Administrator Manual

# SECUDE for R/3

Version 2.0



SECUDE

SICHERHEITSTECHNOLOGIE
INFORMATIONSSYSTEME GMBH

SECUDE SICHERHEITSTECHNOLOGIE
INFORMATIONSSYSTEME GMBH
Julius-Reiber-Str. 17
D 64293 Darmstadt

World Wide Web: http://www.secude.com
Support: sapr3@secude.com



Current SECUDE Library: Version 5.2

Document Status: Version 2.0 / January 1999

# Contents

# Preface

## Target Group

System administrators.

## Overview

Chapter 1 gives an overview of the options for securing online communication between SAP R/3 programs using SECUDE.

Chapter 2 introduces the basic concepts of SECUDE for R/3.

Chapter 3 gives an overview of implementation and the procedures used by SECUDE for R/3.

Chapter 4 discusses the basics and concepts for running a certification authority.

Chapter 5 deals with installing SECUDE for the application server and the SAP programs communicating with it.

Chapter 6 explains all known error sources when operating SECUDE for R/3.

Chapter 7 contains a list of the most important terms used.

Study of the SECUDE manuals *SHORT INTRODUCTION TO SECURITY TECHNOLOGY*, *PSE MANAGEMENT* and *CA MANAGEMENT* is recommended, as required.

## Copyright

# 1 Introduction

Users and providers of computer systems in "open" networks, like the Internet or Intranet, face security problems for which very general security requirements can be defined. These include **authenticity** of communication, i.e. a message which appears to be from a certain originator can also be proved to have actually come from this originator. The originator of a message should also be made responsible for it – **non-repudiability**. A message should be readable only by the intended recipient – **confidentiality**. The recipient should be able to detect any changes which may have been made by a third party – **integrity**.

### SECUDE for R/3

SECUDE provides solutions enabling operations to be carried out securely in the SAP R/3 client/server environment. To do this, **standardized cryptographic procedures** and **algorithms** have been implemented in SECUDE. The link from SECUDE to R/3 is made using an interface created by IETF (Internet Engineering Task Force), called the *Generic Security Services API* – **GSS-API.**

### SECUDE Creates Confidence

All parties involved in communications, such as R/3 users, R/3 Application Servers, and SAPlpds, receive unique **digital identification** by which they can be identified. This identification is contained in the user's personal security environment (**PSE**). The PSE is either located on a smartcard or stored in the user´s home directory.

Confidence is built up on both sides using **3-way authentication**, where a user identifies himself to an R/3 application server and vice versa. The information necessary for this is contained in the PSE. What is new here is that the identity of the server is now also checked by the user. This way both participants can be sure with whom they are communicating.

In 3-way authentication between user and server, a session key is generated which is valid only for the current session and is known only to the two participating partners.

### SECUDE Ensures the Integrity of Data

With SECUDE, the data to be transferred are subjected to an **integrity test**. Each packet is given a checksum. The checksum is then signed digitally, i.e. provided with a signature. The recipient can verify the data packet that has been digitally signed by the sender. The recipient, therefore, immediately notices any changes which may have been made.

### SECUDE Encrypts Data

A further security step is data **encryption**, which is carried out in addition to the integrity test. This protects the data from being viewed by a third party. For encrypting, the same key is used which was created for authentication between client and server, and which is available only to these two partners. It is then impossible for intruders to access or manipulate data without being noticed.

**SECUDE Provides Security**

To cope with the known intrusion methods such as impersonation (**masquerade**), **manipulation** of files, **password tampering**, and **monitoring** or **logging** communications, SECUDE offers various protective measures. Each user has a PSE secured by a password (also called a PIN – Personal Identification Number). In this way, the security-relevant and user-related information required for **authentic** and **confidential** communication is protected from third parties. An integrity test immediately recognizes manipulation of information transferred over an open network. If information is encrypted as well, an intruder can still monitor or log communication, but cannot do anything with the recorded data.

# 2 SECUDE Security Technology

Because networks such as the Internet or Intranet are increasingly being used, the information transferred there is accessible to virtually every user connected to the network. Both users and providers of services face security problems that cannot be solved satisfactorily solely by using the usual security measures such as the allocation of passwords.

Logon procedures are necessary which create a level of confidence acceptable to both sides as well as making the exchanged information available only to the respective end users. This is an ideal area for applying asymmetric cryptography, as provided by SECUDE.

By using asymmetric cryptography, the user achieves authenticity by proving possession of a specific key. Confidentiality is achieved by encrypting the message.

This chapter describes how these security requirements can be fulfilled using SECUDE.

## 2.1 Security Using SECUDE

To achieve a solution for security requirements in open networks that is satisfactory and feasible to all sides, two main points need to be considered. Firstly, the technology used should be familiar to and accepted by the user community. Secondly, security should be achieved by a having a "secret" which is only accessible to the user.

Security with SECUDE is achieved by using security procedures that are recognized by the user community, by keeping keys secret, not by keeping algorithms secret. As a result, each user can comprehend and check the security.

SECUDE for R/3 implements recognized symmetric (DES, IDEA) and asymmetric cryptographic functions (RSA). Hash functions such as MD5, SHA-1 and RIPEMD-160 are also used.

The security technology implemented in SECUDE is based on the combination of symmetric and asymmetric cryptography. Every user receives a PSE (personal security environment). This PSE contains all the information necessary for uniquely identifying a user. An R/3 application server or a SAPlpd are also users in the sense of the security infrastructure.

When an R/3 user logs on to an R/3 server, digitally signed information is exchanged. Each side checks this information to see whether it is genuine and valid. If the check is successful, a symmetric key is created. This key is created during the check communication and can only be used for that session.

For maximum security, SECUDE provides the option of using smartcards. Smartcards are the size of a credit card, but their function differs considerably. A smartcard contains a small computer including memory. The advantage of using a smartcard is that the pair of keys

needed for asymmetric cryptography is stored on the card. When the key is required for operations, the data is transferred to the card and the operations are carried out on the card. The card is in turn secured with a password. Therefore, by using a combination of *possession* and *knowledge,* it is impossible for someone to compromise the user's secret key.

3-way authentication used in SECUDE with RSA takes considerably more CPU time compared to encryption and decryption using a symmetric algorithm like DES. However, this demand on CPU time occurs only once, when logging on at the beginning of a session. The communication which follows is secured using the faster symmetric cryptography.

## 2.2 Asymmetric Encryption

Two different keys are used for asymmetric encryption and decryption. These two keys are mathematically dependent on each other. However, one key cannot be calculated from the other key. What was encrypted using one of the two keys can only be decrypted using the other key. These attributes allow making one of the two keys public and keeping the other one secret at a safe location. This procedure is called a public key procedure because one key is made public.

*Figure 1* shows how plain text is encrypted using a public key. This encryption produces an encrypted data set (cipher text). To obtain the original text from the encrypted data set, the private key that matches the public key has to be used.

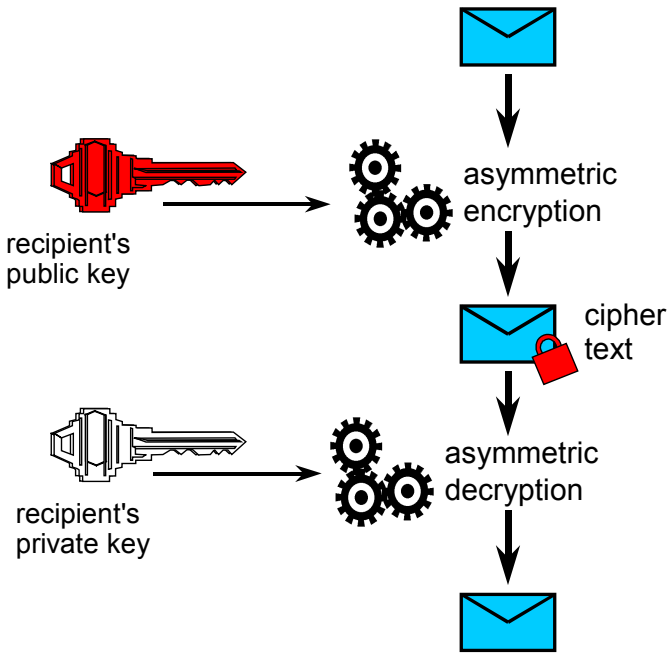


Figure 1: Asymmetric Encryption

Using this procedure a message is generated that only the owner of the private key can decrypt and understand. For logging on to SAP R/3

using SECUDE security technology, the asymmetric encryption procedure RSA is used.

Since asymmetric operations are very CPU intensive, they are often used together with symmetric procedures. This combination is also referred to as a *hybrid procedure*.

## 2.3 Digital Signature

In every day life, signing documents is considered natural and reliable. As soon as a contract has been signed, the signatory can no longer deny performing the signature. A handwritten signature is a generally accepted procedure. However, how can an electronic document be signed? The aim of this chapter is to explain how this is done. Some of the technology required for this, namely asymmetric cryptography, has already been introduced.

The problem with using asymmetric encryption is that it is too slow for large data sets. If mass data need digital signatures, hash functions can first be applied. These functions generate a very small unique output from a large input. The algorithms used are usually very quick. The hash value is then used as the input for asymmetric encryption.

First, a hash function is applied to the information to be signed. The function produces a very small output called the hash value. The hash value can now be encrypted using the private key. The encrypted hash value is called a digital signature. Since no other person has access to the private key, only its owner can execute the operation, making it comparable to a normal "signature".

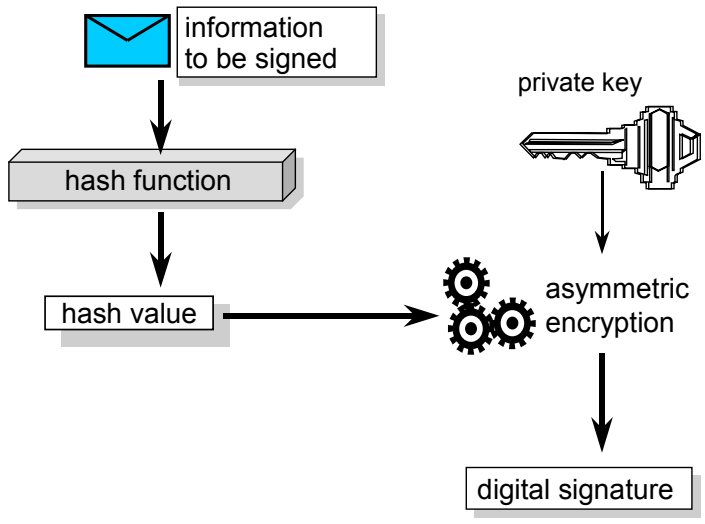


Figure 2: Digital signature

Both the information to be signed and the digital signature are transferred to the recipient. The recipient applies the hash function to the information and the result is a hash value. The recipient then applies the sender's public key to the digital signature to obtain the hash value created by the sender, called the *decrypted signature* – see *Figure 3*. By comparing the hash value of the signed information with the hash value

obtained from the sender's digital signature, the recipient can now determine whether information was changed during the transfer, or whether the sender was the one who actually generated the digital signature.

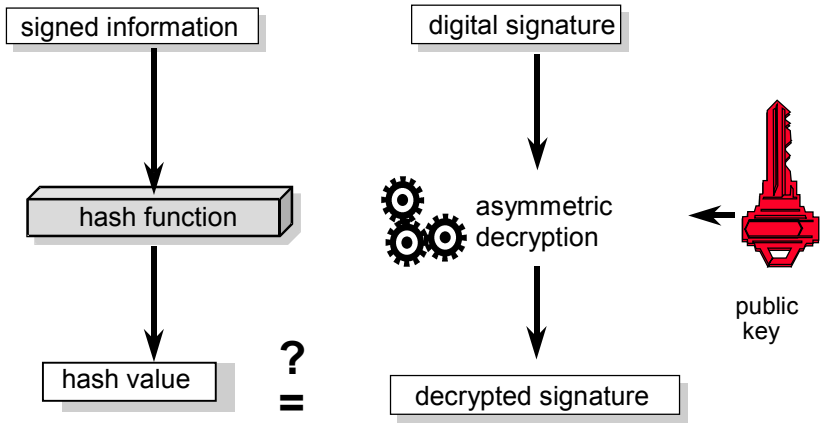


Figure 3: Checking a digital signature

What is gained by using the hash function and asymmetric encryption? By checking the digital signature, the recipient can ensure that the information received actually came from the specified sender, and that it was not changed during transport.

The question arises why the recipient can be sure of having received unchanged information by checking the hash value. For one thing, this is due to the characteristics of a hash procedure; for another, it is due to the private key being stored securely and it being practically *impossible* to calculate the private key from the public key. Good hash procedures have the characteristic that a small change to the information to be signed changes the resulting hash value considerably. For storing private keys securely, there are suitable options, such as smartcards.

The public key for this asymmetric procedure (RSA) is a product of two prime numbers. The strength of RSA is based on the fact that these two prime numbers are required to calculate the private key. This means that the public key has to be factorized. However, for large numbers there are no simple and quick procedures for factorization. One option is to systematically test all possible factors. To test a 200-digit number, all the prime numbers between 2 and $10^{100}$ would have to be examined (worst case scenario). In 1994, a 124-digit number was factorized [Beut-94]. The solution was made possible by splitting the problem into partial problems. These partial problems were then spread over an appropriate number of computers and persons. To factorize the 124-digit number, approx. 600 persons from more than 20 countries were employed for eight months. Resources, both in terms of the required computer capacity as well as time, are therefore considerable and increase exponentially with longer keys.

However – returning to the digital signature – there is another problem. How does the recipient of the information know that the public key used to check the digital signature actually belongs to the sender of the information? What is missing here is a link from the public key to a

person, similar to the photograph, signature, and stamp on an ID card. The following chapter discusses how this can be implemented digitally.

## 2.4 Certification

Each user of a public key has the problem of determining the person to whom the key belongs. Only when this is possible, can a digital signature be trusted. The following procedure ensures the authenticity of the public key.

