written module to unify the signon process to your subsystem of choice.
5. Rewritten administration tools (batch and interactive), which allow easily to configure your users/groups database from any workstation. (Now all workstations are 'administration aware', and it only depends on who did sign on (administrator or user), whether these functions will be available. Note also that the administration tools are common and independent on the environment setup.
6. Improved granularity. Now you can assign any security component to any user or group, being user profiles always overriding existing component group profiles.
7. Improved and simpler installation routines, still allowing for batch and/or interactive installs.
8. Rewritten UCM support. For centrally administered installations, UCM has been rewritten, and the SecureEntry components supporting this alternative solution have also been rewritten, allowing for much more flexibility and robustness, plus great performance enhancements. Note that UCM is not included in this package itself.

Common questions and tips

The following list provides tips and answers to common questions:

- Installation tips
- Boot/signon/sign-off problems and tips
- Configuration/administration tips
- Miscellaneous tips
- UCM common situations

Installation tips

This section describes solutions to common installation problems.

What is the best way to prepare a system to install SecureEntry 3.0 ?

If you choose to use the Security Enabling Services, then since SecureEntry 3.0 requires OS/2 WARP, SES and a fixpack level of at least 17, and because OS/2 fixpacks are cumulative and provide service code for SES also, then for new machines, the fastest sequence would be :

1. Install OS/2 Warp
2. Install SES
3. Install the current OS/2 fixpack level

Note that if you do it in the reverse order, you may lose the service modules that the OS/2 fixpack could have for SES.

I get the error 'UNPACK32 ERROR' or similar... What is wrong ?

If you receive unpack errors while installing, then either :

1. The original SecureEntry diskette bundle files are damaged, or
2. The unpack program does not 'understand' the packed bundle files.

For the first case, it is easy to verify from your sources the correctness of your diskettes... The second one can only happen because you have installed an old version of the FTCOMP.DLL library. This could happen if you had installed in this same machine BranchCare, or other product which uses this DLL. In this case, you can replace the DLL with the one shipped in the first SecureEntry installation diskette.

After installation, what userid and password should I use ?

Please, read the chapter of this same document titled What to do after installing.

How can I know the build level number for my installed product ?

Other than reading at the install/service window while applying this driver, you can always type the file 'SENTRY.SIG' located in the install directory within your SecureEntry path.

How can I distribute my own add on's or profiles with SecureEntry installation ?

Read the Personalized installations chapter

Can I provide service to a machine with another NLS version of SecureEntry ?

Yes, but the workbench object titles will not appear in the new language. You will have to erase the workbench shadow and underlying objects and recreate the workbench by executing the 'EDYCRWRK.CMD' command located in your SecureEntryPath, install directory.

After installation/service the workbench contains missing objects...

This is normally an indication of some small corruption in your system ini files. You can use the EDYCLINI program to attempt to fix this, and then upon reboot, delete the workbench and underlying objects, followed by a recreation of those using the 'EDYCRWRK.CMD' command, located in your SecureEntryPath, install directory.

The service procedure hangs. What should I check for ?

There is one case in which the service procedure will look like 'hanged'. It is when you have activated a Treelock profile which does not let cmd fullscreens to be launched. The underlying subsystem sees the 'access denied' message and attempts to reopen the fullscreen. You will have to reboot the machine, remove the treelock profile and retry.

I get massages installing about too many open files, not enough handles or copy problems...

It may be the case that you have installed in the target machine other software that is using most of the WorkPlace file handles. In order to overcome this situation, edit your config.sys and add the line :

```
SET SHELLHANDLESINC=30
```

I am installing over Warp 4.0, are there any tricks ?

Warp 4.0 (Merlin) support is fully completed. It will work fine. Just remember, before installing SecureEntry, to install the SES component (in case you want to use the real thing instead of SecureEntry SES emulator), which can be done by choosing the security component from the Selective install Operating system components list.

It is possible that you encounter a hangup after installing SecureEntry in the first reboot. This problem is supposed to be fixed by Merlin FP1, but as a workaround, a reboot will normally be enough to bypass it and continue working.

I have just installed. Where is my launchpad ?

The default setup for SecureEntry is to not activate any launchpad. If you want to use the one you had before, just drag the *Launchpad* object located in the *Installation tools* folder into the opened *EdyStart* object within the *NOUSER* folder. You can also create a SecureEntry launchpad profile and place it in the *NOUSER* folder as a new default component profile, or assign it to users/groups as desired, using the administration tools.

I find missing or hidden objects after uninstalling

There was a bug in previous versions of SecureEntry that could make some original object styles not to be stored correctly. To cure from this :

1. Install SecureEntry in standalone mode.
2. Create a binary profile with default attribute set to visible for all objects.
3. Run EDYCLASS /U profilename to set the default styles to those created in the previous step.
4. Uninstall SecureEntry

How to install SecureEntry with other applications that use SES (i.e, Tivoli)

For now only coexistence with such applications is supported. Before you actually install SecureEntry, you must decide whether the session control functions are going to be managed by SecureEntry or the other SES client product.

If you decide that SecureEntry is to manage session control (logon, logoff, lockup,...) then first of all you have to find a way to disable the session control panels in the other application. After this is achieved, install SecureEntry in SES coexistence mode as explained underneath, with the *stealSES* option 1.

