

ePass1000 Developer's Guide



FEITIAN

Copyright © 2001-2007, Feitian Technologies Co., Ltd.

All rights reserved.

<http://www.FTsafe.com>

Feitian Technologies has made all attempts to make the information in this document complete and accurate. Feitian Technologies is not responsible for any kind of direct or indirect loss or damage from inaccuracies or omissions.

This document will not necessarily reflect all of the updates to the ePass1000 hardware or software.

Revision History:

Date:	Version:	Change:
May 2001	2.00	Initial Release
August 2001	2.10	Second Release
November 2001	3.0	Third Release
July 2002	3.5	Forth Release
May 2003	3.5	Forth Release (Second Amendment)
December 2003	3.5	Forth Release (Second Amendment)
February 2007	4.2	Forth Release (Third Amendment)

Software Developer's Agreement

All Products of Feitian Technologies Co., Ltd. (Feitian) including, but not limited to, evaluation copies, diskettes, CD-ROMs, hardware and documentation, and all future orders, are subject to the terms of this Agreement. If you do not agree with the terms herein, please return the evaluation package to us, postage and insurance prepaid, within seven days of their receipt, and we will reimburse you the cost of the Product, less freight and reasonable handling charges.

1. **Allowable Use** – You may merge and link the Software with other programs for the sole purpose of protecting those programs in accordance with the usage described in the Developer's Guide. You may make archival copies of the Software.
2. **Prohibited Use** – The Software or ePass1000 hardware token or any other part of the Product may not be copied, reengineered, disassembled, decompiled, revised, enhanced or otherwise modified, except as specifically allowed in item 1. You may not reverse engineer the Software or any part of the product or attempt to discover the Software's source code. You may not use the magnetic or optical media included with the Product for the purposes of transferring or storing data that was not either an original part of the Product, or a Feitian provided enhancement or upgrade to the Product. You may not place the software on a server and make it publicly available.
3. **Warranty** – Feitian warrants that the ePass1000 tokens and Software storage media are substantially free from significant defects of workmanship or materials for a time period of twelve (12) months from the date of delivery of the Product to you.
4. **Breach of Warranty** – In the event of breach of this warranty, Feitian's sole obligation is to replace or repair, at the discretion of Feitian, any Product free of charge. Any replaced Product becomes the property of Feitian.

Warranty claims must be made in writing to Feitian during the warranty period and within fourteen (14) days after the observation of the defect. All warranty claims must be accompanied by evidence of the defect that is deemed satisfactory by Feitian. Any Products that you return to Feitian, or a Feitian authorized distributor, must be sent with freight and insurance prepaid.

EXCEPT AS STATED ABOVE, THERE IS NO OTHER WARRANTY OR REPRESENTATION OF THE PRODUCT, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

5. **Limitation of Feitian's Liability** – Feitian's entire liability to you or any other party for any cause whatsoever, whether in contract or in tort, including negligence, shall not exceed the price you paid for the unit of the Product that caused the damages or are the subject of, or indirectly related to the cause of action. In no event shall Feitian be liable for any damages caused by your failure to meet your obligations, nor for any loss of data, profit or savings, or any other consequential and incidental damages, even if Feitian has been advised of the possibility of damages, or for any claim by you based on any third-party claim.
6. **Termination** – This Agreement shall terminate if you fail to comply with the terms herein. Items 2, 3, 4 and 5 shall survive any termination of this Agreement.

Contact Information

World Wide Web:

www.FTsafe.com

Feitian Technologies Co., Ltd.

Tel: +86-10-62304466

Fax: +86-10-62304477

Email: world.sales@FTsafe.com

Addr: 5th Floor, Building 7A, No.40 Xueyuan Road Haidian District, Beijing, P.R China

Zip Code: 100083

Please Email any comments, suggestions or questions regarding this document to us at: Feitian@public3.bta.net.cn.

CE Attestation of Conformity



The equipment complies with the principal protection requirement of the EMC Directive (Directive 89/336/EEC relating to electromagnetic compatibility) based on a voluntary test.

This attestation applies only to the particular sample of the product and its technical documentation provided for testing and certification. The detailed test results and all standards used as well as the operation mode are listed in

Test report No.: 70407310011
Test standards: EN 55022/1998 EN 55024/1998

After preparation of the necessary technical documentation as well as the conformity declaration the CE marking as shown below can be affixed on the equipment as stipulated in Article 10.1 of the Directive. Other relevant Directives have to be observed.