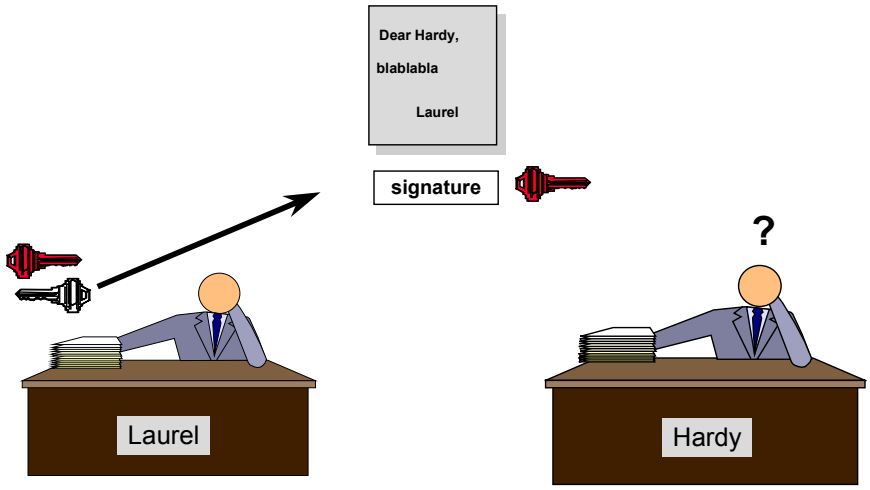


Figure 4: Certification

To link the public key to a person, a third authority is required, called a Certification Authority (CA). The CA checks the identity and links a person's name to the public key of this person by signing it with its digital signature using its own private key.

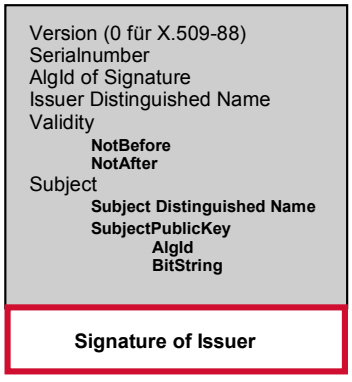


Figure 5: Schematic diagram of an X.509 certificate

The result of authenticating a public key is a certificate. At SECUDE, we use the X.509 standard as a certificate structure. In addition to the public key (*Subject Public Key*), the certificate contains the name of the issuing CA (*Issuer Distinguished Name*), a validity period (*Validity*), the name of the owner (*Subject Distinguished Name*), and a number unique to the issuing CA (*Serial Number*).

For this whole procedure to work, it is necessary that all participating persons have confidence in this CA's public key. If the CA's private key is compromised, all user certificates have to be reissued.

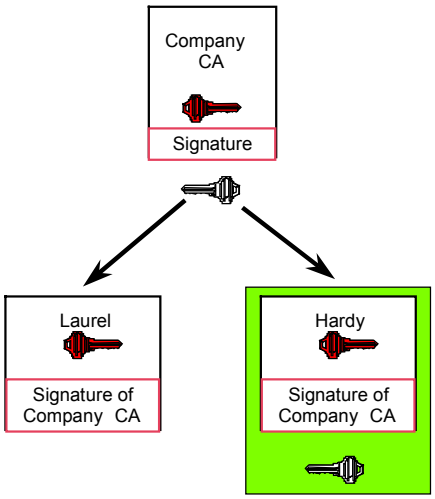


Figure 6: Certification hierarchy

In *Figure 6*, the c*ompany CA* is the highest authentication authority. In theory, the structure can become arbitrarily complex. It is feasible that the company CA has also been authenticated by a CA, and so on. The top CA is the root of the tree, a tree that is upside down, whose leaves are the authenticated users.

The public keys for Mr. Laurel and Mr. Hardy are provided with the company CA's signature. Therefore communication between the two using digitally signed documents is no longer a problem. If Mr. Hardy now sends a digitally signed document to Mr. Laurel (for example, an E-mail) and includes the certificate issued by the company CA, then Mr. Laurel can check the digital signature and determine the authenticity of the signature by checking Mr. Hardy's certificate. To do this, Mr. Laurel also needs the public key of the company CA. He can then check the authentication carried out by the company CA, which created the link between the public key and Mr. Hardy. Changes to the signed document or a forged certificate will be recognized immediately.

All participants, in this example Mr. Laurel and Mr. Hardy, must keep their certificates, their respective private keys, and the public key from the CA safe. If these are required for checking or for creating a digital signature, the participants should be able to access them quickly and easily. The following chapter describes the procedure used by SECUDE to store the sensitive security elements securely.

## 2.5 Personal Security Environment

All users of the SECUDE security infrastructure need security-related information. This information should be protected from third parties and be available to the user only. In SECUDE, a personal security environment (PSE) is used for storing this information. The PSE is protected by a password (also called a PIN) that should be known only to the owner of the PSE.

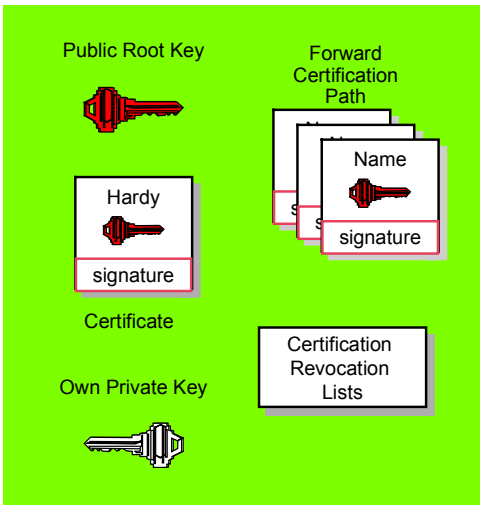


Figure 7: PSE – Personal Security Environment

A PSE contains the root certificate (*Public Root Key*) of the highest level CA, the relevant certificate of the owner, and the relevant secret private key. If certificates from intermediate CAs exist between the user certificate and the root authority, they are also contained in the PSE. These certificates form the *Forward Certification Path*, meaning the chain of certificates that are required for checking a signature.

If the private key of a user has been compromised, i.e if someone else has gained possession without authorization, and the user has noticed this, the CA can then revoke the certificate of the private key. As long as the certificate is still valid, all users need to be informed that this certificate can no longer be trusted. For this, each CA maintains a *Certificate Revocation List*. The revocation list contains the unique serial number of the certificate and the time of revocation. It must be made available to all users of the security infrastructure.

From a technical point of view, there are various PSE implementations. One possibility is a file protected by a password. This file should be in a directory that only the user has access to. The user's home directory is ideal for this, if available. In UNIX, it is commonly /home/username.

Another type of implementation is the storing of sensitive data, such as the private key and the public key of the CA root, on a smartcard. The smartcard is also protected by a password.

The advantage of using smartcards is that the private key, which should be kept secret, never leaves the card. With the card, communication is carried out using a card operating system. This system makes sure that the private key never leaves the card; even the user is not able to read his key from the card. All operations that require this key – for example, generating a digital signature – are executed within the card. A disadvantage of this is slower communication when accessing the card. However, from a security point of view, this is an ideal solution.

# 3 R/3 with SECUDE

SECUDE can be used with two different security interfaces of R/3. *Secure Network Communication*, in short *SNC,* is used both to logon securely to the R/3 system and to guarantee secure communications between the R/3 client and the R/3 application server. *Secure Store and Forward*, in short *SSF*, is used for signing and encrypting data, that are to be used later in this form. Both interfaces are described in a SAP white paper. The implementation by SECUDE is described below.

## 3.1 Secure Network Communication (SNC)

From version 3.1G, R/3 is supplied with an interface that makes it possible to enhance R/3 with the above mentioned security functionality. This interface – Generic Security Services API or short GSS-API – is the link between R/3 and SECUDE. By default, after installing R/3 it is not activated. It can be activated by setting profile parameters for the application server, as described in Chapter *4 Installing SECUDE for R/3 SNC*. This section describes the involved methods.

### 3.1.1 The Security Interface to R/3

So that SECUDE can provide the R/3 system with additional security functions, an interface is required which both products understand. In R/3, the Generic Security Services Application Programming Interface (GSS-API) has been implemented. This interface enables the linking of security products.

The GSS-API has been designed such that the application using GSS-API functionality does not have to know anything about the procedures and algorithms the security product uses and makes available via GSS-API. The advantage is that you can integrate the security functions into applications regardless of the special features associated with implementing certain security procedures.

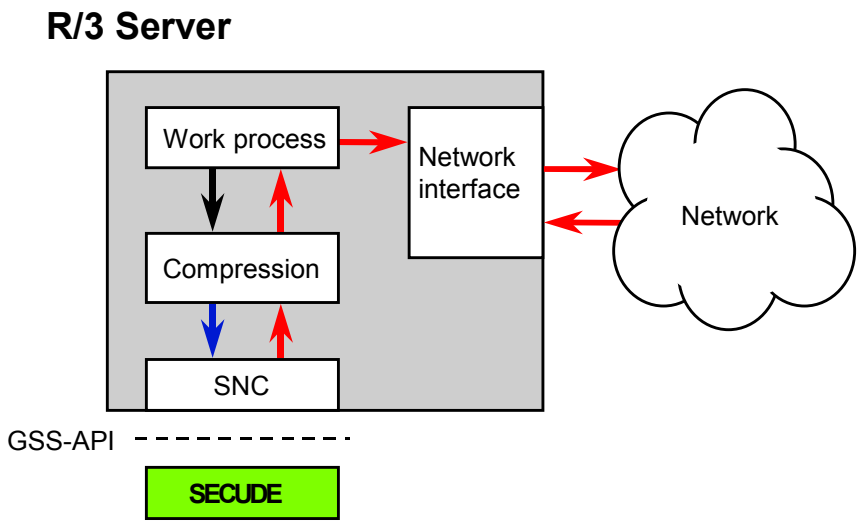


Figure 8: Interface from SECUDE to the R/3 Application Server

The interface from SECUDE to R/3 is displayed as a diagram in *Figure 8*. An R/3 application server communicates with SECUDE via GSS-API. The work process of the application server sends the data to a client. This data is subject to a compressing procedure and is then transferred to the SNC module (SNC = *Secure Network Communications*). The SNC module communicates with SECUDE using GSS-API; in other words, it transfers data by using GSS-API functions. SECUDE either signs, or signs and encrypts the data according to the settings in the security profile. You can configure the algorithm used for signing and encrypting.

SECUDE transfers the secured data to the SNC module. The data passes through the chain back up to the work process to be further transferred to the communication module, called the network interface in *Figure 8*. Here the encrypted data is *packaged* and sent over the connected network.

The client receiving the data must also have a SECUDE security library linked via GSS-API.

The following chapter describes how both participants obtain the required information for secure authentication and communication.

### 3.1.2 Logon Procedure and Communication

SECUDE provides secure authentication and encryption for SAP R/3 based on asymmetric cryptography. Both the R/3 application server, as well as SAPgui, use GSS-API to access SECUDE functions. The security mechanisms in SECUDE are used both for logging the user on to the application server and for protecting communication between SAPgui and the application server.

The logon procedure with SECUDE differs from conventional logon procedures in that, instead of a password being exchanged for logging on, digitally signed information is exchanged. For this, a participant in the security infrastructure (for example, an R/3 user or an application

server) requires digital identification – a certificate. The certificate is found in a PSE. *Figure 9* displays this as a diagram.

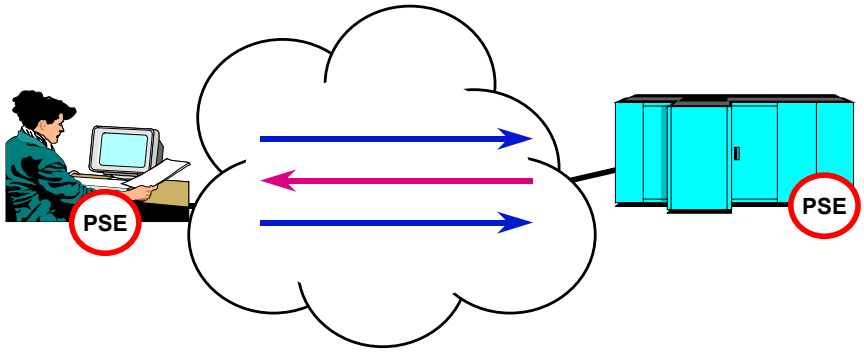


Figure 9: Overview – 3-way Logon

So that SAPgui can access the certificate, the user must prove that he or she is the rightful owner of the PSE. This is done using the *Secure Single SignOn* program and by entering the corresponding password as displayed in Step 1 in *Figure 10*.

You can log on to the R/3 Application Server by using SECUDE as follows: First open the PSE by entering the password using *PSE MANAGEMENT* (local logon) and then start SAPgui. SAPgui accesses the digital identification of the user for the logon using the SNC interface and SECUDE library linked to it. The following discusses the logon procedure in more detail.

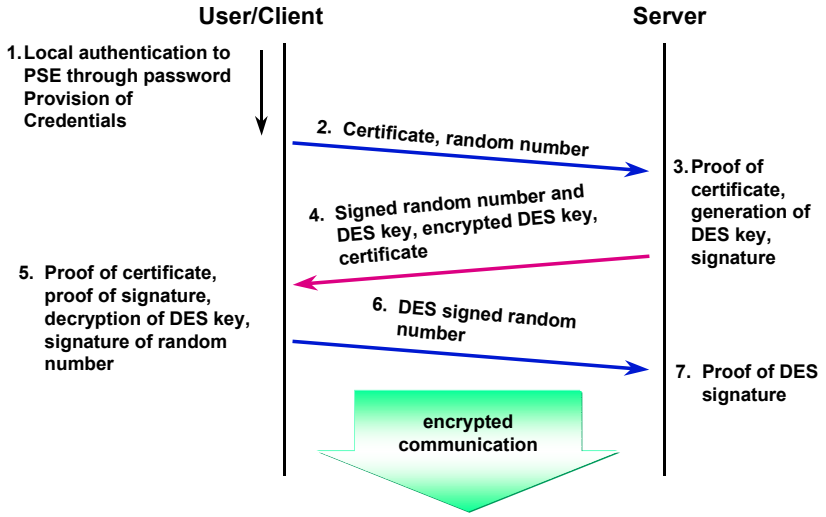


Figure 10: Details – 3-way Logon

If the user is logged on locally using PSE MANAGEMENT, SAPgui can be started. SAPgui receives the user-specific information for authentication via GSS-API (the certificate and a random number). It sends this information to the R/3 application server – Step 2. The server checks the certificate for validity. If the certificate is valid, it generates a symmetric key that is used to secure communication between it and the user after logon. The server encrypts this key using the user's public asymmetric key, as shown in Step 3. The public asymmetric key is contained in the certificate. Only the owner of the matching private key

can decrypt the contents. The server adds its certificate and creates a signature using the random number and symmetric key. This packet is sent back to the user – Step 4. The user checks the server's certificate that he has received – Step 5. If it is valid, he decrypts the symmetric key using his private asymmetric key. Using the symmetric key and the server's certificate, he is now able to check the server's signature. This way he can be sure that he is connected to the correct server. No other server or potential intruder would have been able to create the signature.