If you want the other application to manage the session control flow, then not only you have to install SecureEntry in coexistence mode with SES applications, but also define the required environment variables so that its own session control flow functions are disabled. You can do this by installing with the *stealSES* option 2, as follows.

Installing in coexistence mode with other SES applications

In any case, use the following options :

1. Do **NOT** install treelock support
2. Do use the SES emulator

Then, define the environment variable *SET SGM_STEALSES=x* , where x is the option you choose before installing (1 for session control flow managed by SecureEntry, 2 for Session control flow managed by the other application), and proceed with the installation.

Note that option 2 can also be activated at installation time, regardless of the *SGM_STEALSES* setting if SES is found installed and running during the installation process, and you have choosen to use the SES emulator instead of real SES. To clarify things a bit, if the other SES application is already installed and running, you do not need to define the environment variable for SecureEntry install to notice that coexistence mode is required.

The only difference between both options is that option 2 will also define in your *config.sys* file the following environment variables :

```
set sgm_edylk_show=no
set sgm_back_bitmap=no
set sgm_ses_cad=yes
set sgm_ses_inactivity=no
set sgm_hide_wait_dlgs=yes
```

After installing (before rebooting the machine), and in case you were using option 2, you may wish to configure a *edycust.cmd* user exit file to automatically logon any user as guest, or the desired user, by for instance, invoking EDYUTIL within the UserExitBeforeLogonDialog with the appropriate parameters (read about it under Programming user exits). In any case, notice that the default user for standalone SecureEntry installations is setup with the password expired, so the first logon should attempt to change this password to succeed. You may also need to define a SES security profile in the *NOUSER* folder, so that automatic lockup is avoided, and a desktop restrictions profile (EDYDESK.INI) which suppresses the SecureEntry 'end session' option from the desktop popup menu.

Remember that you can setup this user exit/profile files to be installed in batch mode automatically as explained under Personalized installations.

Integration with NSC/2 (Network Signon Coordinator)

SecureEntry provides with an NSC/2 Logon Modular Procedure, which you can use in place of the RACF or RACF emulator ones. This section explains how to customize and use it in your environment:

INTRODUCTION

This logon modular procedure allows you to integrate the **NSC/2** (Network SignOn Coordinator/2) productivity tool within the SecureEntry logon flow. This procedure must be configured as the main (synchronizer) one, due to the fact that it lacks the ability to force a given user's password at signon time, thus not being able to play the role of other password synchronized subsystems by SecureEntry.

When you perform a SecureEntry logon, password change, or logoff, you will be performing, respectively, a logon, a password change or a logoff to the subsystems defined in the **NSC** configuration file, as well as to all the other configured regular SecureEntry subsystems.

By means of the environment variable SGM_NC you can decide which of the subsystems defined in the **NSC** profile will work as password synchronizers for the rest of SecureEntry LMPs. If you define more than one **NSC** subsystems as synchronizers, then SecureEntry passwords will only be forced, if each and everyone of the NSC synchronizers can logon successfully.

INSTALLATION

Follow the underneath instructions to install this LMP:

1. Define this LMP in the **EDYSSLMP.DAT** file at the second position after the **SS** definition. To define this LMP use the **NC** identifier and use parameter **0**. Remember to erase your previous synchronizer definition, if any.After you include this subsystem in the **EDYSSLMP.DAT** file, it will look more or less as follows:

```
SS 1 EDYDOM
NC 0
...
Rest of subsystems
```

2. Define, if necessary, the SGM_NC environment variable in your *config.sys* file, as needed.
3. Specify in your **CONFIG.SYS** file the SGM_LS_IFLOGGED environment variable with the **FORCEUSE** value.
4. Reboot your machine.

IMPORTANT CONSIDERATIONS

If you decide to use this logon procedure, keep into account :

It may be advisable to protect the **NSC** configuration files against write access by unauthorized people in order to avoid security exposures. Only administrators ought to be able to modify NSC/2 configuration files. You can use the Treelock SecureEntry component to restrict access to this files.

The Lan Server logon will be performed to the domain specified in the SecureEntry logon dialog and not to the lan domain specified in the **NSC** configuration file.

If the different NSC passwords become desynchronized,for instance, because a password change request is processed while some subsystems are not available, follow the next procedure to get them synchronized again and be able to signon :

1. make sure you that all of the NSC/2 subsystems are available.
2. Try a SecureEntry logon with password change. Supply as signon password the password for the first subsystem that is not synchronized and as new password the password for the synchronized subsystems. You will notice that the *invalid password* message is returned. However, the password change request should have been honored, thus synchronizing the password for the intended subsystem. Repeat this step for the rest of NSC/2 subsystems to synchronize.
3. Try a SecureEntry signon again, this time, it will succeed.

Boot/signon/sign-off problems and tips

This section describes solutions to common problems during a work session with SecureEntry related to event handling.

I get a signon error saying that the system can not open the file 'EDYREGDB.VLB'. What is wrong ?