FCC certificate of approval



This Device is in conformance with Part 15 of the FCC Rules and Regulations for Information Technology Equipment.

USB



This equipment is USB based.

WEEE



Dispose in separate collection.

Table of Contents

Chapter1 Introducing ePass1000	1
1.1 Product Overview	2
1.2 Why Use ePass1000.....	2
1.3 Information for ePass1000 Developers.....	2
1.4 ePass1000 Architecture.....	5
1.4.1 Security States.....	5
1.4.2 Device Attributes	5
1.4.3 Cryptographic Services.....	6
1.4.4 File System	6
1.4.5 Multi ePass1000 Token Applications.....	8
Chapter2 Installing the ePass1000 Software	9
2.1 Platforms Supported by ePass1000.....	10
2.2 Installation of the ePass1000 SDK	10
2.3 Installing ePass1000 Drivers and Run-time Library.....	14
2.4 Uninstalling ePass1000 SDK and Run-time Libraries.....	16
Chapter3 The ePass1000 Console Editor	20
3.1 Using the ePass1000 Console Editor	21
3.2 Opening ePass1000.....	22
3.3 Turn on/off the LED	22
3.4 Formatting ePass1000.....	23
3.5 Get and Change Access Control Settings.....	23
3.6 Managing the ePass1000 File System.....	24
3.7 Management of the SO and User PIN.....	25
3.8 Closing ePass1000	25
Chapter4 The ePass1000 Manager	26
4.1 Configuring ePass1000	27
4.2 Initializing ePass1000	27
4.3 Changing the User PIN	28
4.4 Changing the Token Name.....	28
4.5 Unblocking the User PIN.....	29
4.6 Changing the SO PIN	29
4.7 Managing Certificates.....	30
Chapter5 Integrating ePass1000 into Your PKI Application	32
5.1 The ePass1000 PKI Architecture	33
5.2 The ePass1000 PKCS#11 Module	33
5.2.1 Supported PKCS#11 Object Class	34
5.2.2 PKCS#11 Mechanisms Supported by ePass1000	34
5.2.3 ePass1000 PKCS#11 Function Library.....	35
5.3 The ePass1000 CSP for MS CAPI.....	38
5.4 Multi-Token Application.....	40
5.5 The ePass1000 PKI Application Guide.....	40
5.5.1 Configure Certificate Authority	40
5.5.2 Install Root Certificate.....	45
5.5.3 Configuring SSL Encrypted Web.....	48
5.5.4 Request Digital Certificate with ePass1000.....	61
5.5.5 Access SSL Secure Website with ePass1000	62
5.5.6 Receive/Send Signed or Encrypted Email with ePass1000	62

5.5.7 Win2000 Smart Card Logon with ePass1000	67
5.5.8 VPN Remote Logon with ePass1000.....	67
5.6Other Applications	70
Chapter6 Distributing the ePass1000 Application	72
Appendix I Frequently Asked Questions	73
Appendix II Technological Specifications for ePass1000.....	74

Chapter1

Introducing ePass1000

Security is an essential requirement of doing business on the Internet. Doubtless your clients understand the need to secure their E-business. ePass1000 can provide the convenience and powerful protection that they require.

This reference guide is intended to assist you in integrating ePass1000 security into your applications or customer solutions.

This chapter will cover the topics:

- ✓ Product Overview
- ✓ Why Use ePass1000
- ✓ Information for ePass1000 Developers
- ✓ ePass1000 Architecture

1.1 Product Overview

ePass1000 is a fully portable and low cost device that connects to the Universal Serial Bus (USB) port of any Windows 98 or above Personal Computer (ePass1000 may also be used with MAC and Linux operating systems. See Appendix II - Technical Specifications for ePass1000.) ePass1000 does not require an additional power supply or reader and is only about the size of your thumb.



ePass1000 is the ideal solution for user authentication and access control functions. Applications can benefit from ePass1000's built-in MD5-HMAC algorithm unit which provides a powerful challenge/response mechanism for authentication services. The challenge/response authentication model is more secure than the traditional user-name & password model because in challenge/response the "shared secret" information is never exposed during the authentication process.

ePass1000 is also a perfect solution for portable storage of sensitive information. Digital certificates, private keys, passwords, credit card numbers and other security credentials may be safely and conveniently stored on ePass1000 and taken with you.

1.2 Why Use ePass1000

The benefits that ePass1000 can provide to your organization or application center are around protecting and securing network communications.