The R/3 application server still requires the user's authentication. The symmetric key is used for this. Using this key, the user signs the random number from Step 1 and sends the result to the server – Step 6. The server now checks the digital signature – Step 7. If the signature is verified, then the server is sure that it is communicating with the rightful owner of the certificate. The server now logs the user on to R/3 using the name that is found in the user's certificate.

The logon procedure creates a symmetric key for the two participants, which is used for this session only. With this key all communication that takes place between the user and the R/3 application server is encrypted.

The advantage of what seems like a complicated procedure is that no third authority is required. All participants in the SECUDE security infrastructure can locally access all of the information necessary for identifying themselves to other participants.

### 3.1.3 CPU Time with Encryption

The symmetric key created at logon, which is used to create the signature for user authentication, is also used for encrypting the data sent by the two participants.

A symmetric procedure is used for encryption. Symmetric means that both encrypting and decrypting are carried out using the same key. In comparison to asymmetric procedures such as RSA, symmetric procedures are approximately 75 times faster when encrypting a 1024-bit key, and 1000 times faster for decrypting. This makes symmetric procedures more suitable for encrypting and decrypting large data sets.

For the user, the CPU time and the time required for encrypting and decrypting is hardly noticeable compared to the time needed for transmitting the data over a network. At the R/3 application server end, where many users work simultaneously, encryption takes up additional CPU time, resulting in an approx. 5-10 percentage increase in the CPU computation time (depending on the tasks being executed). To accommodate for this, it is possible to distribute the load over additional R/3 application servers.

## 3.2 Secure Store and Forward (SSF)

*Secure Store and Forward*, in brief *SSF*, is used to sign and encrypt data that is required for later processing in this form. R/3, from version 4.5A, will be automatically provided with this interface.

The SSF functions are made available to SAP application programmers via an ABAP interface. The modules PPPI or QM already contain access to SSF; this permits, for example, the compliance with FDA guidelines. The signed data are saved in the standard format PKCS#7.

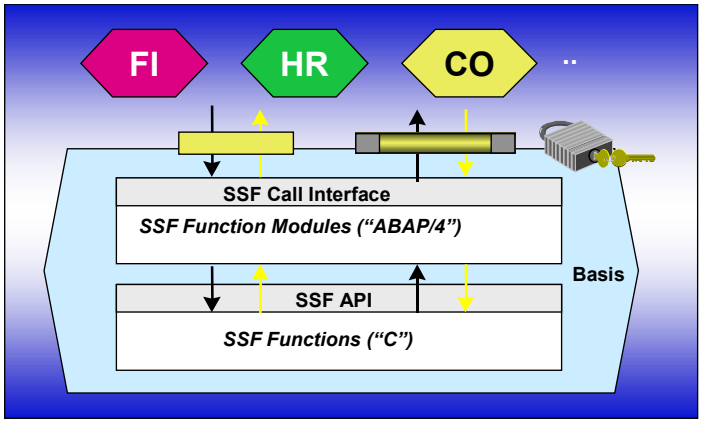


Figure 11: SSF Structure

The SSF interface is available on both the application server and presentation server. Usually, however, it will only have to be activated on the presentation server. It then gives users the possibility to make digital signatures and optionally to encrypt. For this purpose smartcards can also be used on which the users' keys are stored.

The ABAP commands, in turn, are converted to SECUDE library functions. To enable use of the SSF ABAP commands this interface must be activated.

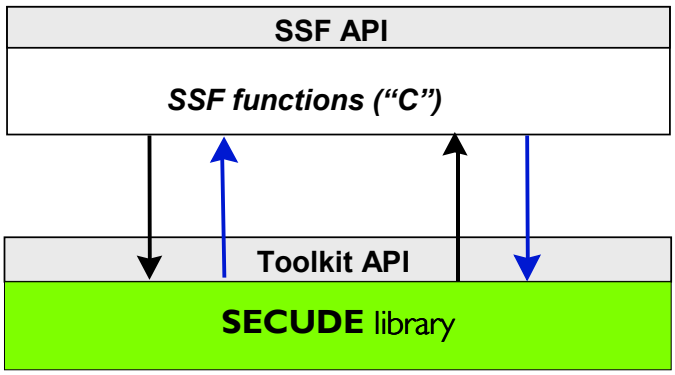


Figure 12: SSF access to SECUDE library

## 3.3 SECUDE CA MANAGEMENT

To ensure secure operations with SECUDE, each participant in the security infrastructure requires digital identification. This identification is

the admission ticket to the SECUDE secured R/3 system. When SNC is installed, the participants in the security infrastructure include application servers and printers (SAPlpd) which have to be used securely. For SSF only the users, and in certain special cases the application servers, need these tickets.

As an option to SECUDE for R/3 a program (*SECUDE CA MANAGEMENT*) can be supplied with which all tasks and requirements allotted to a certification authority can be implemented.

## 3.4 SECUDE PSE MANAGEMENT

To use the SAP client with the digital identification (PSE), SECUDE PSE MANAGEMENT must be used by the client.

SECUDE offers optionally the PSE either on a smartcard or as an encrypted file to be stored on the hard disc. Depending on the version of the PSE selected, (smartcard or file), various degrees of difficulty are encountered when trying to gain unauthorized access to sensitive data. With a file PSE it might even be possible that the legitimate owner of the PSE does not notice its loss. An intruder who succeeds in spying out the password and copying the file PSE has access to all necessary information. The advantage of the smartcard PSE is that the owner will, in normal cases (not while he is on holiday or similar), quickly notice the loss of his card. A disadvantage, however, is that a special reading device for the card is required.

The administration of the PSE is undertaken by SECUDE PSE MANAGEMENT.

# 4 Installing SECUDE for R/3 SNC

SECUDE for R/3 SNC requires an R/3 system as of at least version 3.1G. This system is equipped with the necessary interfaces and enhancements to use security functions, like those provided by SECUDE.

Existing PSEs for each participant (application servers, SAPlpd and users) are required for operating SECUDE for R/3. For generating and using PSEs, please refer to the manuals PSE MANAGEMENT and CA MANAGEMENT.

This chapter explains how to install SECUDE for R/3 SNC on the application server, SAPgui and SAPlpd. All the required parameters and installation steps are explained in detail. The SECUDE team is available for questions at any time.

## 4.1 Preparing the Installation

SECUDE for R/3 is supplied on a CD. This CD contains all programs and libraries that are required for the installation.

The installation procedure varies for the different operating systems. You can install SECUDE for Windows 95 (at least SR1), Windows 98 and Windows NT (from Version 4.0, at least service pack 3) directly from the CD. For the different UNIX operating systems first the appropriate version must be copied into a directory from which the installation is to be carried out.

**Information about the CD**

The CD has the following contents:

```
\Adobe
\secSAP<ver>
    \aix-<ver>
    \doc
    \hp_ux<ver>
    \irix-<ver>
    \linux-<ver>
    \ntalpha
    \osf1-<ver>
    \PSE Management-<ver>
    \sinix-<ver>
    \solaris-<ver>
    \sunos-<ver>
    \winnt
```

Depending on the version of secSAP various operating systems and versions are supported. <ver> stands here for the version of the operating system.

N.B.: Most operating systems are upwardly compatible so that, for example, the AIX version 4.1.3 also runs on 4.2.

The directory *doc* contains the PDF-files of this manual and the manuals for PSE MANAGEMENT and CA MANAGEMENT, as well as Version 3.01 of Adobe Acrobat Reader for installation under Windows 95, Windows 98 or Windows NT.

The directories of the UNIX operating systems contain the SECUDE library and the SECUDE program. To install these the files have to be copied from the CD.

The directory *winnt* contains the library *secude.dll* and the executable program *secude.exe*, as well as the installation file for "SECUDE 2.0 for R/3 for NT server (from version 2.0)". These files have to be copied from the CD.

The directory *PSE Management* contains the software for Windows 95, Windows 98 and Windows NT.

## 4.2 Installing SECUDE for R/3 on a UNIX server

The R/3 application server requires access to the SECUDE library. In UNIX operating systems, the library is called *libsecude* and has a file extension such as *".o"* or *".so",* depending on the operating system. The *secude* program also needs to be installed.

**Storage and Time required for Installation**

Roughly 2 megabytes of disk space are required for installing SECUDE 2.0 for R/3, depending on the operating system and version. The installation takes approx. 30 minutes; or longer, depending on the administrators' knowledge of the operating systems.

The following lists the steps required for the installation:

**Installation Steps**

(1) Install SECUDE 2.0 for R/3 (library and program) from CD
(2) Set the search paths and environment variables (HOME, SECUDE_ETC, and SNC_LIB)
(3) Install the application server's PSE
(4) Make the PSE available to the R/3 application server
(5) Set the instance profile parameters for the application server
(6) Restart the R/3 application server
(7) View the log file
(8) Enter the external user names in the user master record

The installation steps will now be explained in detail.

**(1) Install SECUDE for R/3**

SECUDE for R/3 can be installed in any directory. The description of the installation below uses the path /usr/local/secude. It is important that the application server can access the SECUDE library during runtime.

The directory for the operating system under which the installation is to be executed, is selected, for example, \solaris 5.5. Two files can be found in it.

The installation directory must be in a path that the application server can access during runtime, for example, /usr/local/secude. If the path /usr/local/secude does not yet exist, it, together with the subdirectory *etc,* must be created using the UNIX command ***mkdir***.

The library *libsecude* is used by both the *secude* program and the R/3 application server. In UNIX operating systems it is necessary to set the search path for runtime libraries such as *libsecude* (see below).

The command *chatr* (change attribute) is required to use the *secude* program in the HP-UX operating system with shared libraries.

```
chatr +s enable secude
```

This command activates the loading of the shared libraries for *secude*. The *chatr* command without parameters, only with a program name, displays the current status.

**(2) Set Search Paths and Environment Variables**

The search path for runtime libraries for the operating systems DEC and SunOS is set using *setenv LD_LIBRARY_PATH*.

Example

```
setenv LD_LIBRARY_PATH /usr/local/secude/lib:$LD_LIBRARY_PATH
```

AIX searches the runtime library using *LIBPATH*.

Example `setenv LIBPATH /usr/local/secude/lib:$LIB_PATH`

HP-UX uses the variable *SHLIB_PATH*.

Example `set SHLIB_PATH /usr/local/secude/lib:$SHLIB_PATH`

All other systems search the runtime library using the *PATH* variable. This means */usr/local/secude/lib* must be added to *PATH*.

Example `set PATH ($PATH /usr/local/secude/lib)`

The path setting may differ from the above example and have another syntax. The UNIX *man pages* of the operating system should be consulted to find this out for the UNIX shell in use.

The environment variables HOME and SECUDE_ETC must be set. The HOME variable has to be set to the home directory of the R/3 user under which the application server is started. SECUDE_ETC is required for the SECUDE configuration files.

Example
```
setenv HOME /home/sapr3-31g
setenv SECUDE_ETC /usr/local/secude/etc
export SECUDE_ETC=/usr/local/secude/etc
```

The syntax for setting the environment variables may differ from the example. *Export* is possible instead of *setenv*. The three environment variables should be written with upper case letters. The values for the variables must correspond exactly to the path names.

**(3) Install the application server's PSE**

The PSE for the application server is created by SECUDE PSE MANAGEMENT. The PSE contains the security relevant information needed by the application server for secure operations. This includes the certificate, the private key and the root certificate.

During operations, the application server must access this information to check the certificates of users logging on. For this it requires the root certificate and the certificates of the *Forward Certification Path*, if available (see *Figure 7*). The application server needs access to its own private key to identify itself to the user logging on. All of this information is stored in the PSE.

The PSE should be located in the home directory of the R/3 administrator whose user ID is used to start the server (for example, /home/sapr3-31g or /home/sapadm). A software PSE can be one of two versions, either a binary file (as of SECUDE 5.1) or a file structure. In UNIX, the file names are case sensitive. The file structure of a PSE can contain the following files:

```
PSE file structure:
     Cert.sf
     PKList.sf
     FCPath.sf
     PKRoot.sf
     SKnew.sf
     Toc.sf
     pse.pw
```

As of the library version 5.1 (CA MANAGEMENT 1.3 and later) PSEs are represented as files. A PSE file also contains the listed objects, handling a single file is, however, much easier.

The entire PSE has to be copied. For this, UNIX creates a *tar* archive of the PSE and decompresses it in the home directory. The file or file structure can be given any name as required – for example, *pse* or *sapr3.pse*.

The contents (files) of a PSE are coded system-independently. Thus a PSE can be created in Windows 95 or Windows NT and used in UNIX.

**(4) Make the PSE Available to the R/3 Application Server**

When the PSE for the application server is at its designated location, this must be opened to operate R/3 with SECUDE. Since the PSE is protected by a password, an additional program is required which can access and check the password. For UNIX, this is the *secude* program. This program has two parameters: a command parameter and a value parameter. The value parameter is the PSE to be opened. The program checks the password and, if successful, allows the SECUDE library linked to the SAP R/3 application server access to the information available in the PSE.

If SECUDE for R/3 has been installed as described above, then *secude* has been added to the search path. The following command can be entered

in a UNIX shell in the home directory of the SAP R/3 administrator under whose account the application server has been started:

```
58 /home/sapr3-31g>
secude seclogin -p /home/sapr3-31g/sapr3.pse
```

The command is entered in the shell in a single line and confirmed with *Return*. In the above example, the directory /home/sapr3-31g is searched for the PSE *sapr3.pse*. If this exists, the password (called PIN here) for the PSE is requested:

```
Enter the PIN for pse:

CADIR   :
PSENAME : /home/sapr3-31g/sapr3.pse
DNAME   : CN=sapserv, O=SECUDE GmbH, C=DE

Credentials added
59 /home/sapr3-31g>
```

The *secude* program creates a file with *credentials*, which is stored in the home directory. The filename is *cred*. Using this file, the SECUDE library, which the R/3 application server now loads, is able to access the PSE. The contents of the *credential* file can be displayed using the following command:

```
60 /home/sapr3-31g>secude seclogin -l
CN=sapserv, O=SECUDE GmbH, C=DE /home/sapr3-1g/sapr3.pse
61 /home/sapr3-31g>
```

Help on the parameters of *secude seclogin* is availabe using the following command:

```
62 /home/sapr3-31g>secude seclogin -h
sec_login    installs credentials (name path and pin)
             for further use of your PSE

usage:
sec_login [-hVWld1] [-p<pse>] [-c <cadir>]
with:
-p <psename>  PSE name (default:
                 environment variable PSE or pse)
-c <cadir>    Name of CA-directory (default: ca)
-h            this help text
-l            list all credentials ( -p/-c not needed )
-d            delete credentials
-1            add as default credentials
-R            set CRL checking
-D            set Directory access
-v            Verbose
-V            Verbose
-W            Grand Verbose (for testing only)
```

The *credentials* file remains in the home directory of the R/3 administrator until it is removed by calling *secude*.

```
63 /home/sapr3-31g>
     secude seclogin -p /home/sapr3-31g/sapr3.pse -d

Credentials deleted
```

Removing *credentials* uses almost the identical command as when it is entered. The difference is that there is a *-d* at the end of the command line. This parameter indicates that the *credentials* file should be removed.