This file is necessary to be accessible right after the physical signon in order to extract the security profiles from it. Verify the following points :

That you have installed SecureEntry in your Lan Server domain controller, specifying, at installation time, that it is the security profiles server, either by responding *'YES'* to the interactive dialog, or by using the *'SERVER'* parameter for unattended installs. If you are not sure, look for the file *'EDYREGDB.VLB'* in the server's SecureEntry path, NOUSER directory. This file is created only in the SecureEntry servers.

That the user logging in is a Lan Server administrator one, or

That the user belongs to a SecureEntry group (group name starts by letters 'SG').

If the user belongs to a SecureEntry group, then verify that this group has an access defined to resource 'SGMSHELL' of the domain controller with 'RWA' Attributes. This access is provided by SecureEntry administration tools when defining a SecureEntry group, but can be missing if the group was defined through the Lan Server admin. tools directly, or you have deinstalled and reinstalled SecureEntry in the server machine after creating the group.

Check that the Lan Server resource 'SGMSHELL' is currently being shared in the server machine.

Finally, this error can also be caused by an old bug existing in the MPTS code. Verify that you have applied fixpack level WR08610 at least.

I get an error at signon saying 'Lan Server Fail'. What is wrong ?

The first cause of this error is that the edysrv.exe program is not running at the domain server site. This program should be started with a detach sentence within the 'EDYSTART.CMD' file of the domain server machine. Note that this line is normally placed by SecureEntry installation. Remember that the correct sentence for starting it under IBM Lan Server environment is :

```
DETACH EDYSRV.EXE /N:EDYdomainname

where domainname is the name of your domain.
```

A second possibility may be that either the Lan requester program is not running in the requester (client) machine, or that the server is unreachable, because of a logical or physical breakage of the lan.

Yet another cause for this sort of problem can be a mismatch between password maintenance rules for Lan Server against RACF, if this one is being used. For instance, you try to use for an user the same password that was used two weeks ago, and RACF allows for that, but Lan Server does not because it matches one of his last memorized passwords for the user. This is why it is recommended that, when using RACF, you reset the Lan Server password restrictions to a less strict ones than those of RACF. For instance, The max. and min. password length, the uniqueness factor (historic memory for used passwords), etc.. You can modify this parameters using the NET ACCOUNTS command after signon as an administrator. Note that in this situation, the only way to be able to be able to signon with the 'problematic' user is to delete his LAN definition using EDYERASE and change his RACF password.

Failing all of the above, please verify the user definitions at the UCM repository, since it is probably one of those that does not follow the required Lan Server required rules.

How can I debug the processing of EDYSTART.CMD ?

First of all, you can debug it normally during a user session with the EDYLKINI.EXE utility. But for those cases where errors only happen during boot, you can do the following :

1. Rename the file 'SecureEntryPath\DLL\EDYLKSTR.DLL' to another thing.

   From now on the machine will boot through execution of STARTUP.CMD

2. Build a STARTUP.CMD file with the contents :

   ```
   EDYLKINI /NOMODAL
   ```

3. Now you can boot and debug the processing of EDYSTART.CMD

   Do not forget to leave things as previously, once you are sure the EDYSTART processing is correct.

I cannot signon to the machine, or the machine does not start. What can I do ?

Being SecureEntry a security product, it can prove itself especially difficult to go after this kind of problems, since the same mechanisms that protect the machine under normal operation will now interfere the diagnosis and correction of the error.

Just to start, you will have to remove the boot protection feature from the machine, if any one is installed. Refer to the appropriate chapter for further information on this step.

After this, you will be able to, at least, boot the machine from a DOS bootable disk, or OS/2 boot diskettes (installation diskettes 1 and 2), up to the point where you will be able to edit your config.sys file, and add the following line at the beginning :

```
Call=c:\os2\cmd.exe
```

Then, and up on rebooting the machine, you will be able to investigate a bit further. First of all, and if you are in a hurry, verify the following :

That the LIBPATH sentence in your *CONFIG.SYS* file contains SecureEntry directories and SES directories (when using real SES) in the first position.

That the desktop background bitmap is **not** defined as a 4 bpp bitmap. OS/2 does not like them!.

That you do not have installed a very old and expired SecureEntry evaluation copy.

That you have not played/modified your configuration files (i.e, SENTRY.CNF, SENTRY.SIG, EDYSSLMP.DAT, EDYSLA.INI). Note that some of these reside in the NOUSER directory.

That the SYS_DLLs required by SecureEntry to startup are correctly registered in the *OS2.INI* file, as explained under EDYWINI., as it could happen if you have just rebuilt your system files through *MAKEINI*

That you do not have user exit code or startup code that enters in an endless loop.

That you have not just applied service or installed SecureEntry under OS/2 Warp 4.0 (Merlin). If this is the case, and you have not applied Merlin FixPack 1, then probably just by forcing a reboot of the machine, the error will not reappear.

Erase any instance of the file 'EDYTRDSP.INI' that may be left in the machine, as well as 'EDYTRC.DAT' and retry boot. If you have been playing around with the trace utilities and its profiles have been left in an unconsistent state due to an abnormal IPL, it is possible that the machine is looping when attempting to read those.

If the error persists, then keep on pursuing the problem by making sure that :

1. That the EDYSTART.CMD file has been completely processed. Look at the 'EDYLKINI.LOG' file that the startup process writes for this purpose. Remember that you will see in this file all messages that your own edystart file displays through the edylkmsg command.