- ✓ Passwords alone are not secure enough. Users may share passwords, or write them down near their PC. Passwords can be intercepted by network sniffing devices and there are free software programs on the Internet to help hackers crack user passwords. ePass1000 is a terrific solution for two factor security measures. Two factors mean that you need to know something, like a PIN or password, and have something. ePass1000 can be the *something you need to have*.
- ✓ ePass1000 was designed for easy integration with PKI applications that support MS-CAPI or PKCS #11, such as Internet Explorer, Outlook, Outlook Express and Netscape Communicator.
- ✓ Credit card numbers, bank account numbers, certificates and other security credentials can be damaged or deleted by computer viruses, or stolen by hackers or otherwise compromised if they are left in the insecure Personal Computer environment. The ePass1000 on-board MD5 Hash algorithm guarantees that information stored in it is far more secure than it would be in the PC environment.
- ✓ Many of the functions of a smart card security solution may be achieved with ePass1000. But unlike smart card solutions, ePass1000 requires no additional reader, saving money and reducing complexity.
- ✓ ePass1000 is low cost and portable. It is a convenient means of moving security credentials between home and office Personal Computers.

1.3 Information for ePass1000 Developers

There are two kinds of ePass1000 developers:

- ✓ Public Key Infrastructure (PKI) solution developers who want to create programs in a PKI environment.
- ✓ System developers who want to use the low level API interface of ePass1000 to add access control and entity authentications to their programs, and manage memory inside ePass1000 to store sensitive information.

Note: Now ePass1000 provides a low-level API for Win32, Linux and Macintosh. But a PKI interface is only implemented on the Win32 platform.

For PKI developers who want to make use of Public Key cryptography to control access to web servers, intranets, virtual private networks (VPN) or encrypted E-mail, ePass1000's low cost belies the fact that it is a powerful cipher device. In addition to the built-in MD5 algorithm, ePass1000 may also be implemented with RSA, DSA, DH, DES, 3DES, RC2, RC4 and other encryption algorithms.

ePass1000 currently supports two industry standard PKI interfaces:

- ✓ PKCS#11 (Public Key Cryptography Standards from RSA Security Inc)
- ✓ MS CAPI (Microsoft's Cryptographic Applications Programming Interface)

Applications based on these standards may be integrated to ePass1000 without modification. Some common examples of applications that comply with these standards are: Internet Explorer, Outlook, Outlook Express, Netscape Navigator and Netscape Messenger.

ePass1000 has a hierarchical file system similar to those found on PCs, except that the ePass1000 file management system is more specific regarding classifications of permission and security.

Each ePass1000 unit has a unique hardware serial number. The serial number may serve as a unique identifier for ePass1000 enabled applications or administrative functions.

ePass1000 uses the MD5 Hash algorithm to protect passwords and other security information. The hash algorithm, as implemented in ePass1000, makes it possible for the user password (or other security information) to never be get from ePass1000. Instead, ePass1000 will pass a calculation involving the password and a random character string to the application for comparison. This procedure is explained in more detail below.

In Diagram 1.1 ePass1000 is shown connected to a simplified network.

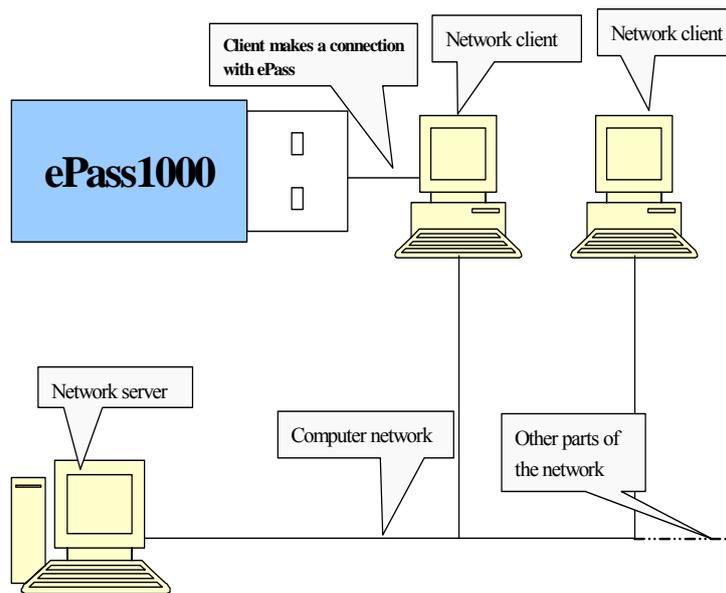


Diagram 1.1

The client machine will first make a logon in request to the server when use secret key to do identification services. The server will then extract the password (or “Key”) that corresponds to the user name. (See Diagram 1.2)