*Credentials* need to be deleted only if the application server receives a new PSE or the password for the PSE is changed. If, for example, the PSE has to be replaced, it must be ensured that the application server cannot accept any new links until after the new PSE has been established, and that there are currently no active links. The *credentials* file and the old PSE can then be deleted. Proceed as described under (6) and (7).

**(5) Set the Instance Profile Parameters for the Application Server 3.1 and 4.0, 4.5**

The following SAP profile parameters for the instance profiles have been added for operating the R/3 application server using SECUDE for R/3.

The current SAP profile parameters can be found from the relevant R/3 Release (3.1G, 3.1H, 4.0A, 4.5) by using the R/3 transaction RZ11.

| | |
|---|---|
| snc/enable | Activating or deactivating external authentication. |
| snc/user_maint | Activates the field in user maintenance (SU01) for the names required for logging on with SECUDE. |
| snc/permit_insecure_gui | Allows insecure logons, although SECUDE has been activated. |
| snc/permit_insecure_comm | Allows insecure CPIC communication, although SECUDE has been activated. |
| snc/permit_common_name | Allows the identity of the application server to be transmitted to external programs. |
| snc/data_protection/min | Minimum requirement for protecting incoming data. |
| snc/data_protection/max | Limits the protection of incoming data. |
| snc/data_protection/use | Protecting outgoing data. |
| snc/gssapi_lib | Path and filename of the SECUDE library. |
| snc/identity/as | Name of the R/3 application server as known to SECUDE (Distinguished Name). |

### New as from Version 4.0:

| | |
|---|---|
| snc/r3int_rfc_secure | Use of SNC to initiate internal RFC links |
| snc/r3int_rfc_qop | Quality of Protection (QoP) for internal RFCs secured with SNC |
| snc/accept_insecure_cpic | Accepts insecure incoming CPIC links on an application server secured with SNC |
| snc/accept_insecure_gui | Accepts logons from SAPguis that are not secured with SNC on an SNC secured application server |
| snc/accept_insecure_r3int_rfc | Accepts insecure internal RFC links on a SNC secured application server |
| snc/accept_insecure_rfc | Accepts insecure incoming RFC links on an SNC secured application server |
| snc/accept_insecure_start | Allows the start of an insecure program when SNC is activated |

### New as from Version 4.5:

| | |
|---|---|
| snc/force_login_screen | Shows the logon display for all SNC secured logons |

The profile parameters are discussed in detail below. A value of "0" indicates that the parameter is deactivated, a value of "1" activates the parameter.

snc/enable default = 0 (value set [0,1])

When this parameter is set to "1", the external security system SECUDE is activated. SECUDE is used for authentication and for protecting data transfer. All insecure communication requests that come from SAPguis are rejected after activating. By setting the additional profile parameter snc/permit_insecure_gui, insecure logons are allowed, permitting "mixed" operation.

Example: snc/enable=1

snc/user_maint default = 0 (value set [0,1])

If you enter a value in this parameter in the profile, the *External user name* field then appears in user maintenance (SU01, change user). For each user who uses SECUDE to authenticate at logon, you have to enter the name in this field as it is known to SECUDE. The name can be found in the *Owner* field of the user's certificate.

Example snc/user_maint=1

snc/permit_insecure_gui default = 0 (value set [0,1])

Using this parameter conventional data transfer (insecure data transfer) is allowed from SAPgui while using SECUDE. This mixture is required during installation and possibly during a transitional period.

Example: snc/permit_insecure_gui=1

snc/permit_insecure_comm default = 0 (value set [0,1])

By setting this parameter to "1" insecure CPIC communication is allowed.

Example snc/permit_insecure_comm=1

snc/permit_common_name default = 0 (value set [0,1])

For secure communications, an external program needs its own identity in order to be authenticated by the application server. If the external program has not been given a name, and this parameter flag has been set, the name of the application server will be used.

Example: snc/permit_common_name=1

snc/data_protection/min default = 3 (value set [1, 2, 3])

Using this parameter, the minimum requirement for protecting incoming data (in 3.1G, from SAPgui only) is determined - that is, when links are set up to the application server. If the incoming data does not fulfill this minimum requirement, the server terminates the link. If the incoming data fulfills the minimum requirement, the server uses the same quality for protecting the data when transmitting the response. The three possible protection levels are:

1. Secure logon, no additional protection of data
2. Secure logon and data provided with integrity protection.
3. Secure logon, data provided with integrity protection and encryption

Example: snc/data_protection/min=2

snc/data_protection/max default = 3 (value set [1, 2, 3])

Using this parameter the available measures for protecting incoming data are limited. For security reasons, this parameter should not be used, or should always be set to 3. If confidential data (in the broader sense) are not being processed, or large datasets are being moved between an application server and frontends, and performance matters, prevention of CPU time wastage for data encryption may usually be preferred.

Example: snc/data_protection/max=2

snc/data_protection/use default = 3 (value set [1, 2, 3])

This parameter does not affect the link to SAPgui. Using this parameter protection for outgoing data is determined; that is, when the application server sets up the link (for example, when starting external programs using CPIC).

Example: snc/data_protection/use=3

snc/gssapi_lib Platform-dependent

This parameter tells the R/3 application server the path and file name where the SECUDE library is located. The specified library is dynamically loaded when the R/3 components are run and included in the program flow.

Example: snc/gssapi_lib=/usr/local/lib/libsecude.so
(Operating systems: DEC, SINIX, SunOS)
snc/gssapi_lib=/usr/local/lib/libsecude.o
(Operating systems: AIX)
snc/gssapi_lib=/usr/local/lib/libsecude.sl
(Operating system: HP-UX)
snc/gssapi_lib=c:\programme\secude\secude.dll
(Operating system: Windows NT)

snc/identity/as No default exists

This parameter tells the application server its name as it is known to SECUDE. A PSE (personal security environment) has to be set up or stored for this name. The PSE must be located where the application server has continual access to it, during booting as well as during runtime. When entering the name, the prefix, which specifies the name format or syntax, must be included. For SECUDE there is currently only one valid name format, called a *Distinguished Name*. The prefix is: "**p:**"

Example: snc/identity/as=p:CN=sapserv, O=SECUDE GmbH, C=DE

## New as fromVersion 4.0:

snc/r3int_rfc_secure default = 0 (value set 0, 1)

This parameter determines whether RFCs to internal destinations in the same R/3 system should be secured with SNC. As performance problems may arise it should be resolved whether it is necessary to secure with SNC RFCs to internal destinations. An equal degree of security can be achieved with other methods (see the *SNC User's Guide* from SAP, chapter *2.6: Recommendations*).

snc/r3int_rfc_qop default = 8 (value set 1, 2, 3, 8, 9)

This parameter determines the quality of protection (QoP) to be used for internal RFCs when SNC is set for these (**snc/r3int_rfc_secure = 1**).

The valid parameters are:

1 secure authentication only
2 protection of integrity of data
3 protection of confidentiality of data
8 the value from snc/data_protection/use is used
9 the value from snc/data_protection/max is used

snc/accept_insecure_cpic default = 0 (value set 0, 1, U)

When SNC is activated (**snc/enable=1**), the R/3 application server automatically rejects all incoming CPIC links from external C programs or other R/3 systems that are not secured with SNC. This parameter is set to overwrite the standard setting to accept insecure

links (e.g. links from old CPIC programs or from R/3 systems not secured with SNC).
The valid parameters are:
0 insecure links rejected
1 insecure links accepted
U insecure links are accepted when the appropriate identification is set in the user master record.

snc/accept_insecure_gui default = 0 (value set 0, 1, U)

When SNC is activated (**snc/enable=1**), the R/3 application server automatically rejects all link requests from SAPguis that are not secured with SNC. This parameter is set (see valid entries, formats) to overwrite the standard setting and to accept insecure link requests (e.g. links from old SAPguis not able to to work with SNC). When insecure link requests are accepted it is at the user's discretion whether he makes a secure logon, thus protecting his data during transmission.

snc/accept_insecure_r3int_rfc default = 1 (value set 0, 1)

This parameter is set to accept insecure RFC links proceeding from internal R/3 systems and having internal destinations (see transaction SM59), even when insecure RFC links are deactivated (**snc/accept_insecure_rfc** = 0). This parameter is only effective when **snc/accept_insecure_rfc** = 0; Otherwise all RFC links (internal and external) are accepted notwithstanding the setting of this parameter.

snc/accept_insecure_rfc default = 0 (value set 0, 1, U)

When SNC is activated (**snc/enable=1**), the R/3 application server automatically rejects all incoming RFC links from external C programs or other R/3 systems that are not secured with SNC. This parameter (see valid entries, formats) is set to overwrite the standard setting and to accept insecure RFC links (e.g. links from old RFC programs or systems that are not secured with SNC). This parameter applies to all RFC links, also internal ones whereas the parameter **snc/accept_insecure_r3int_rfc** applies solely to internal RFCs.

snc/permit_insecure_start default = 0 (value set 0, 1)

When SNC is activated (**snc/enable=1**), it is a Gateway standard to start no programs that are not secured with SNC. This parameter (see valid entries, formats) is set to overwrite the standard configuration and to allow the Gateway to start insecure programs.

## New as from Version 4.5:

snc/force_login_screen default = 0 (value set 0, 1)

When this parameter is set at "1", the logon display is shown for every SNC secured logon. Otherwise the system shows the logon display only when this is required. In this case no display is shown when the SNC name or the external name from the logon can be unambiguously allotted to an R/3 user. (The R/3 user with the given SNC name or external name resides in R/3 one time only in a single customer.)
The valid parameters are:
0 The logon display is shown only when required
1 The logon display is always shown

Here is an example of the profile parameters that activate SECUDE for R/3.

```
…
#-------------------------------------------------
# snc(general)
#-------------------------------------------------
snc/enable                    =1
snc/user_maint                =1
snc/permit_insecure_gui       =1
snc/permit_insecure_comm      =1
snc/permit_common_name        =1
snc/data_protection/min       =2
snc/data_protection/max       =3
snc/data_protection/use       =3
snc/accept_insecure_gui       =1
snc/accept_insecure_cpic      =1
snc/accept_insecure_rfc       =1
snc/accept_insecure_r3int_rfc=1
snc/r3int_rfc_secure          =0
snc/r3int_rfc_qop             =3
snc/permit_insecure_start     =1

#-------------------------------------------------
# snc(secude)
#-------------------------------------------------
snc/gssapi_lib=/usr/local/secude/lib/libsecude.so
snc/identity/as=p:CN=sapserv, O=SECUDE GmbH, C=DE
```

**(6) Restart the R/3 Application Server**

After installing the library, setting the environment variables, and logging on using *seclogin* for *secude,* the application server is restarted. This process is described in the SAP R/3 documentation.

**(7) View the Log File**

The log file should now show that the SECUDE library has been loaded successfully.

```
...
X  Fri Mar. 14 08:14:09 1997
X  <ES> client 0 initializing ....
X  ES initialized.
N
N  Fri Mar 14 08:14:11 1997
N  SncInit():   found snc/data_protection/min=3,
                using 3(Privacy Level)
N  SncInit(): missing snc/data_protection/max= ,
                using 3 (Privacy Level)
N  SncInit(): missing snc/data_protection/use= ,
                using 3 (Privacy Level)
N  SncInit(): found snc/gssapi_lib=
                /usr/local/secude/lib/libsecude.so
N    File "/usr/local/secude/lib/libsecude.so"
                dynamically loaded as
                GSS-API v2 library.
N    The internal Adapter for the loaded
                GSS-API mechanism identifies as:
N    Internal SNC adapter (Rev 1.0) to
                SECUDE 5/GSS-API v2
N SncInit():    found snc/identity/as=
                p: CN=sapserv, O=SECUDE GmbH, C=DE
N SncInit(): Accepting Credentials available,
                lifetime=Indefinite
N SncInit(): Initiating Credentials available,
                lifetime=Indefinite
M  SNC (Secure Network Communications) enabled
M  snc/permit_insecure_gui is set
...
```

When the above messages appear in the log file (../work/dev_w0), the installation has been carried out successfully. You can also display this log file (syslog) from the R/3 System using the transaction ST11.

### (8) Entering the External User Name in the User Master Record

To allow a user to log on securely, a new field, the "External user name," has been introduced into the user master record.

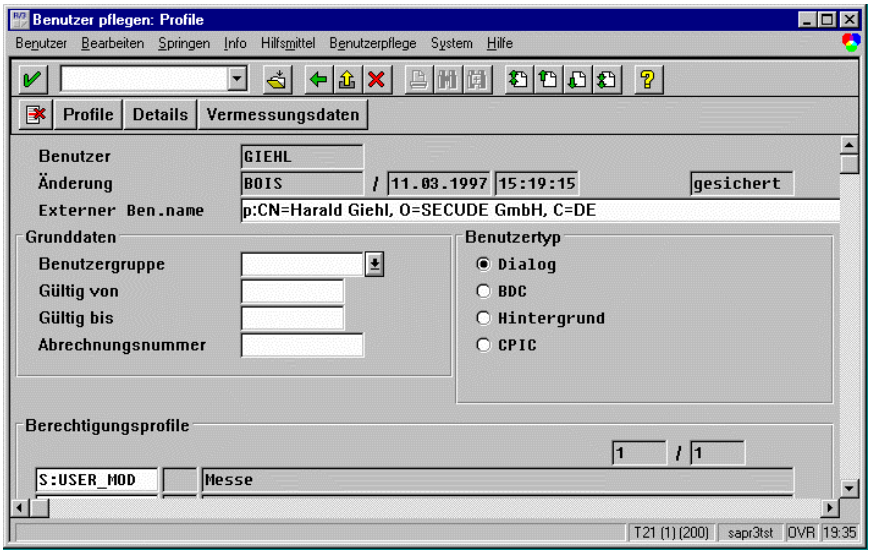


Figure 13: User maintenance in R/3

In this field, the *External user name* for each user as found in his or her certificate is maintained. In *Figure 13*, the *External user name* field is shown. The complete name, also called the *Distinguished Name*, including the prefix "p:" is entered; the blanks and delimiters must be included. After successful authentication (see chapter *3.1.2 Logon Procedure and Communication*), the user privileges can be selected using these names.