   Once you are sure about the startup process, you should be able to see the signon panel, and attempt to logon. If this is not successful, keep on with the list :

2. Check that the user and password you are trying to signon with are valid, correctly configured and activated. You may have to go to RACF and UCM definitions, or Lan Server administration to do so. If you are not sure, then try to signon with another userid for which you are sure that access is granted.
3. In Lan Server and/or UCM environments, make sure that the EDYSRV module is running in the server machine. This module is the one that acts as a 'gateway' to the host before the real signon is done.
4. Also in Lan Server environment, verify that the user you are issuing logon with has access to the alias 'SGMSHELL'. This is the alias which defines the access to the security profiles database.
5. Make sure that your user exits are working properly. You can add 'beeps' or trace points in the EDYCUST.CMD file to know the logon points reached.
6. In order to know the operations performed during logon in UCM environments, check at the EDYADMIN.LOG file. It may be a good idea to set the SGM_SL_LOGMODE environment variable to 'TEST' value in your config.sys to see the maximum of available information.
7. If everything else fails, then you can start the SecureEntry traces in your EDYSTART.CMD file, and analyze the resulting trace file or report it to the lab.

I start lan requester and peer in a per user basis, and can not restart after sign-off

When you sign off, SecureEntry does its best to kill all user tasks started during the ending session. Unfortunately, if you load system and/or base software during your work session (i.e, not during startup processing, but after logon), some times this software can not be killed in an orderly way and it may be that a daemon running at ring 0 does not even permit SecureEntry to kill it. This is the case of the WKSTAHLP process of the peer services. The only safe way to unload it is by a Lan Requester command. So, if you intend to allow peer services to be started by signed on users, then adding the following line within the user exit before imminent logoff is mandatory :

```
NET STOP REQ /Y
```

Refer to the lan requester command reference for more info about this command.

Signon/sign-off performance. How to tune it

First of all, read Fine tuning SecureEntry. Performance considerations, to know what can be done to improve performance.

Then, if you want to confirm your data, and taking into account that the numbers can vary a lot depending on available hardware and software, we have obtained the following numbers :

Environment	Standalone	Lan Server	Lan Server + UCM
Signon time	10 seconds	20 seconds	30 seconds

Note that this numbers have been obtained with no memory constrained machines, running a 486 DX2 processor.

Logoff hangs the machine on server/overloaded systems, or other signon errors

If you are using Lan Server 4.0, make sure you have installed PTF level IP08227 for this product. Similarly, IP08260 level is required for Lan Server 5.0.

I get 'Another user is already logged on in this machine'

In Lan Server environments, only a single simultaneous domain logon is supported at a given machine. If you have logged on to the domain through edystart.cmd or a user exit, SecureEntry will report this error when attempting to re-logon. Note also that the machine will in some instances force a domain logon if there is a local logon already done and you issue a NET command (i.e NET SHARE IPC$). To cure from this, you can either force a domain logoff (LOGOFF /D) right before SecureEntry attempts to logon, or touch the environment variable SGM_LS_IFLOGGED to leave this task to SecureEntry.

Configuration/administration tips

This section describes solutions to common problems that may arise when administering SecureEntry.

How can I access more than one SecureEntry Lan from a single requester ?

In order to be able to sign on to different SecureEntry LANs, and provided that you are working under an IBM Lan Server Environment, you have to configure it by checking the 'Enable domain change at sign on' at startup.

Note that this will let you log on only to other SecureEntry 3.0 domains which are of similar configuration (same institution name and administration characteristics).

How can I add my own components ?

You will have to edit the file 'SENTRY.DSC' from the first installation diskette before installing, adding the appropriate components. Alternatively, if you want to update an already installed workstation, use the command UPDATEDB located in the SecureEntryPath, INSTALL directory. The syntax of this command is :

```
UPDATEDB descriptionfile

Where descriptionfile is your modified SENTRY.DSC file.
```

Take into account that :

You have to be logged on as an administrator.

Modifying the SecureEntry base components definitions may cause the database not to recognize already assigned components.

Can I change password expiration interval, Max. signon attempts or min. password length ?

Yes. To do so,

If your installation is a standalone one (no Lan Server), then You will have to edit the file 'SENTRY.DSC' from the first installation diskette before installing SecureEntry 3.0, setting up your preferred values.

For machines that have already been installed, read the previous question in order to know how to run 'UPDATEDB' to activate this changes.

If your installation is of Lan Server type, then refer to the appropriate Lan Server documentation in order to do so (Book : *Network Administrator Tasks*, section : *Managing Users and Groups*, Chapter : *Setting a user password expiration period for a domain*).

How can I write my own user exits ?

You can either program those in C (there is a skeleton and necessary files within your SecureEntry path, API\SOURCES\EDYCUST directory), or in REXX (the skeleton is placed in the same directory). If you write an EDYCUST.DLL, then place it afterwards within your SecureEntry path, DLL directory. If you write those in REXX, then place the resulting 'EDYCUST.CMD' in the SecureEntry path, EXEC directory.

All of this is fully detailed under Programming user exits.

How can I change the startup bitmaps ?