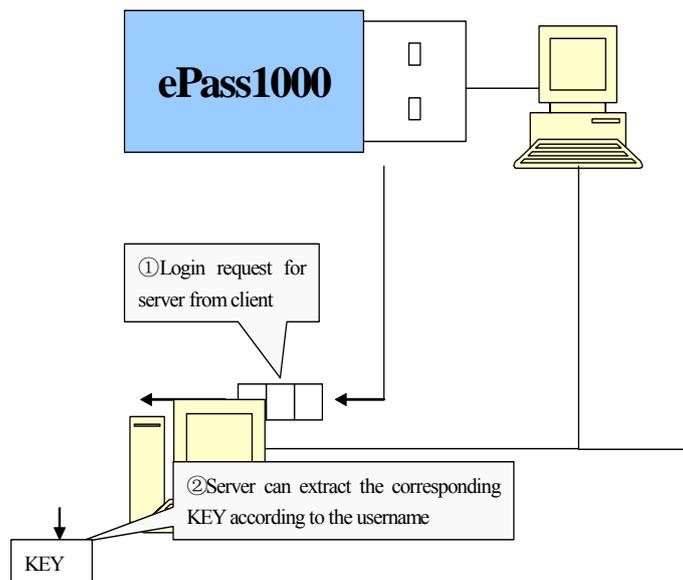


Diagram 1.2

When the server receives the login request from the client, it replies with a random character string X sent back to the

client machine. The client machine forwards the random character string to ePass1000 for calculation. The server also extracts the password corresponding to the user name from its database. The server performs a calculation involving the random character string X that it sent to the client and the user's password. The result is referred to as Rh. (See Diagram 1.3)