## 4.3 Installing SECUDE for SAP R/3 NT-Server

The installation for Windows NT differs from the installation described in *4.2 Installing SECUDE for R/3 on a UNIX server* in the items *(1) – (4)* .

**(1) Install SECUDE for R/3**

The program *SECUDE20forR/3NTServer.exe,* to be found on the CD, should be started. A target directory is requested in which the SECUDE libraries and the SECUDE program can be unpacked. No environment variables or registry entries are set.

**(2) Set search paths and environment variables**

No search paths or environment variables are set.

**(3) Install application server's PSE**

The PSE for the application server is created by SECUDE PSE MANAGEMENT. The PSE contains the security relevant information needed by the application server for secure operations. This includes the certificate, the private key and the root certificate.

During operations, the application server must access this information to check the certificates of users logging on. For this it requires the root certificate and the certificates of the *Forward Certification Path*, if available (see *Figure 7: PSE – Personal Security Environment*). The application server needs access to its own private key to identify itself to the user logging on. All of this information is stored in the PSE.

The application server's PSE should be copied into a directory to which the user under whose account the application server is running has full rights. No other users should have rights to this directory.

The environment variable *CREDDIR* should be set under *Start/Settings/Control Panel/System*. The tab *Environment* is selected and the value of the variable to the directory path which contains the application server's PSE set.

**(4) Make the PSE Available to the R/3 Application Server**

Log on with the account under which the R/3 application server is running.

If the PSE for the application server is at its designated location, this must be opened to operate R/3 with SECUDE. Since the PSE is protected by a password, an additional program is required which can access and check the password. This is the *secude.exe* program. This program is started and the request

```
secude>
```

is shown.

A command and a parameter with a value are then entered. The command is *seclogin* and the value of the parameter is the the PSE to be opened. The program checks the password and, if successful, allows the SECUDE library linked to the R/3 application server access to the information available in the PSE.

```
secude> seclogin –p c:\secude\sapr3.pse
```

The command is entered in a single line and confirmed with *Return*. In the above example the PSE *sapr3.pse* is searched for in the directory c:\secude. If this exists, the password (called PIN here) for the PSE is requested:

```
Enter PIN for sapr3.pse:

CADIR   :
PSENAME : c:\secude\sapr3.pse
DNAME   : CN=sapserv, O=SECUDE GmbH, C=DE

Credentials added for owner <username>
secude>
```

The *secude.exe* program creates a file with *credentials*, which is stored in the directory determined by the environment variable *CREDDIR*. The filename is *cred*. Using this file, the SECUDE library, which the R/3 application server now loads, is able to access the PSE. The contents of the *credential* file can be displayed using the following command:

```
secude> seclogin -l
CN=sapserv, O=SECUDE GmbH, C=DE /home/sapr3-
31g/sapr3.pse secude
```

Help on the parameters of *seclogin* is availabe using the following command:

```
secude> seclogin -h
sec_login    installs credentials (name path and pin)
             for further use of your PSE

usage:
sec_login [-hVWld1RD] [-p<pse>] [-c <cadir>]
with:
-p <psename>  PSE name (default:
                 environment variable PSE or pse)
–c <cadir>    Name of CA-directory (default: ca)
-h            this help text
-l            list all credentials ( -p/-c need needed )
-d            delete credentials
-R            set CRL checking
–D            set Directory access
-v            Verbose
-V            Verbose
-W            Grand Verbose (for testing only)
```

The *credentials* file remains in existence until it is removed by calling *secude.exe*.

```
secude> seclogin -p \secude\sapr3.pse -d

Credentials deleted
```

Removing *credentials* uses almost the identical command as when it is entered. The difference is that there is a *-d* at the end of the command line. This parameter indicates that the *credentials* file should be removed.

*Credentials* need to be deleted only if the application server receives a new PSE or the password for the PSE is changed. If, for example, the PSE has to be replaced , it must be ensured that the application server cannot accept any new links until after the new PSE has been established, and that there are currently no active links. The *credentials* file and the old PSE can then be deleted. The *credentials* file can then be created anew as described above.

The environment variable *CREDDIR* has to be added to the registration database for the Windows NT application server as shown below:

- HKEY_LOCAL_MACHINE\\SOFTWARE\\SAP\\<*SYSTEM-NAME*>\\ENVIRONMENT

  **Stringvalue** with **Value**: CREDDIR <directory path which contains the application server's PSE>

SAP R/3 will set this environment variable for the application server. The computer has to be rebooted to activate this registration database entry. It is now best to set the instance profile parameters as described under (5) in Chapter *4.2 Installing SECUDE for R/3 on a UNIX server*. The application server and database server should then be stopped and the computer restarted. The procedure is continued from step 7.

## 4.4 Activate Revocation Lists

If certificates are compromised (e.g. because the smartcard got lost or bcause it is suspicted that someone used a false name) these certificates have to be revoked. For this purpose the certification authority generates a so-called *Revocation List*.

To prevent users from logging on to the R/3 with a revoked certificate, the application server can optionally check user certificates against revocation lists. For this the parameter *–R* is used when generating the credentials file for the application server (in step 4 of the installation).

**Attention**: Activate the checking against revocation lists only if you can make sure that the revocation list of the application server is updated regularly. A missing or obsolete revocation list makes the user authentication fail. The application server needs revocation lists from all certification authorities in its certification path.

# 4.5 Installing SECUDE for SAP R/3 Client

## 4.5.1 Preparing the Installation

The program *SECUDE20forR3Client.exe* to be found on the CD should be started. A target directory is requested in which the SECUDE Client software can be unpacked. No environment variables or database registration entries need to be made. The target directory should be a network directory so that the actual client installation can be started from it.

Under Windows 95 and Windows NT (from version 4.0 upwards) the program is installed in the directory *SECUDE*, which is to be found in the Windows start menu under *Program Files*.

The installation reads the SAPgui program icon from the SAP start menu directory *SAP Frontend 3.1G* (depending on the SAP release). A new SAPgui program icon is then initialized in the SECUDE start menu. Its parameters must be defined in the file *SAPGui.ini* in the setup source directory.

The files *SAPgui.ini* and *Saphelpd.wri* are in the directory selected for unpacking. *SAPgui.ini* has to be adapted before the installation. *Saphelpd.wri* contains detailed information for the processing of *SAPgui.ini*. An example of a *SAPgui.ini* file is in the appendix.

All presettings of *SAPgui.ini* are to be replaced; there is no automatic search for components. If the presettings are not replaced, the following error message is displayed:

*No valid settings in the SAP GUI INI file.*

The file *SAPGui.ini* can be roughly divided into two areas: the first is concerned with the configuration using *sapgui.exe* and is introduced with the field [SAPGUI]; the second is provided for the configuration of *saplogon.exe*.

The following settings are to be entered in the *SAPgui.ini* file*:*

### *For the field [SAPGUI] (see example in 10.1, 10.2)*

**Credentials**

The value *'creddir'* can contain the path to the SECUDE credentials of all users, it is written in the environment variable CREDDIR. If it is not to be used, *'creddir'* must be either empty or not available.

**SSF Library Path-Variable**

The value*'ssflib'* can contain the name of the SSF-DLL, it is written in the environment variable SSF_LIBRARY_PATH. If it is not to be used, *'ssflib'* must be either empty or not available.

**Path-Variable**

When *'setpath'* has an arbitrary value the environment variable PATH is extended by the SECUDE path. If it is not to be extended, *'setpath'* must be either empty or not available.

**SNC Lib Variable**

When *'setsnclib'* has an arbitrary value the environment variable SNC_LIB is set to SECUDE-DLL. If it is not to be set, *'setsnclib'*must be empty or not available.

**SAP-Path**

The value *'sappath'* must contain the complete path without program names of the local SAPgui program (the program name is configured separately). Only when *'sappath'* has a value, are the following SAPgui entries evaluated. To configure, for example, SECUDE for R/3 only for SAPlogon '*sappath'* is not needed.

**Start Menu**

The value *'folder'* has to be the local SAP start menu directory. Thus for different installations of SAP GUIs different SECUDE setups have to be configured.

**SAP Icon**

The value *'icon'* has to be the local icon for the SAP GUI program in the SAP start menu directory.

**SECUDE Icon**

The value *'newicon'* determines the icon of the new SAP GUI program link in the SECUDE start menu directory. Only characters admitted for Windows program names may be used (not: /, \, :, ?, <, >, or |).

**Executable**

The value *'exe'* must contain the program names of the local SAPgui program (without parameters or options).

**Parameter**

The value *'param'* has to contain the standard parameters of the local SAP GUI program.

**Server Name**

The value *'srvdname'* has to contain the X.500 Distinguished Name of the SAP Server.

### *For the field [Itemx] (see example in 7.2)*

The file SAPGui.ini offers, moreover, the opportunity to create entries, with which *saplogon* entries can be automatically created. This is, however, optional.

The entries have to be carried out in the form [Item*x*], where *x* corresponds to a number. The first entry must begin with one, i.e.

[Item1], any number of entries can follow. The numbering must, however, be consecutive, i.e. without gaps.

The installation routine checks only whether the values listed below are available and, in the case of the system number, correspond to a number. If the check discovers an error, an error message with the item number and the parameter is shown.

If an entry is made in the form [Item*x*], the following settings must be carried out:

**Servername**

The value '*server*' contains the DNS name or the IP address of the SAP server.

**Database**

The value 'database' contains the database number of the SAP server being used.

**Description**

The value 'description' allows for a description for the SAP server to be given.

**Sncname**

The value 'sncname' must contain the Secure Network Communications Name of the SAP server.

**SncChoice (optional)**

The optional value 'sncchoice' contains the quality of protection of the SNC link. If this parameter is not set in the SAPGui.ini the value is automatically set at 9.

| Value | Meaning |
|---|---|
| 1 | Authentification |
| 2 | Integrity |
| 3 | Encryption |
| 9 | Maximum available |

## 4.5.2 Carrying out the Installation

The software has to be installed directly at the client computer. To carry out the installation, administrator authorizations are necessary.

The following example of an installation uses the *English* language. In the startup window of the installation program you are requested to exit all other active applications. This is necessary; otherwise the installation program may not be able to carry out all of the activities that are required for a faultless installation of PSE MANAGEMENT.

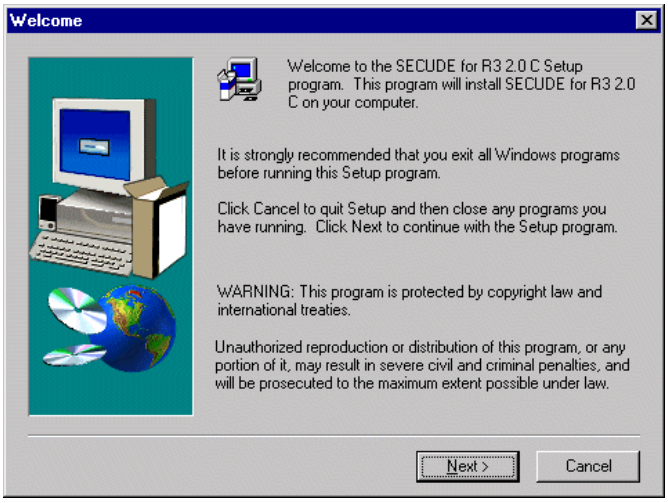


Figure 14: Installation – Welcome

The next dialog is brought up by means of the *Next* button. The software license agreement should be read. Clicking the *Yes* button signifies acceptance of this agreement. Otherwise, the *No* button should be clicked.

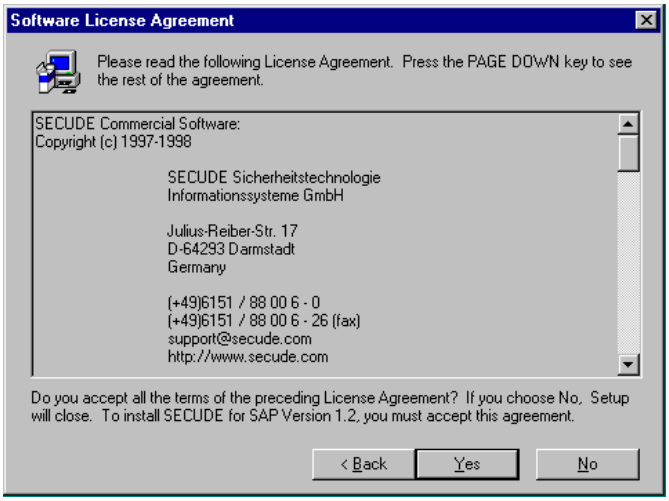


Figure 15: Installation – Software License Agreement

Please enter your name and the name of your company in the following dialog. The fields have to contain these names, otherwise you cannot continue with the installation.

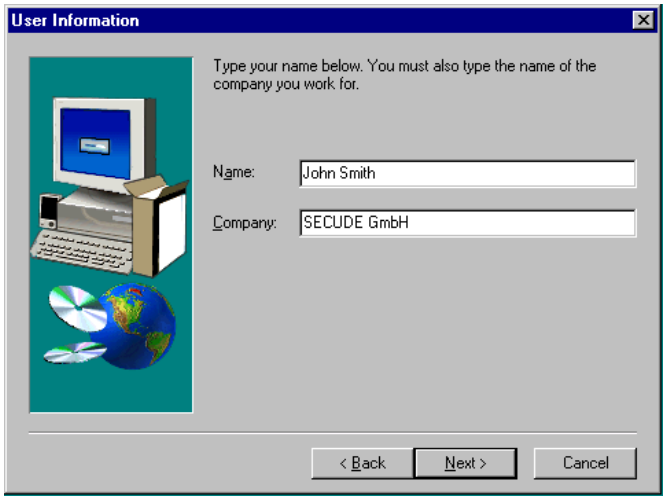


Figure 16: Installation – User Information

Windows 95 and Windows NT (as of Version 4.0) provide the *Program Files* folder for the installation of application programs. When installing PSE MANAGEMENT, we suggest you set up a subfolder named *SECUDE*. You can select a different target directory for the installation by clicking the *Browse* button.