Edit a SES behavior profile, and setup the lockup background bitmap and startup bitmaps of your choice, and then copy it with the default SES behavior component name 'EDYSES.INI' to the NOUSER directory within the SecureEntry path. They will be used at startup.

I get 'LS API Error 53' when accessing group definitions. What is going on ?

The error 53 means network path not found. You probably have an alias defined to a nonexistent path.

I can not see objects in the tree view of my drives objects

Each drive object by itself is considered a desktop object, but not its contents. When the workplace goes to create the tree view, it sees no object to expand its tree from (hang on the underlying tree) if the default object restrictions visibility setting is set to off, so it does not go on with the tree expansion. If this is not the behavior you want, then the solution is to add to your desktop restrictions profile an entry for each drive object you want to make visible with its visibility style set to on, or change the default visibility style setting of desktop objects to visible.

Note that setting the profile to not apply restrictions to the non-desktop objects would not correct the situation since, as before said, the drives objects are really objects belonging to the desktop.

Objects do not open in an autostart personal folder

Probably the objects to open are non visible, and as such they are not shown and not found within the folder when running the autostart thread. If you want the objects to be invisible, but be opened anyways, you should make them visible but belonging to an invisible folder, so that the user can not find them even if visible.

How are desktop object positions managed ?

Read and follow the section Managing desktop objects position.

How can I override the default logon/unlock panels ?

You have to make use of the logon and lockup user exits, coding a call to your own logon/lockup panel display programs, which in turn call the 'EDYUTIL' session event launcher program in order to prefill the required input fields. Take into account that your presented dialog (logon or unlock) has to be system modal. Also note that the screen saver function will be fully active during the user exits processing. In any case, never forget to return system modality to the previous sysmodality owner once your dialog is destroyed.

You can interact with the user in the following user exits :

```
EDY_USER_EXIT_AFTER_STARTUP
EDY_USER_EXIT_BEFORE_LOGON_DIALOG
EDY_USER_EXIT_BEFORE_LOGON
EDY_USER_EXIT_BEFORE_LOGON_CHANGING_PASSWORD
EDY_USER_EXIT_BEFORE_UNLOCK_DIALOG
EDY_USER_EXIT_BEFORE_UNLOCK
EDY_USER_EXIT_AFTER_UNSUCCESSFUL_UNLOCK
EDY_USER_EXIT_AFTER_UNLOCK
EDY_USER_EXIT_BEFORE_LOGOFF_FROM_UNLOCK
EDY_USER_EXIT_BEFORE_SHUTDOWN_FROM_UNLOCK
```

Also, you may want to read about the **SGM_USER_DLGS** environment variable, to achieve maximum robustness in overriding the default logon/unlock dialogs.

How can I manage object's popup restrictions for non-desktop objects ?

Read the chapter about SecureEntry environment variables especially the environment variable SGM_WPS_NONDESKTOP.

How can I manage the restrictions on printer spooler objects ?

Read the chapter about SecureEntry environment variables especially the environment variable SGM_WPS_PRINTJOBS.

I am attempting to do remote distribution or administration. What problems should I face ?

All this kind of programs work by loading a 'daemon' of some sort which is the agent that really does the work in the background. If you are using a strict treelock profile, and also not starting this 'daemon' as a superuser context running process, then the first thing to do is to locate the name of this process and give it treelock access to all of your system's files. This will allow it to touch files such as config.sys, etc, overriding the SecureEntry implicit and user's restrictions.

For NDM/2, read also the tip about NDM ACTIVATE on how to avoid the activate command from failing if using a diskette drive restrictions profile.

How can I open Workplace objects from a user exit ?

Look at the Sample user exit implementation, for a sample of the usage of the REXX SysOpenObject function.

Note that if all you want to do is open a given object at startup or a given user logon time, it may be an easier choice to drag a shadow of the desired object to the list of shadows editor while editing the *EDYSTART* object in the NOUSER folder, or a given personal desktop profile's dynamic folder, which you then assign to the desired user.

How can I remove the 'Original' menu item from shadow objects ?

You can do this during after logon user exits processing by sending the appropriate SecureEntry setup string to the object. So, for instance

```
/* */
If RxFuncQuery('SysLoadFuncs')  Then Do
  Call RxFuncAdd 'SysLoadFuncs',  'REXXUTIL',  'SysLoadFuncs'
  Call SysLoadFuncs
End

call SysSetObjectData  '<SGM_MAIN_WB_SHADOW>','EDYMENUITEM=REMOVE,ORIGINAL;'
```

Would remove the 'Original' menu item from the SecureEntry workbench popup menu.

Note that this is an alternative to that of using the Windows restriction workbench for the same purpose.

The bitmap for the background or screen saver function, does not look like working

This can be for two main reasons :

The SES behavior profile is not active (either not assigned to the user or group, or not present in the NOUSER directory, or not 'tested'),

Or the referred bitmap file is not present or invalid at the specified path.

See also the next question,

I have configured a NOUSER profile, but it does not get activated

Machine profiles configured into the NOUSER directory are not granted to be activated at every sign-off, unless a profile of the same kind was assigned to the user or group that has signed off. After the next shutdown this profile will become the default one. Note that this is done for performance reasons in production machines.