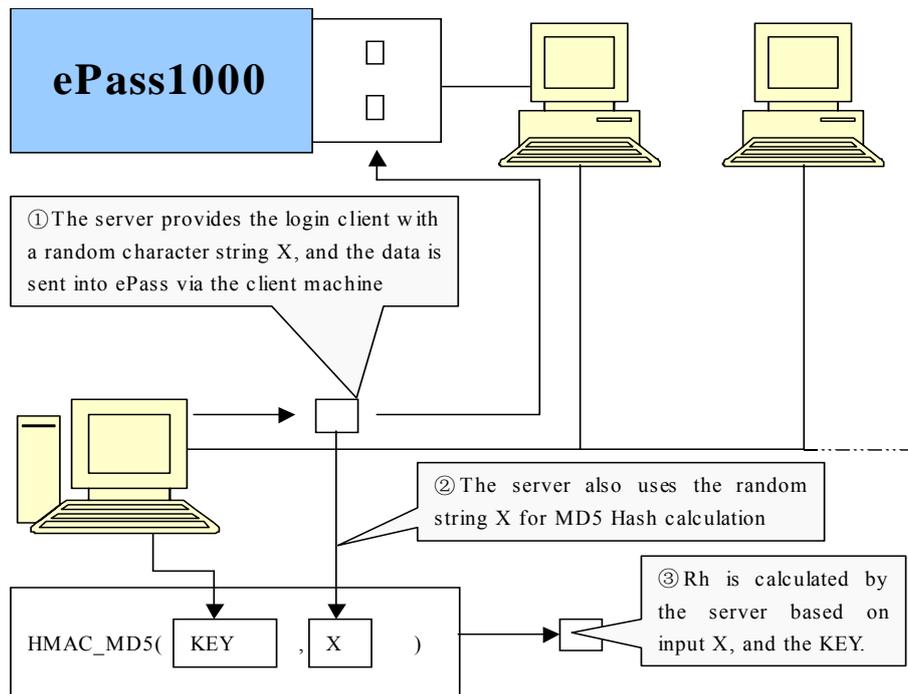


Diagram 1.3

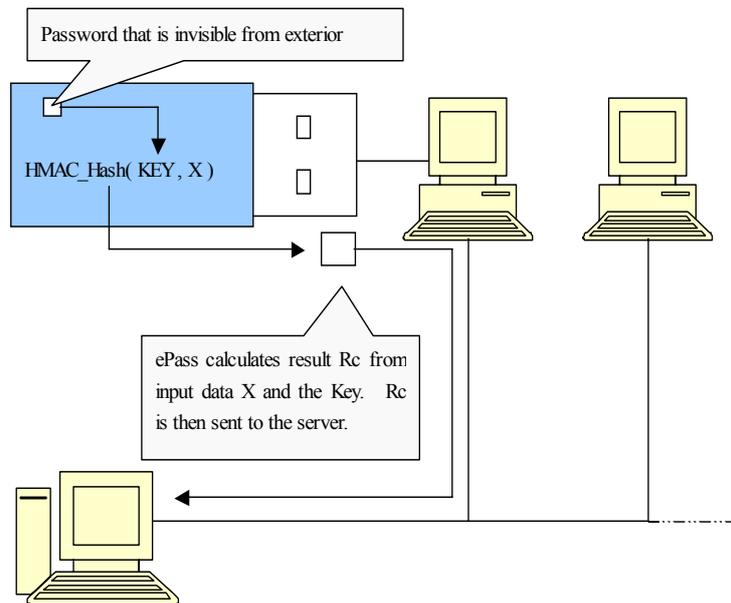


Diagram 1.4

Client sends the random character string X to ePass1000. ePass1000 uses the random character string from the server and the password in its own file system to calculate a result, Rc. The calculated result Rc may then be returned to the server. (See Diagram 1.4)

The server compares the two results, Rh (server) and Rc (client). The user is authenticated to the network if the two results are equal. The logon attempt is denied if the results are not equal. (See Diagram 1.5)

The results (Rh and Rc) change according to the value of the randomly generated character string and the password may not be induced from the calculated results. With ePass1000 in place then, the password is kept safe and secure inside the ePass1000 container, even during the authentication process.

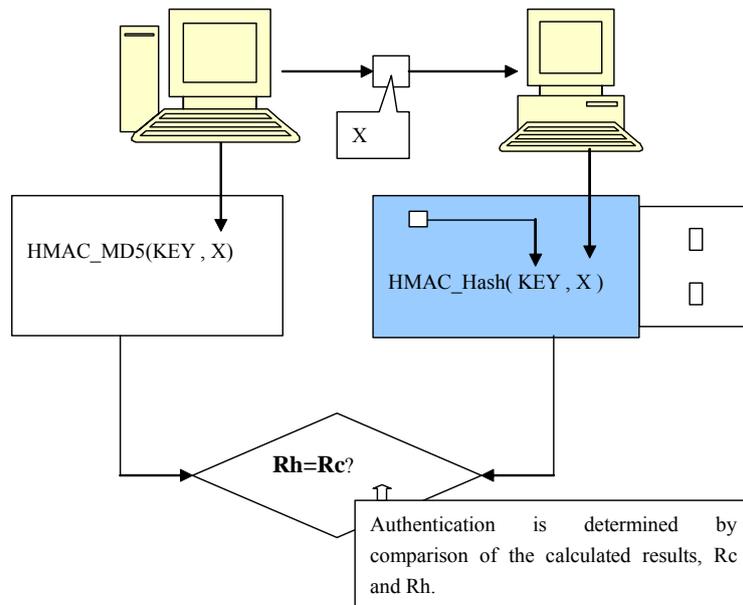


Diagram 1.5

1.4 ePass1000 Architecture

This section describes the basic structure and concepts pertaining to ePass1000.

1.4.1 Security States

The ePass1000 hardware supports a three level security structure: Security Officer (SO), User and Guest.

Security Officer (SO) State

SO is the most privileged security state. SO requires a Super Operator Personal Identification Number (SO-PIN). This state allows changes to sensitive parameter settings and token initialization. If the ePass1000 SO PIN is lost or forgotten it cannot be retrieved. In that case the device must be returned to Feitian where it will be reset to factory defaults.

User State

The User state also requires entry of a PIN. ePass1000 may be configured to allow a user to reset or change the User PIN. Personal information stored in ePass1000 is normally accessed in the User state. There is a hardware counter in ePass1000 to track user logon failure. The counter decreases each time the user fails at an attempt to logon ePass1000. The user is locked out of ePass1000 if the counter decreases to zero. The SO PIN would then be needed to reset the hardware counter.

Guest State

The Guest state is the default state for access to ePass1000. Guest state allows read-only access to limited public information only.

1.4.2 Device Attributes

Serial Number

Each ePass1000 unit has a 64-bit globally unique serial number. The serial number is burned into the unit at the factory and may be used by applications for quick reference to a specific unit.