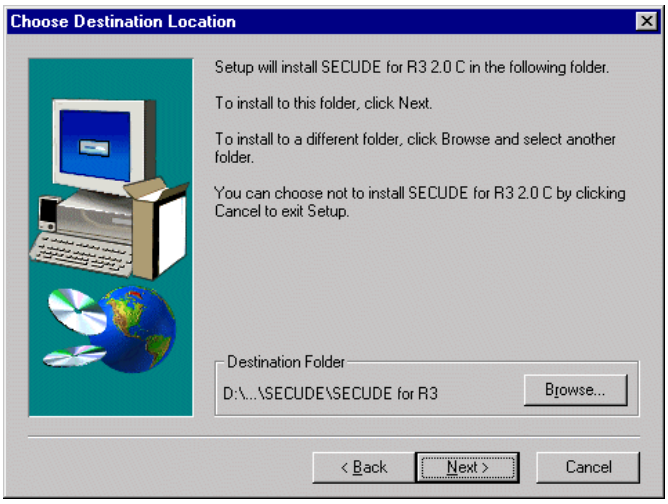


Figure 17: Installation – Choose Destination Location

Once you have selected the installation path, click the *Next* button.

This is your last opportunity to end all programs which access older versions of SECUDE and those for which SECUDE is now to be installed.


Figure 18: Installation – Information: installation starts now

The first dialog box after starting the installation requests the name of the directory where the SECUDE file "ticket" is parked. Click "*Cancel*" if you do not have a "ticket".

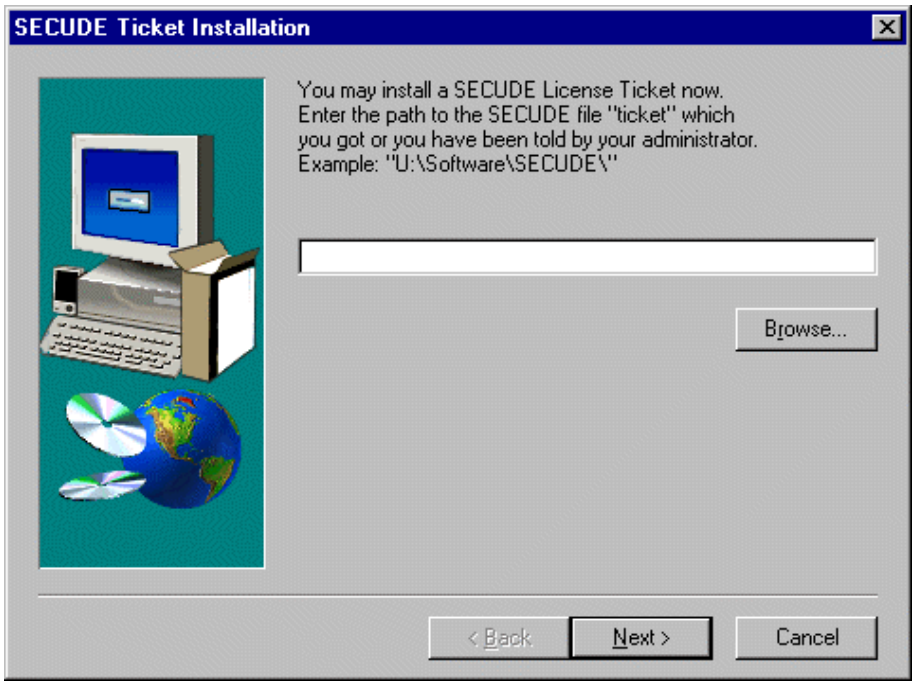


Figure 19: Installation – Secude Ticket Installation

After the "ticket" path has been entered the files are copied onto the hard disc.

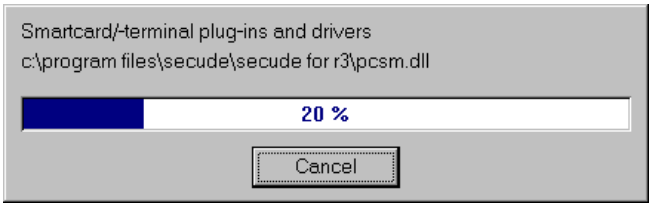


Figure 20: Installation – Files are copied into the installation directory

Once the installation procedure is finished, the following window is displayed. Confirm by clicking *OK*.

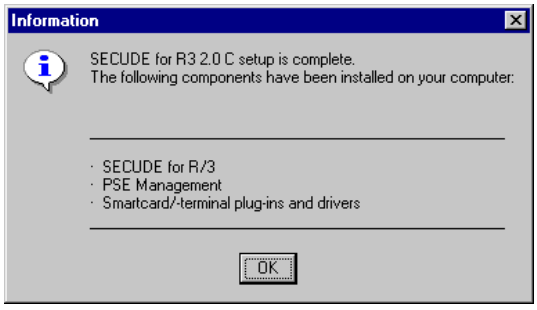


Figure 21: Installation – Information: installed components

The request to restart the computer now appears. This is necessary because some components can only then be installed and configured.

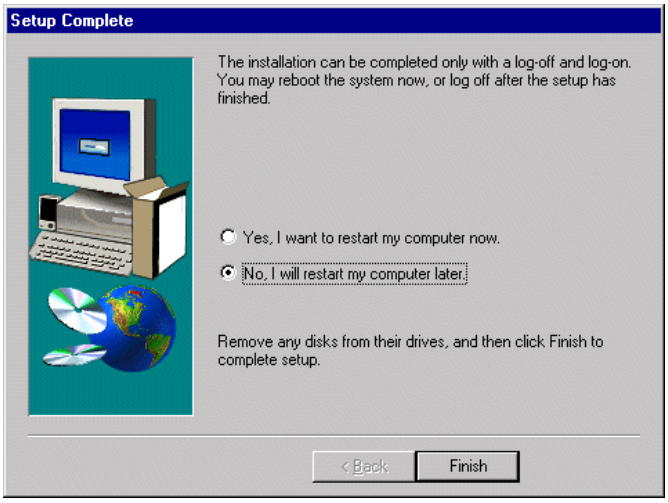


Figure 22: Installation – Setup complete

After successfully concluding the installation you can use the program at once.

### 4.5.3 Installation Termination

The installation program can be interrupted at any time by pressing the *Escape* button (*ESC*) or by clicking *Cancel* in any of the installation windows.

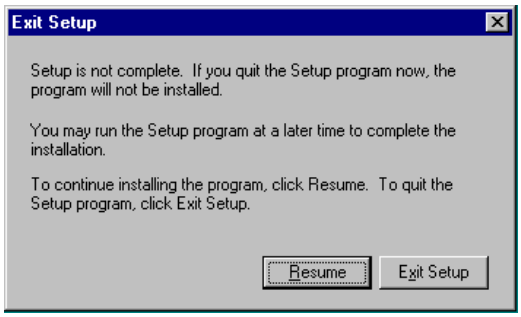


Figure 23: Installation – Exit Setup

Then, to exit the installation, you must press *Exit Setup* in the following window or press the *ESC* key.

### 4.5.4 R/3 Settings

To allow PSE MANAGEMENT to operate with an R/3 front end or SAPlpd, two variables must be set in the system environment and an additional command line parameter must be created to call up R/3. This is done by the installation program.

In Windows 95 the variables are set in the file AUTOEXEC.BAT; under Windows NT they are set in the Control Panel.

The following environment variables are set:

- SNC_LIB <Inst drive:\path..\secude.dll>

The variable *snc_lib* enables the SAPR/3 GUI to locate the SECUDE library containing the security functions (Dynamic Link Library or DLL). The path and the name of the library are entered in full.

Example: set SNC_LIB=C:\Program\SECUDE\...\secude.dll

- SNC_QOP *Value*

  The parameter *SNC_QOP* controls the level of quality that is used for the communication between R/3 server and the SAPgui. Possible values are 0, 1 and 2. The value *zero* ensures that a secure log-on is carried out between the user and the server. A value of *one* also includes an integrity test for the exchanged data when you log on. The highest security level is obtained with the value *two*. Here, the data is also encoded before being sent.

  This is an *optional* parameter. If it is not specified, the highest security level is used.

  Example: set SNC_QOP=2

  If the R/3 application server has set a different QOP, the higher QOP is used.

In addition to the environment variables, the SAP R/3 GUI call must have an additional parameter.

- /snc "p:<Name>"

  The parameter */snc* must contain the server's unique name, the so-called *Distinguished Name,* that is contained in the certificate. The syntax must match exactly, even with blanks in the unique name.

Example: /snc="p:CN=SAPR3srv, O=SECUDE GmbH, C=DE"

**Note:**

The value in the profile parameter */snc/identity/as* for the application server has to have exactly the same name entered as the target name in the variable /snc= in SAPgui at program start.

### 4.5.5 Uninstall

A SECUDE installation is easy to remove. In the *Control Panel* folder (*Start/Settings/Control Panel*) under Windows 95 or Windows NT click on the *Add/Remove Programs* icon. Select the tag *Install/Uninstall* and look for the entry *SECUDE for R/3* in the list of installed software products (the list is in alphabetical order). Select the entry *SECUDE for R/3* and click the *Add/Remove* button. The uninstallation of SECUDE for R/3 begins. Please follow the instructions provided by the uninstaller.

### 4.5.6 Installation Problems and Error Messages

**Wrong configuration of SAPgui.ini**

The following messages indicate a wrong configuration of the SAPgui.ini file:

*No valid entries in the SAPgui.ini file.*

*No entry for SAP R/3 GUI application in the Start menu.*

*Unable to add new entry for SAP R/3 GUI in Start menu.*

*Wrong entry!* [Item*x*] *parameter = in SAP GUI INI file. Please contact your administrator.*

**Installation by different Users**

If SAPgui and SECUDE are installed by different users, it might not be possible to configure some parts of the program completely. This may have effects on the entry in the start menu too. The following message is displayed:

*Unable to add new entry for SAP R/3 GUI in Start menu.*

## 4.6 Configuration and Trace Settings

In the configuration file "profile" in the PSE directory settings for SECUDE can be made.

The file consists of lines in the form

*Setting= VALUE*

The settings below are possible:

| Setting | VALUE | Examples and pre-settings |
|---|---|---|
| GSS/SERVER/ENCRALGS | Encryption algorithms accepted by the server. Possible algorithms:<br>DES<br>DES3<br>IDEA<br>The first algorithm accepted by the client is used for the security of the data to be sent. | = DES3<br>= DES DES3 IDEA<br>*This is the presetting !* |
| GSS/SERVER/HASHALGS | Hash algorithms accepted by the server. Possible algorithms:<br>md5<br>sha1<br>ripemd160<br>The first algorithm accepted by the client is used for the security of the data to be sent. | = sha1<br>= md5 sha1 ripemd160<br>*This is the presetting !* |
| GSS/CLIENT/ENCRALGS | Encryption algorithms accepted by the server. Possible algorithms:<br>DES<br>DES3 | = DES DES3 IDEA<br>*This is the presetting!* |

| | | |
|---|---|---|
| | IDEA<br>The first algorithm accepted by the client is used for the security of the data to be sent. | |
| GSS/CLIENT/HASHALGS | Hash algorithms accepted by the server.<br>Possible algorithms:<br>md5<br>sha1<br>ripemd160<br>The first algorithm accepted by the client is used for the security of the data to be sent. | = md5 sha1 ripemd160<br>*This is the presetting!* |
| GSS/CRLCHECK | When this setting is present, the certificates of the communication partner are checked against revocation lists (CRLs) during the connection phase. | *Presetting: None, no checking of revocation lists* |
| GSS/DIRACCESS | When this setting is present, revocation lists are automatically requested from an LDAP directory server during the connection phase, in so far as this is required. | *Presetting: None, no access to directory* |
| GSS/TRACE | Protocolling connections.<br>The actions below can be protocolled:<br>crd : Request for one's own security information (Credentials). A separate file is opened for each request.<br>in : Connections of a client. A separate file per connection.<br>ac : Connections of a server. A separate file per connection.<br>err : Error in the communication. A separate file per error.<br>The protocol files are stored under "trace" in the PSE directory. When the operating system permits it, the file name includes the name of the communication partner. | = crd in ac err<br>*The presetting does not exist, no protocol is made* |
| GSS/TRACE/NAME | Restricting the protocols to specific communication partners.<br>Separated by "\|" names (or parts of names) of different clients/servers can be specified.<br>Only these communications are then protocolled. | Smith\|Clinton<br>*The presetting does not exist, all connections are protocolled.* |
| GSS/TRACE/LEVEL | Amount of information in the protocols.<br>0 : Short description of the packages<br>1 : Details | = 1<br>= 0<br>*This is the presetting !* |

Please note that the variable names must be upper case.

It is possible that the server is using SECUDE 5.1 whilst the client is working with 5.2. In this case no settings or algorithms should be made in the configuration file. A 5.1 client **cannot** communicate with a 5.2 server.

## 4.7 Installing SECUDE for R/3 Printers

SAPlpd requires four parameters for operating SECUDE securely. To use *secude*, enter the parameters in the WIN.INI file. Using the

SYSEDIT.EXE program, you can edit the file to enter the additional parameters for the SAPlpd. The program SYSEDIT.EXE can generally be found in the directory \WINDOWS\SYSTEM\SYSEDIT.EXE for Windows 95 or in \WINNT\SYSTEM32\SYSEDIT.EXE for Windows NT.

If the environment variable SNC_LIB has already been set globally for using SAPguis securely, you do not need the parameter *gssapi_lib* in WIN.INI.

The four parameters for the SAPlpd should be included in the [snc] section in WIN.INI as follows:

snc/data_protection/use default = 3 (Value set [1, 2, 3])

This parameter does not affect the link to SAPgui. Using this parameter, you determine the protection of outgoing data (for example, when starting external programs using CPIC).

Example: snc/data_protection/use=3

enable default = 0 (value set [0, 1])

Activates or deactivates the protection of the links between an application server and SAPlpd.

Example: Enable=0

gssapi_lib Path for the SECUDE library.

File name with path for the SECUDE.DLL. As an alternative, you can use the environment variable SNC_LIB.

Example: gssapi_lib=c:\programs\secude\secude.dll

identity/lpd (STRING, no default exists)

The distinguished name of the SAPlpd. SAPlpd should have its own identity and therefore its own PSE. You can then print even if nobody is logged on to the PC.