Other things to check are :

That the profile has the correct default profile file name. (the same as the associated template in the TEMPLATE folder).

That the profile is of the correct type.

How can I deactivate any new menu option for the desktop or its objects ?

SecureEntry, as any other product, will always be behind in development, trying to 'catch' the last features incorporated to the operating system. Similarly, the SOM architecture allows for any third party program to enhance the function provided by the standard workplace by adding new menu entries, which may not always be desired within a protected environment as the one provided by SecureEntry. For this specific need, however, you can use the Windows behavior component and deactivate the required entries by name.

What kind of bitmap files can I use ?

Except for the SES component, desktop background bitmap, for which only OS/2 standard bitmaps are accepted, in general the following bitmap formats are accepted :

OS/2 and Windows Bitmaps

The file extensions .BMP, .VGA, .BGA, .RLE, .DIB, .RL4, and .RL8 are recognized as OS/2 1.1, 1.2, 2.0 or Windows 3.0 bitmaps. The newer multimedia Windows bitmap format that allows 16 and 32 bits-per-plane is not supported. The files are written in OS/2 2.0/Windows 3.0 format.

CompuServe Graphics Interchange Format

The .GIF file extension is recognized as a GIF file.

ZSoft PC Paintbrush Image File Format

The .PCX file extension is recognized as a Paintbrush file.

Microsoft/Aldus Tagged Image File Format

The .TIF and .TIFF file extensions are recognized as TIFF files.

Truevision Targa/Vista Bitmap

The file extensions .TGA, .VST, and .AFI are recognized as Targa/Vista files. This class only supports 8 bit-per-plane and 24 bit-per-plane images.

Amiga IFF/ILBM Interleaved Bitmap Format

The file extensions .IFF and .LBM are recognized as interleaved bitmap files.

X Windows Bitmap

The .XBM file extension is recognized as a X Bitmap file. This class supports X10 and X11 1bpp bitmaps. Some .XBM files with text strings inside look to be sprites or icons and are not supported.

IBM Printer Page Segment

The following file extensions .PSE, .PSEG, .PSEG38PP and .PSEG3820 are recognized as PSEG files. PSEG files are used to include image data in BookMaster documents. PSEG files only contain 1 bit-per-plane, which is always ink on paper, ie: black on white.

I can not run EDYSWLV. it hangs the session

Verify that EDYSWL2.EXE is located in a directory accessible through your OS/2 native PATH environment variable.

I can not customize an object because it requires an Object ID

Read the section About Object IDs.

How to suppress the WarpCenter function

If in your installation you will never use the WarpCenter, do the following changes to your *CONFIG.SYS* file

Remove the *WARPCENTER* keyword from the *SET AUTOSTART* sentence.

Define the line *SET SGM_EDYSC_DISABLE=YES*.

This will remove SecureEntry functionality and make warpcenter not to be opened at startup.

Note however, that if some of your users will use warpcenter and some others not, then your only option is to remove the profile *EDYSC.INI* from the *NOUSER* directory, and assign appropriately a warpcenter profile to the required users/groups.

Miscellaneous tips

This section gives generic tips for working with SecureEntry.

What's the order in which security profiles are accessed / activated ?

Each component has its own default profile settings. These settings are overridden by any profile for the component placed within the SecureEntry NOUSER path, which is in turn overridden by the the current logged on user's SecureEntry group profile, which is in turn overridden by any existing profile for the given component assigned directly to the user.

How can I run a unattended/centralized administration policy ?

Ask about UCM. UCM is a corporate administration SecureEntry plug-in product which allows you to administer your LANs from a Host site, among other features.

How can I integrate my own network more seamlessly ?

If you do not have IBM Lan server, you can run with the 'other networks' support that SecureEntry provides, but that is not all that can be done. If your network has an interface for signon/sign-off, it is quite easy to program an LMP (Logon modular procedure) which will sign you on to your network at SecureEntry signon time, synchronizing the password to the one the SecureEntry registry stores. Read the programmers reference in order to do so.

Is there any administration activity log file ?

Yes. The file is named EDYADMIN.LOG, and located where the environment variable SGM_DB points to. It is a text file which you may want to purge in a periodic basis. Note that you can setup what gets logged there by setting an environment variable in your config.sys named SGM_SL_LOGMODE with the following values :

```
SET SGM_SL_LOGMODE=ALL      : Log all activity
SET SGM_SL_LOGMODE=UPDATES  : Log only activity which modifies the database.
                              This is the default setting
SET SGM_SL_LOGMODE=NONE     : Log no activity
```

Backing up a SecureEntry installation

You need only to backup from the server machines (or standalone machines), the following files :

SecureEntryPath\NOUSER*.*

SecureEntryPath\INSTALL\SENTRY.CNF

SecureEntryPath\DLL\EDYCUST.DLL (if any)

SecureEntryPath\DLL\EDYFILT.DLL (if any)

SecureEntryPath\EXEC\EDYCUST.CMD (if any)

Being the rest of files not specific to your installation.

What settings can I touch from the SES config.sys entries ?

Basically, the following variables are supported with its possible values by SecureEntry :

GUESTNAME

This environment variable defines the user name to use by SecureEntry when a guest logon has been performed. The default setting is 'GUEST'.