LED

Each ePass1000 is equipped with a Light Emitting Diode (LED) that can be controlled by applications. And its status can indicate if the driver is successfully installed.

Access Control

ePass1000 supports Global Access Control, which defines access rights required for device command and retrieve functions. Global Access Control applies to all directories and files. There are two access controls: Create and Delete. Create and Delete controls can have one of the following four attributes:

Attribute	Access
ALWAYS	Access is always permitted. Security state is ignored.
NEVER	Never grant access. Security state is ignored.
PIN	Access is granted in User State or SO State
SO PIN	Access is granted only in SO State

Note: Security Officer can access USER PIN protected resources.

1.4.3 Cryptographic Services

Random Number Generator

ePass1000 can generate random numbers in hardware. Random numbers may be used when creating authentication digest code as well as seed for other cryptographic functions.

MD5 algorithm

The MD5 algorithm is an industry standard hashing algorithm that takes a message of arbitrary length as input and produces a 128-bit message digest as output. The output digest is believed non-reversible, meaning that no one can figure out the input data from the output MD5 digest.

MD5 HMAC

Although much more reliable than simple checksum methods, MD5 does not provide a data integrity check, so anyone can alter the input data and generate a corresponding output digest. Obviously, the hashed value needs to be protected. That is the target of the Hashed Message Authentication Code (HMAC). HMAC can be used with the MD5 hash algorithm and a secret key to authenticate a message or collection of data. ePass1000 supports this industry standard method to provide a secure way for end users or applications to be authenticated without exposing their secret keys.

1.4.4 File System

ePass1000 has a built-in file system which can be fully managed from the API library. The file system is a flexible method for storing, protecting and retrieving data in ePass1000.

The ePass1000 file system has the following attributes:

- ✓ 2 levels of directories.
- ✓ File sizes are pre-allocated upon creation. The size of a file cannot be changed after it has been created. To change the size of a file, you must first delete the file and then recreate it.
- ✓ Free space is calculated for the entire token.
- ✓ Files are named with a digital ID, not a character string.
- ✓ The ePass1000 file system supports both 32-bit and 16-bit directories and file IDs.
- ✓ The scope of directory IDs is local to the current directory.
- ✓ The scope of file IDs is local to the current directory.
- ✓ ePass1000 supports named directories, not longer than 32 bytes. Names are global to the entire file system and are case-sensitive. Duplicates are not permitted.
- ✓ ePass1000 supports 16-byte Globally Unique Identifiers (GUIDs) for directories. GUIDs are global to the file system. Duplicates are not permitted.

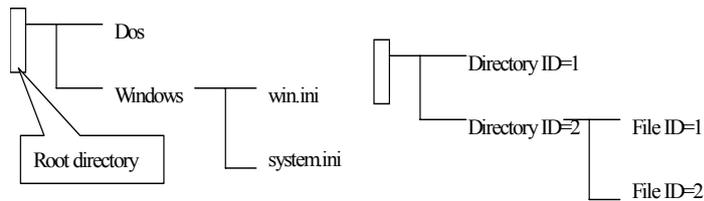


Diagram 1.6

Notes:

The Master File (MF) is the root directory and uses the ID of zero.

The MF may contain files and directories.

Directories are defined by 32-bit ID: 1... 0xFFFFFFFF (ID of 0 is reserved for MF).

File IDs are defined by 32-bit ID: 0...0xFFFFFFFF

Applications should use 16-bit file IDs (0...0xEFFF), and 16-bit Directory IDs (1...0xEFFF) to ensure future compatibility with a new device. Applications should avoid using 32-bit IDs.

Type of Files:

The ePass1000 file system uses two types of files:

Type	Descriptions
DATA	Any variable length binary data
KEY	Data used for cryptographic operations

File Access Settings:

Each file on the ePass1000 has three access types. Each file has its own access setting.

Access Types	File Types	
	Data	KEY
Read	Control	Prohibited
Write	Control	Control
Crypt	Meaningless	Control

Note: Read and Write access are the functions that control ePass1000 data transfers. Crypt access is meaningless for DATA file type. Cryptographic operations on KEY files are performed inside of ePass1000.

File Access Rights:

Attributes	Description
ALWAYS	Access is always permitted. Security state is ignored.
NEVER	Never grant access. Security state is ignored.
PIN	Access is granted in User State or SO State.
SO PIN	Access is granted only in SO State.