Example: identity/lpd=p:CN=saplpd, O=SECUDE, C=DE

Here is an example of an entry for the WIN.INI file. You should add the section [snc]:

```
[snc]
enable=1
gssapi_lib=C:\programs\secude\secude.dll
identity/lpd=p:CN=saplpd, O=SECUDE GmbH, C=DE
```

If SAPlpd was started successfully, the following messages are displayed:

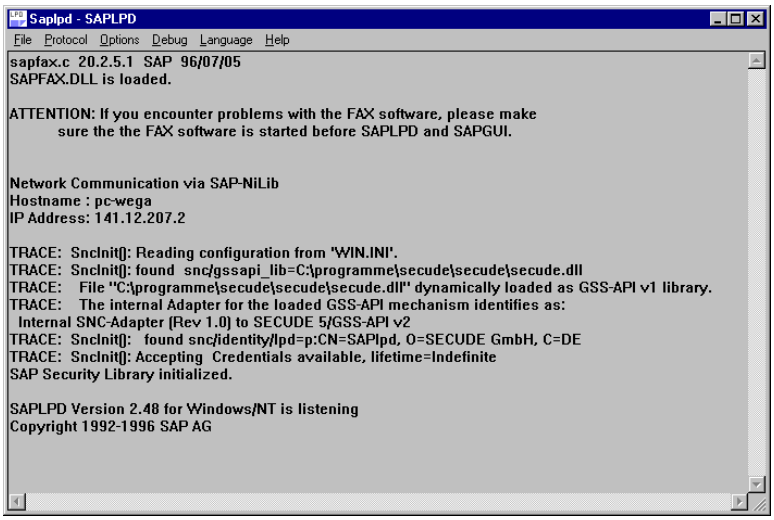


Figure 24: SAPlpd with SNC

Within SAPlpd the security settings are entered via the menu item *Options/Secured Connection*.

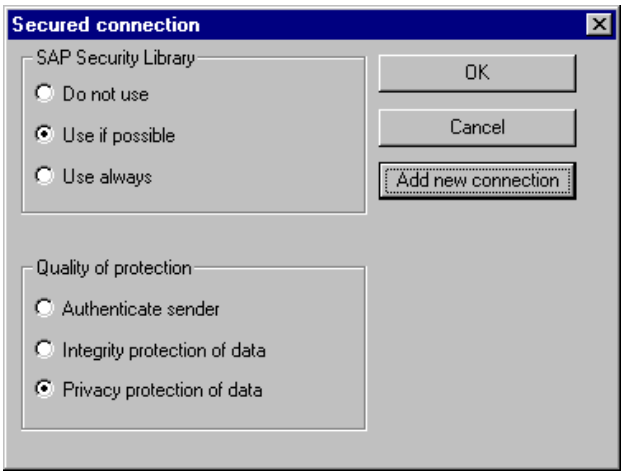


Figure 25: SAPlpd Options: Secured connections

## 4.8 Installing SECUDE for SAProuter

SAProuter supports the use of SECUDE via the SAP-SNC layer. Communication between SAProuters can be secured this way.

To establish secure communication with the SAProuter, the SAProuter has to be notified of the SNC names of the communication partners. The following command line parameters are provided for this purpose:

| | |
|---|---|
| -X myname | activate SNC; myname is one's own SNC name |
| -Y contoname | If SNC is active, contoname is the SNC name of the partner of outgoing connections. |
| -Z accfromnam | If SNC is active, accfromnam is the SNC name of the partner of incoming connections. |

The SNC name is composed of a prefix and the name for the external product. For SECUDE as an external product the prefix is *p:* (see example).

The *saprouttab* is not altered.

### 1:1 Communication

Communication between exactly two SAProuters (SAProuter ⇔ SAProuter) is supported.

Example (NTRouter establishes the connection to UXRouter):

Start of SAProuters on computer 1:

```
saprouter       -r -X "p:CN=NTRouter, O=SECUDE, C=DE"
                -Y "p:CN=UXRouter, O=SECUDE, C=DE"
```

Start of SAProuters on computer 2:

```
saprouter       -r -X "p:CN=UXRouter, O=SECUDE, C=DE"
                -Z "p:CN=NTRouter, O=SECUDE, C=DE"
```

Command lines, of course, have to be entered in a single line and are split in the above example only to provide better legibility. Because of the blanks the SNC name must to be put between quotation marks.

### n:1 Communication

Under certain circumstances several SAProuters can communicate with the same target router.

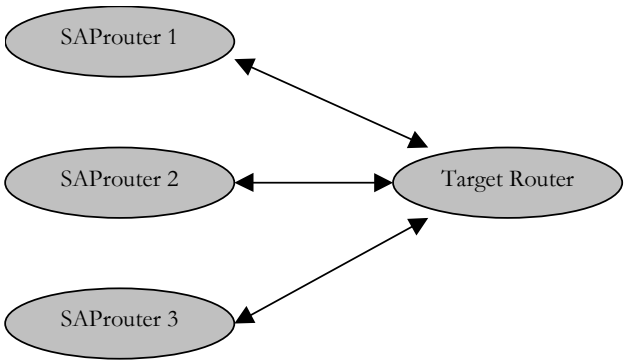


Figure 26: n:1 SAProuter Communication

To do this all partners of the target router are required to have the same name, i.e. the same PSE and the same credentials file.

Example:

Start of SAProuter 1, 2, 3... on different computers:

```
saprouter       -r -X "p:CN=NTRouter, O= SECUDE, C=DE"
                -Y "p:CN=UXRouter, O= SECUDE, C=DE"
```

Start of target SAProuter:

```
saprouter       -r -X "p:CN=UXRouter, O= SECUDE, C=DE"
                -Z "p:CN=NTRouter, O=SECUDE, C=DE"
```

Should problems arise with n:1-communication, the corresponding number of SAProuters may be started on the target computer to work

exclusively with 1:1-communications. In this case the port number has to be started via -S from the 2nd router up, and the SAPgui has to include the port number into the router string.

# 5 Error Handling – SNC

## 5.1 R/3 Application Server

The transaction *st11* contains, when errors occur, additional information. Some examples of errors and their elimination are shown below:

**seclogin was called several times and contains a false or more than one path to PSEs.**

```
N  *** ERROR => SncProcessInput()==SNCERR_GSSAPI
   { GSS-API(maj): Miscellaneous failure
     GSS-API(min): The PSE is not existing  } ***
         => [sncxx.c  1689]

M  *** ERROR => ThSncIn: SncProcessInput
        (SNCERR_GSSAPI) [thxxsnc. 0885]

M  *** ERROR => ThSncIn: SncProcessInput [thxxsnc. 0890]

M  *** ERROR => ThSncIn: SncProcessInput
          (step 3, th_errno 44) [thxxhead 6746]

M  ***LOG R68=> ThIRollBack, roll back ()
          [thxxhead 9410]

M  ***LOG R47=> ThResFree, delete (001044)
          [thxxmode 0703]
```

The error is eliminated by deleting the *cred* file and by then creating the new *cred* filewith the command *secude seclogin -p pfad\psename.pse* die *cred*. It should be noted that the full path and name of the PSE has to be given.

## 5.2 SAPgui Error Handling

The error messages listed here, their descriptions, and the respective actions to take, apply only to error messages that can occur when using SECUDE with SAP R/3.

The following error messages are those displayed by SAPgui:

**Unable to load the GSS-API DLL**

named “sncgss32.dll”

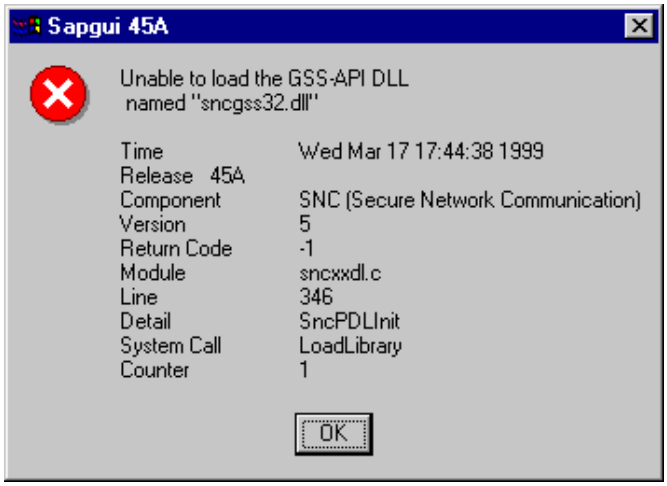


Figure 27: SAP/SNC Error code – GSS-API DLL not found

SAPgui cannot find the necessary SECUDE library. Ensure that the environment variable *SNC_LIB* is set and that it contains a valid path and file name for the SECUDE library (see Chapter *4.5.4 R/3 Settings*).

If the error cannot be eliminated, then reinstall the program.

**Unable to load the GSS-API DLL**

named "D:\Program Files\SECUDE\SECUDE for R3\secude.dll"

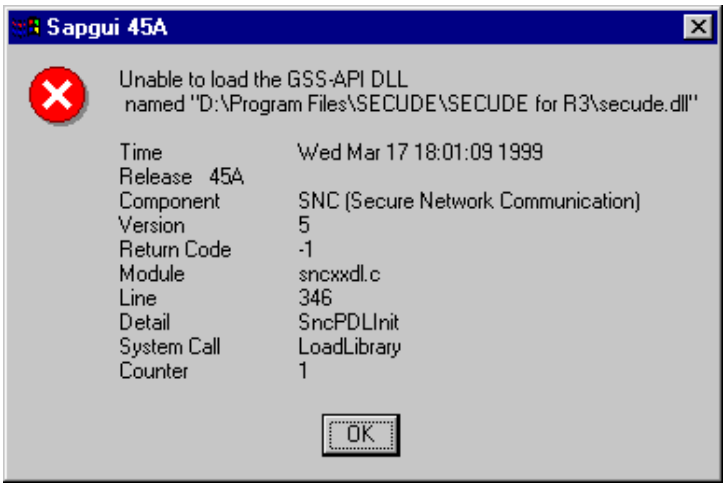


Figure 28: SAP/SNC Error code – SECUDE.DLL not found

SAPgui cannot load some smartcard libraries dependent on SECUDE. Ensure that the smartcard libraries are in the Windows system (see Chapter *4.5.4 R/3 Settings*).

If the error cannot be eliminated, then reinstall the program.

**GSS-API (major): Miscellaneous failure**

GSS-API (minor): No default credentials found

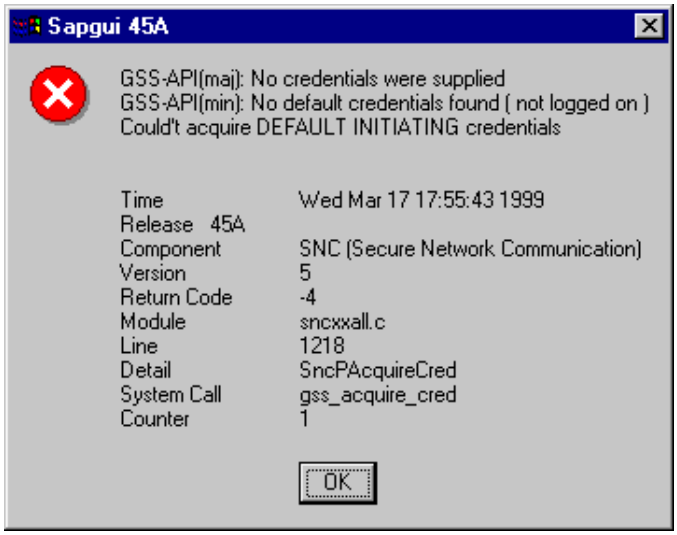


Figure 29: SAP/SNC Error code – no default credentials found

This error occurs if the PSE has not been opened with PSE MANAGEMENT. Please check whether the program PSE MANAGEMENT has been started and that your Distinguished Name is displayed in the status bar (for this, see the manual PSE MANAGEMENT).

**GSS-API (major): A token had an invalid signature**

GSS-API (minor): The name is wrong

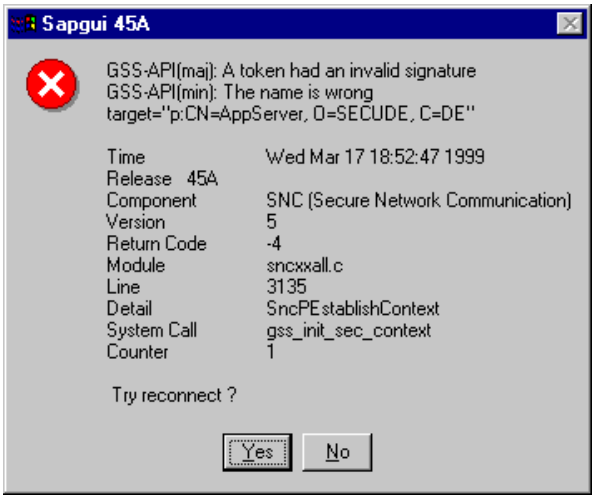


Figure 30: SAP/SNC Error code – The name of the application server is wrong

For the SAPgui call with SECUDE there exists an extra parameter. SAPgui expects to find the *Distinguished Name* of the application server in the parameter */snc*. This error message indicates that the name passed on is not the name of the application server with whom SAPgui is connected (see Chapter *4.5.4 R/3 Settings*).

**SNCERR Invalid Nametype**

Unrecognized name: "..."

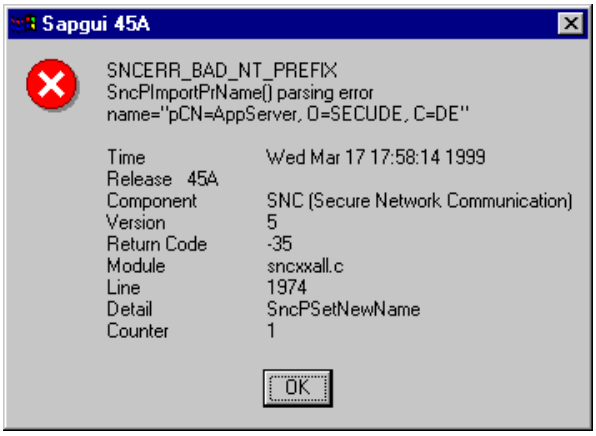


Figure 31: SAP/SNC Error code – Name format is not known

For the SAPgui call with SECUDE, there exists an extra parameter. SAPgui expects a certain naming convention for the name contained in the */snc* parameter. This error message indicates a syntax error. The name as it is contained in the */snc* parameter is displayed in the second line as an *unrecognized name* (see Chapter *4.5.4 R/3 Settings*).

### SAP system message

Error in SNC layer, ...

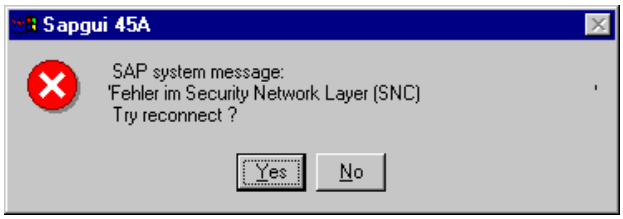


Figure 32: SAP/SNC Error code – Error in the SNC layer

This error occurs if the issuer's name in the PSE's certificate does not match the issuer's name in the application server's certificate. Please ensure that the issuers' names match.