AUTOGUEST

This environment variable tells SES to directly start in a guest logon user session without launching a real logon event. Since SecureEntry SLA does require a real logon event to activate and deactivate security profiles, you are encouraged not to change this variable default setting (value='NO'), unless you are using your own written SLA program. In case you want to achieve the same effect with SecureEntry as provided, you can use any proper user exit before logon to force a guest logon.

TRUSTEDPATH

This environment variable forces logon/unlock events to start automatically after the corresponding logoff/lock event has been processed (when value is set to 'YES'). If you set this variable to 'NO', then the events will be started by a key press.

RESTARTUSERSHELL

This environment variable allows for choosing when should the second PMSHELL be restarted . The 'YES' value implies that the PMSHELL is to be started in a per logon basis, where the value 'NO' (default setting by SecureEntry installation), indicates that PMSHELL is to be started only once, at startup time. Normally this value is the one which provides for best results in terms of performance.

Refer to the SES documentation for further explanation on those.

Is there any other information available ?

The following can be used to obtain further documentation about SecureEntry :

Do not forget to read the readme.doc file that comes together with the present driver.

The online manual 'UCM administrator's guide' can be used as a reference about installing UCM and debugging UCM related errors.

Most of the graphical programs that SecureEntry provides contain a detailed online help for its operation.

There is also paper printed version of this document, which may explain in more detail some of the features explained here.

The Lan Server documentation may help you understand better the administration and configuration related tasks for this kind of environments.

The SecureEntry 2.0 documentation is also a good reference for the desktop restrictions workbench and VDM utilities, plus startup component.

How can I provide NLS support for other than the supplied language ?

You have all the necessary source material in the SecureEntry path, API\NLS subdirectory. You can do the necessary translations and generate the required object code. Note however, that if you just want to translate for the runtime environment (session control panels), you only have to translate the EDYERROR.TXT file and make

a new message file EDYERROR.MSG through MKMSGF, placing it in the appropriate directory (SecureEntryPath\EXEC).

How can I grant CM/2 comms sessions start in a per user basis ?

The rule of thumb that SecureEntry uses for isolating user sessions is to kill at logoff time all processes that were started after the logon was issued. This may not be good enough in some cases where the base software is always up and running, but the user sessions (real units of work) are established within those. This is the case of, for instance, 3270 emulations, or other applications that have a quite high startup time that would penalize the signon performance if started at each logon. The idea for dealing with such applications is to use the SecureEntry provided user exits (before logoff) to make sure that the logical working sessions are stopped and restarted every time a logoff event is processed. In the case of CM/2 3270 emulators, you can use the tool 'EDYE3270', as provided by SecureEntry to do this activation/stopping automatically.

Alternatively, you can use also the CM/2 command 'CMLINKS' to halt and reactivate the communications links, thus effectively bringing down any logical session that could be left opened by the outgoing user. The suggested commands are :

```
CMLINKS H *       to stop (halt) all links
CMLINKS A *       to activate all links
```

Note that using this command will reset ALL links of the machine, so beware when using it on gateways. i.e, specify only those links you really want to deactivate instead of an asterisk.

How can I avoid users to boot with the ALT-F1 combination ?

There are several things you can do to avoid ALT-F1 :

1. If you want to avoid it *completely* and do not care about receiving an error message, then just rename the file \OS2\BOOT\ALTF1*.SCR to anything else.
2. If you want to avoid this error message, then :
    1. Leave the .SCR files alone
    2. Change the file \OS2\BOOT\ALTF1.CMD so that it only processes the desired entries (you have control over V,M,1... options). The file is a regular .CMD file, and just placing a 'goto end' in the desired option routine processing part would do it.
    3. If you want to avoid also the 'C' option (goto command line), then note that this option uses the \OS2\BOOT\CONFIG.X file, so renaming it will avoid the function to operate, and changing it to force a logon would ensure it does not break the security.
    4. Now, you can optionally change the ALTF1TOP.SCR file to show only the options you have enabled.

Can not run a NDM/2 ACTIVATE command

NDM checks to see if there is a diskette present before accepting the activate command. When SecureEntry is restricting access to the floppy disk, this check fails normally with a more severe error, and NDM decides to cancel the activate command ... just in case.

In order to make NDM skip this checking, set the following environment variable in the target machine :

```
SET ANXCHECKBOOT=NO
```

A second problem you may face, is that the activate command requires to be able to access and modify the config.sys file. Note that this may only happen if the NDM 'daemon' is running under user context (i.e, not started through config.sys, edystart.cmd,...). In this case,you must use in this type of environments, treelock profiles that override the default restriction of not letting any executable to modify the config.sys file for the NDM processes. You can do this by adding the following line to your treelock profiles :

```
F NdmPath\ANXCMCLx.EXE
```

Where *NdmPath* is the path where your NDM executables reside, and *x* is 'C' for NDM client machines, 'S' for NDM server machines.

UCM related common situations

This section contains recipes and solutions to common errors and situations when working with UCM.

I get 'SQL error -805' when attempting to start the administration utilities

Make sure you have created the corresponding packages and correctly bound the UCM administration access plans, as explained under Setting up the User Centralized Management workstation Note that if you have applied service to the UCM management workstation, it may be necessary that you rebind the plans to the host database.