### Not enough Environment Memory (Windows 95)

SAPgui needs the environment variable *SNC_LIB*, which contains the path of the SECUDE for R/3 library *secude.dll*.. For this purpose a batch job is started that sets *SNC_LIB* to the installation directory of the SECUDE library. If the environment memory is not big enough, SAPgui cannot be started. In this case the MSDOS environment memory has to be increased manually. This problem can only happen with Windows 95. Under Windows NT the environment variable *SNC_LIB* is entered to the user profile by the installation program.

# 6 Installing SSF

SECUDE for R/3 – SSF requires an R/3 system from version 4.5A. This system is provided with the necessary interfaces and extensions to use the the security functions offered by SECUDE. Working with a 4.0B client is possible. In this case copy the files *ssfrfc.exe* and *libssf.dll* from the 4.5 version into the SAPgui directory of the 4.0 version.

It is a condition for the operation of SECUDE for R/3 – SSF that PSEs exist for all the participating users. How PSEs are created and used can be found in the manuals for PSE MANAGEMENT and CA MANAGEMENT.

This chapter explains the installation of SECUDE for R/3 – SSF.

## 6.1 Installation of the Client Software

The installation of the client software follows the same steps as under *4.5 Installing SECUDE for SAP R/3 Client.* Please read this section to install the software.

The SECUDE frontend *PSE MANAGEMENT* is required in the SSF installation only to install PSEs, to configure the smartcard reader or to change the password. It does not have the logon function as in the SNC installation.

Please note that the following settings must be made in the file *SAPgui.ini.*

### *For the field [SAPGUI] (see example in 7.2)*

**SSF Library Path Variable**

The value *'ssflib'* may contain the name of the SSF-DLL, it is written into the environment variable SSF_LIBRARY_PATH. If it is not used, *'ssflib'* must be empty or not available.

**Path Variable**

When *'setpath'* has an arbitrary value the environment variable PATH is extended by the SECUDE path. If it is not to be extended, *'setpath'* must be either empty or not available.

Please note that the environment variable *SSF_LIBRARY_PATH* need not be set when you create the file *ssflib.ini.* This file must be located in the same directory as the program *ssfrfc.exe.* You can establish the following parameters:

**`SSF_LIBRARY_PATH`**

SSF library including complete path on the computer.

Presetting: The library *libssf* (with the relevant platform specific ending, e.g. dll for Windows NT) in the same directory in which the program *ssfrfc.exe* is located.

**Note:** The complete file name of the library must be given, not only the path. The somewhat misleading name was kept for compatibility reasons.

**SSF_MD_ALG**

Presetting: MD5

Possible settings at the moment: MD2, MD4, MD5, SHA1, RIPEMD160

**SSF_SYMENCR_ALG**

Presetting: DES-CBC

Possible settings at the momentt: DES-CBC, TRIPLE-DES, IDEA

**SSF_TRACE_LEVEL**

Presetting: 0

Possible settings at the moment:

0 Only the start of the SSF RFC server programs, the loading of the library of the security product and the RFC capable functions in the trace file dev_ssf are protocolled;

1 In addition individual calls of SSF functions with the name of called function and return code are protocolled;

2 Additionally the given signatory/recipient information for every call of an SSF function is protocolled;

3 Additionally the transferred entered data for every call and the relevant display data for every function call are protocolled.

The trace functions are activated with the transaction *SM59*.

The syntax of the ini file must be observed exactly:

- lines beginning with * and empty lines are interpreted as commentaries.
- Every line must be confirmed with *Return*. This applies especially to the last line!
- The option names must be written completely in the upper case.
- Blanks are not allowed between the option name, the equals sign and the parameter.
- The process ends if a faulty line is found. The number of this line is shown in the SSF trace file dev_ssf.

**Note:** The ini file can be explicitly specified by the environment variable SSF_INI_FN. If this environment variable is defined, this specified file is read instead of the file *ssfrfc.ini*.

**Example**:

```
************************************************
* Secure Store & Forward (SSF) Initialization File *
*                                              *
* Michael Friedrich                            *
*                                              *
* 11.09.98                                     *
************************************************

SSF_LIBRARY_PATH=c:\sappcadm\libssf.dll
SSF_MD_ALG=MD5
SSF_SYMENCR_ALG=DES-CBC
SSF_TRACE_LEVEL=0
```

## 6.2 Testing the SSF Basic Functions

Within the transaction *SE38* you can start the test report *SSF01*. With this you can determine whether SECUDE for R/3 – SSF was installed correctly.

Select first the function *SSF_VERSION* and enter *SAP_SSFATGUI* as the RFC destination. Start the report. As result you should get *SSF_API_OK*. SECUDE is then, in principle, correctly installed.

If you get another result, check whether the file *ssfrfc.exe* (*ssfrfc* under Unix) is in the SAPgui directory. This program loads the library *libssf.xxx*. (whereby xxx, depending on the operating system, may be *dll, sl, o* or *so*). The environment variable *SSF_LIBRARY_PATH* must contain the complete path name of this library or the variable must be set in the file *ssfrfc.ini*. This library, in turn, loads the SECUDE library (*secude.dll* or *libsecude.xxx* whereby xxx may be *sl, o* oder *so*). The SECUDE library must, therefore, be in the search path of the operating system.

If you were able to execute the function *SSF_VERSION* successfully, test the function *SSF_SIGN*. To do this you must designate a file which is to be signed. If this file is local, mark also *PC_UPLOAD*. Enter in the wrapper fuctions *IN_ENC*. Then enter the data of the signatory: *RFC Destination* is *SAP_SSFATGUI*. *U/E Name* is the Distinguished Name, as entered in the certificate in the PSE. The *Namespace* is called *SMARTCARD* or *LOCALDIR*, according to whether you are working with smartcards or file PSEs. The *Profilename* contains the complete path name of the PSE or *tcos:* if you are working with smartcards. Then enter the password of the PSE. Note: the password is shown here in clear. When you start the report you should get *SSF_API_OK* as the result. If you get an error message, try to localize the cause using the error message. If you are working with smartcards, try, first, to get the SSF functions to run with file PSEs. Some error sources are eliminated by doing this. An error with smartcards occurs, for example, if the smartcard reader cannot be addressed.

If problems occur when running these tests, please contact the SECUDE team.

## 6.3 Activating the SSF Settings in the R/3 Basis

In order that a user of R/3 can use the SSF functions, such as the digital signature, his user settings must be activated accordingly.

Use the transaction *O07C*.

Enter "*SAP_SSFATGUI*" under *RFC Destination,* when users at the frontend are to use the SSF functions. Enter "*SAP_SSFATAS*", when the functions are to be used at the application server. Normally you will enter *SAP_SSFATGUI*.

As *U/E Name* you enter the Distinguished Name of the user. The Distinguished Name is a part of the certificate that the user has received from the CA. It has the format: *CN=sample name, O=company, C=DE*.

The *Namespace* is called "*SMARTCARD*", when the user is working with a smartcard, otherwise "*LOCALDIR*". Note: even when the file exists as a file, do not enter "*LOCALFILE*" here!

The *Profilename* is the complete path name of the PSE including the file name. When the user is working with a smartcard, the smartcard name given by SECUDE is entered. For TCOS cards you enter "*tcos*" here. If you are using another card brand, please ask the SECUDE team for the correct name.

After you have activated the SSF functions for the users and tested that they function with the report *SSF01* you can use SSF within the SAP application programs. If you have any further questions on this subject, please contact SAP.

# 7 Appendices

## 7.1 The SAPgui.ini file

```
; SECUDE GmbH 02/98
;
[Startup]
;
; DONT'T MODIFY THIS SECTION
; DIESE SEKTION NICHT AENDERN
;
AppName=SECUDE Add-On for SAP R/3
FreeDiskSpace=650
16on16=N
;
[SAPGUI]
;
; ENTER REAL VALUES HERE
; HIER RICHTIGE WERTE EINTRAGEN
;
folder=        SAP Frontend 3.1G
icon=          MySAPGui
newicon=       MySAPGui.secure
path=          C:\SAPGUI\SAPGUI\SAPGUI.EXE
param=         /H/MyAppServer/S/3200 /3
srvdname=      CN=MyAppServer, O=MyOrganisation, C=DE
```

## 7.2 Datei SAPgui.ini for SECUDE for R/3 Version 2.0

```
; SECUDE GmbH 10/98
;
[Startup]
;
; DON'T MODIFY THIS SECTION
; DIESE SEKTION NICHT AENDERN
;
AppName=SECUDE Add-On for SAP R/3
FreeDiskSpace=650
16on16=N
;
[SAPGUI]
;
; ENTER REAL VALUES HERE
; HIER RICHTIGE WERTE EINTRAGEN
;

;creddir=

ssflib=        C:\SAPGUI\SAPGUI\libssf.dll
setpath=       TRUE

setsnclib=     TRUE
sappath=       C:\SAPGUI\SAPGUI\
```

```
; IF sappath IS GIVEN THE FOLLOWING VALUES ARE
EVALUATED:
; WENN sappath ANGEGEBEN IST, WERDEN DIE FOLGENDEN
EINTRÄGE AUSGEWERTET:

folder=        SAP Frontend 3.1G
icon=          MySAPGui
newicon=       MySAPGui.secure
exe=           SAPGUI.EXE
param=         /H/MyAppServer/S/3200 /3
srvdname=      CN=MyAppServer, O=MyOrganisation, C=DE

; ENTER REAL VALUES HERE
; HIER RICHTIGE WERTE EINTRAGEN
;
; if you want to create/modify saplogon.ini
; wenn sie die Datei saplogon erzeugen/anpassen wollen

[Item1]
server=        MyAppServer
database=      00
description=   MySystem
sncname=       p:CN=MyAppServer, O=MyOrganisation, C=DE
sncchoice=     9
```

## 7.3 Registry Entries

The software requires some information that is stored in a registration database, and that is evaluated and set during the running time of SECUDE for R/3. Since these entries are vital to the working of the software, it is advisable not to process the values manually.

The installation program generates the following entries in the registry. These global entries are located in the registry under <HKEY_LOCAL_MACHINE\\SOFTWARE>. There you find the entry <...\\SECUDE>.

- ...\\SECUDE etcdir

  The entry *etcdir* contains the path where PSE MANAGEMENT searches for files (e.g., for initializing smartcards).

  The standard value for this is
  <installation disk drive:*ProgramFiles**SECUDE**etc*>)

- ...\\SECUDE\\5.1.8 no value

  This entry contains the version number for the SECUDE software installed.

## 7.4 Traces – SECUDE for R/3 - Version 1.2

(Traces for version 2.0 are discussed in chapter *4.6 Configuration and Trace Settings*.)

Traces allow the supervison of the behaviour of the SECUDE library.

For this purpose a value under the key SECUDE must be entered in the registry:

- HKEY_LOCAL_MACHINE\SOFTWARE\SECUDE\trace_gss = Trace-File

*Trace-File* stands for the name of the file in which the trace information is to be stored. The file must be given with the complete path. The type of entry is a series of characters, for example "C:\TEMP\gsstrc" could stand there.

To demonstrate this please find a logon under the user ID 'user' and the following call of a test program (here gsstest) under a changed user ID (heer testgss).

Call:

```
secude.exe seclogin –p testgss
```

Display:

```
Enter PIN for testgss:
CADIR  :
PSENAME: F:\secude\testgss
DNAME  : CN=testgss
Credential added for owner 'user'
```

Call:

```
secude.exe gsstest –m 0 "CN=testgss"
```

Display – could look like this when an error exists:

```
gss_init_sec_context(Major) : Miscellaneous failure
gss_init_sec_context(Minor) : Invalid password (PIN)
Can't initiate security context
```

The trace file is now in the C:\TEMP directory. As it is possible that several traces are running simultaneously the trace file is given a set time to guarantee the uniqueness of the file name.

Call:

```
C:\TEMP>type gsstrc905340267_0
```

Display:

```
GSS TRACE FILE: created 980909112427Z
Got PSE directory path "F:\secude" from
        HKEY_CURRENT_USER\SOFTWARE\SECUDE\psedir
acquired_cred : name = CN=testgss (searching in file F:\secude\cred)
found credential owned by 'unknown' for PSE F.\secude\testgss
```

# 8 Glossary

**CA**

Refer to *Certification Authority*.

**Certification Authority**

*Certification Authority* (CA); creates certificates for users of a security infrastructure and maintains blocking lists (revocation lists).

**DES**

*DES* stands for Data Encryption Standard and is a procedure for encrypting, where the same key is used both for encrypting and decrypting. The key length is 56 bytes. The DES was published on January 15, 1977, in [FIPS PUB 46 "Data Encryption Standard"] by NIST (National Bureau of Standards and Technology).

**GSS-API**

Generic Security Service Application Programming Interface. One of the interfaces developed by the Internet Engineering Task Force (IETF) which enables you to apply security functions to applications.

**Hybrid Procedure**

A combination of symmetric and asymmetric cryptography is called a *hybrid procedure*.

**IDEA**

*IDEA* is a procedure for encrypting, where the same key is used both for encrypting and decrypting. It works with a key length of 128 bytes.

**PIN**

A Personal Identification Number. A *PIN* is comparable to a password or the secret number for a bank card.

**Prototype Certificate**

A prototype certificate is a certificate with a signature that has been created using a private key. The prototype certificate only becomes an actual certificate when the prototype certificate has been certified by a certification authority.

**PSE**

The *PSE* is a personal security environment, required for each user by SECUDE. Security related information is stored in the PSE, including the certificate as well as the relevant secret key.

**Revocation List**

A *revocation list* is a list of certificates that are declared invalid by the issuing certification authority before their expiry date. The certification authority maintains this list and publishes it, this means that the CA updates it at regular intervals and makes it available to all participants.

**RSA**

A cryptographic procedure named after Rivest, Shamir, and Adleman. It is based on the fact that there exist pairs of keys which have a special relationship to each other. You can only decode what has been encrypted by one of the two keys using the other key.

**SNC**

*Secure Network Communications*; refers to the module that enables communication to an external library in the SAP R/3 System. The library is called using GSS-API functions and consequently enables R/3 to access security functions, such as those implemented in SECUDE.

**SSF**

*Secure Store and Forward*, the interface in the R/3 system for digitally signing and encrypting documents.

# 9 References

**[Beut-94]**

Beutelspacher Albrecht, Cryptology, Fourth Publication, Vieweg 1994; Page 135.

**[RFC2078]**

Internet Engineering Task Force – Request for Comment – The GSS API Version 2.

# 10 Figures

# Notes