I get 'SLAG -8002E: Dynalink error' from the UCM Administration applications

```
SLAG -8002E: Dynalink error EDYUCM01.
```

This means that you do not have A DDCS/2 license. You must Purchase a DDCS/2 license an update the nodelock file in the DB2/2 directory.

I get 'SQL error -204...' from the UCM Administration applications

```
SQL error -204 "Name" is an undefined name.
```

The program does not find the UCM tables in the DB2/MVS.

You must check the synonyms defined in MVS. You can use the member UCM.V30.DB2(CREASYNR) as an entry model with SPUFI.

I get 'SLAG -1013E: Dynalink error' from the UCM Administration applications

```
SLAG -1013E: Dynalink error EDYUCM01.
```

It could mean that you attempt to access a database alias name that could not be found. Ensure that the DDCS/2 system database directory is properly configured. Check if the environment variable SGM_UCM_DBDFT is defined in the UCMADM.CMD or similar file and has the correct value.

Of course, do not forget to verify that the file EDYUCM01.DLL file is found within your libpath.

Is there any performance tuning for UCM that I can do ?

If you plan to have many workstations or many branches in your scenario, you can tune the following parameter:

In the VTAM major node definition for UCM in SYS1.VTAMLIST or in the appropriate library, define EAS = 3999. This parameter sets the maximum session number within that LU. This definition does not mean any resource wasted at all.

Optionally, in the Communications Manager/2 Setup, in the DLC configuration, you can check the "Free unused links" check box. This will cause inactive sessions to become free. Note that this parameter may produce problems if you use CP-CP sessions, since these sessions are permanently allocated.

Other than this, and for UCM specific performance tuning, please read the related information as explained in the appropriate chapter of the UCM administrators guide.

How can I maintain standalone and Lan Server installations with UCM ?

If you use the Refresh branch online procedure, then this environment is supported. The EDYSRV process will discard all resources information when downloading branch or users data to a standalone workstation.

WHAT'S NEW !!

This chapter is intended as an 'easy guide' for upgraders. It lists in chronological, descending order, the links to the appropriate documentation points where new function has been added to the product. Note that for a full reference of fixes and additions, we encourage you to look into the README file supplied within the first installation diskette image, since the ones down under are only what we consider main improvements to the product.

New SecureEntry log files browser utility.

Treelock \DEV\ support. Allows for restricting access to system devices. Remember to specify the internal device name (i.e, \DEV\PCMCIA$ instead of PCMCIA.SYS)

New support for IBM Network Station series 2800 clients, when running under WSOD.

New SecureEntry/2 tutorial, distributed as a separate package file.

New Processes Auditor component for CPU usage measurements.

New NSC/2 LMP for password synchronization.

New WorkSpace On-Demand support

New Support for coexistence with other SES clients (i.e Tivoli).

New support for the RACF emulator, to be used in UCM installations.

New Public Applications component for LAN Server environments.

New sample logon and unlock dialogs

New EDYRVUCM utility to correct the change tables of the UCM Database.

New EDYQRYBR utility to query the level of all your branches.

New EDYPHOTO utilities to obtain a machine configuration image file for servicing and error reporting.

New UCM Refresh branch method to update definitions of groups and resources in Lan Server and new UCM Logging facility.

New included Security Enabling Services emulation. You can now install SecureEntry without the real SES code. Note, however, that FP17 or superior are still required if you plan to use the treelock component.

New WarpCenter component for Merlin systems : The WarpCenter component.

Increased functionality of the SES component. You can now configure the desktop background, and avoid/hook specified systemwide hotkeys. Read about it in the SES behavior component chapter.

New section in this document about Error codes and messages.

New Hooked objects component.

New EDYDUMP utility.

About processes and running contexts improved function.

New File integrity check SAF2GEN utility.

The Treelock component got redesigned new documentation plus several more neat features, including a fully graphic profiles editor.

New Master boot record saver : EDYRWMBR utility.

The Desktop restrictions component got support for directory propagation of styles, and new Merlin menu entries.

New diskette translator utility within the The Floppy restrictions component.

New shortcuts component.

New EDYLOGFS utility to automatically control the logs file size.

The new Personal desktop component.

Within Setting up a backup domain controller for Lan Server you fill find how to set up the machines for this new support.

The new Window list component.

Besides, and if you are a SecureEntry integrator, it is always advisable that you have a look into the Environment variables section and the Programming user exits one, since new environment variables and user exits are quite frequently added.

Contacting us

If you have a comment or find a problem not reported in the previous paragraphs, we would like to hear from you... Please contact the following IDs :

SecureEntry team management

Fernando Velasco Internal : Fernando Velasco Benavente/Spain/IBM @ IBMES
External : VELASCO @ es.ibm.com

Solution responsible

Fernando Trius Internal : Fernando Trius Chassaigne/Spain/IBM @ IBMES
External : FTRIUS @ es.ibm.com

Technical coordination

Ramon Gonzalez Internal : Ramon Gonzalez Compta/Spain/IBM @ IBMES
External : RAMON_GONZALEZ @ es.ibm.com