Note: Multiple applications accessing the same ePass1000 hardware should coordinate their use of directory IDs and file IDs to avoid collision.

1.4.5 Multi ePass1000 Token Applications

ePass1000 supports multi-token applications; multiple ePass1000 tokens may work together on the same computer.

Chapter2

Installing the ePass1000 Software

The ePass1000 software must be properly installed before it may be used with the computer. This chapter will provide instructions for installing the ePass1000 software in a Windows environment.

- ✓ Platforms Supported by ePass1000
- ✓ Installation of the ePass1000 SDK
- ✓ Installing ePass1000 Drivers and Run-time Libraries
- ✓ Uninstalling ePass1000 SDK and Run-time Libraries

2.1 Platforms Supported by ePass1000

The following platforms are currently supported by ePass1000:

- ✓ Windows 98SE
- ✓ Windows ME
- ✓ Windows 2000
- ✓ Windows XP
- ✓ Windows 2003
- ✓ Windows Vista
- ✓ Linux
- ✓ Mac 8/9/10.X

Note: Please first logon with Administrator privileges before installing the ePass1000 software.

2.2 Installation of the ePass1000 SDK

The ePass1000 SDK software will install the following components on your computer:

- ✓ ePass1000 Console Editor
- ✓ ePass1000 SDK Documents
- ✓ ePass1000 Header Files and Libraries
- ✓ ePass1000 Redistribution Package
- ✓ ePass1000 Sample Codes

Note: Please uninstall any earlier version of the ePass1000 SDK before installing the new version.

The installation of ePass1000 SDK begins by inserting the installation CD into the CD-ROM drive and executes the setup.exe.

The installation welcome window will be displayed. See Figure 2.1:

Note: Here, we use the interface of V4.1 which is almost the same as V4.2 to show the installation process.

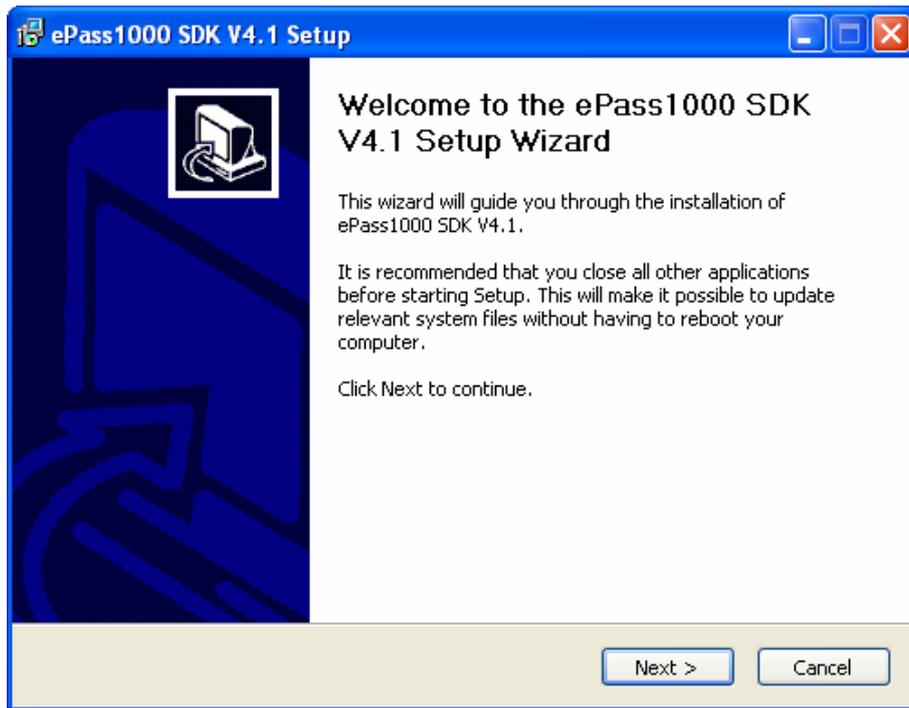


Figure 2.1

Click the “Next” button to continue. The license window will appear. See Figure 2.2:



Figure 2.2

Please read the Software Developer’s Agreement carefully. Click “Yes” if you agree with the terms therein. Program enters the next window. See Figure 2.3:

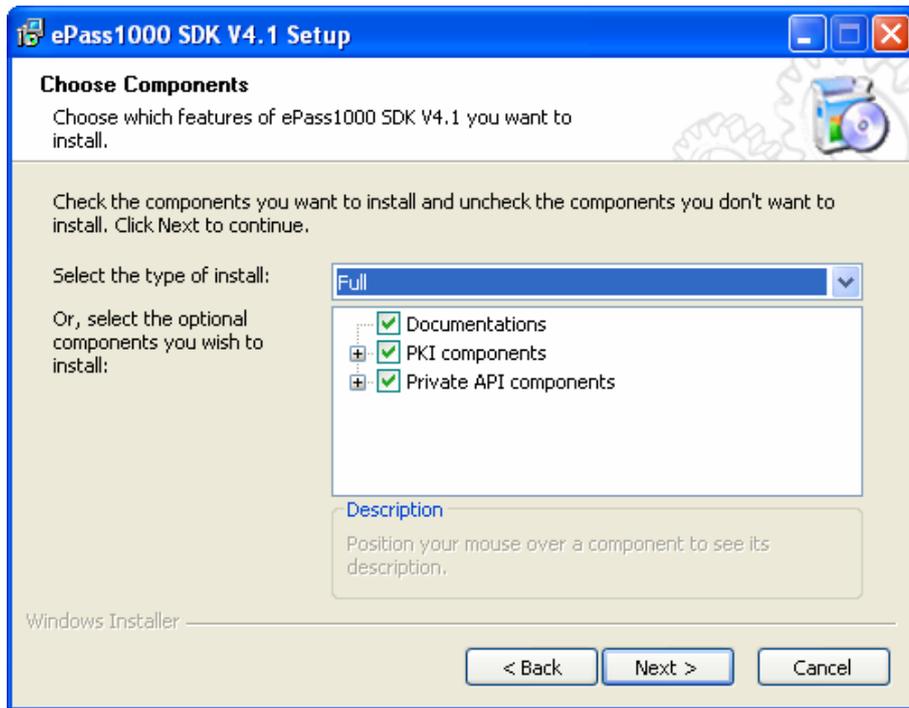


Figure 2.3

Installing program lists all the components for user to choose. User could install the full resources. After selected, click “Next” to enter the next window. See Figure 2.4:

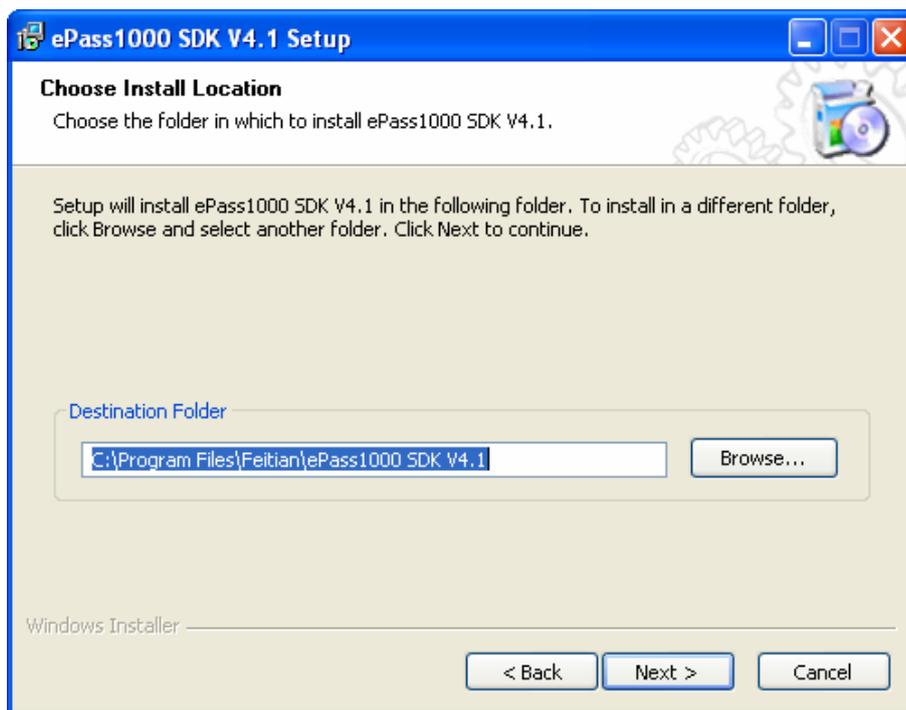


Figure 2.4

Choose the target location to install the SDK. Here uses the default path. User can click “Browse...” button to change to another location. After the location is set, click “Next” to continue.

Enter the “Choose Start Menu Folder” window. Program will install the program shortcuts to the specified program folder. See Figure 2.5:

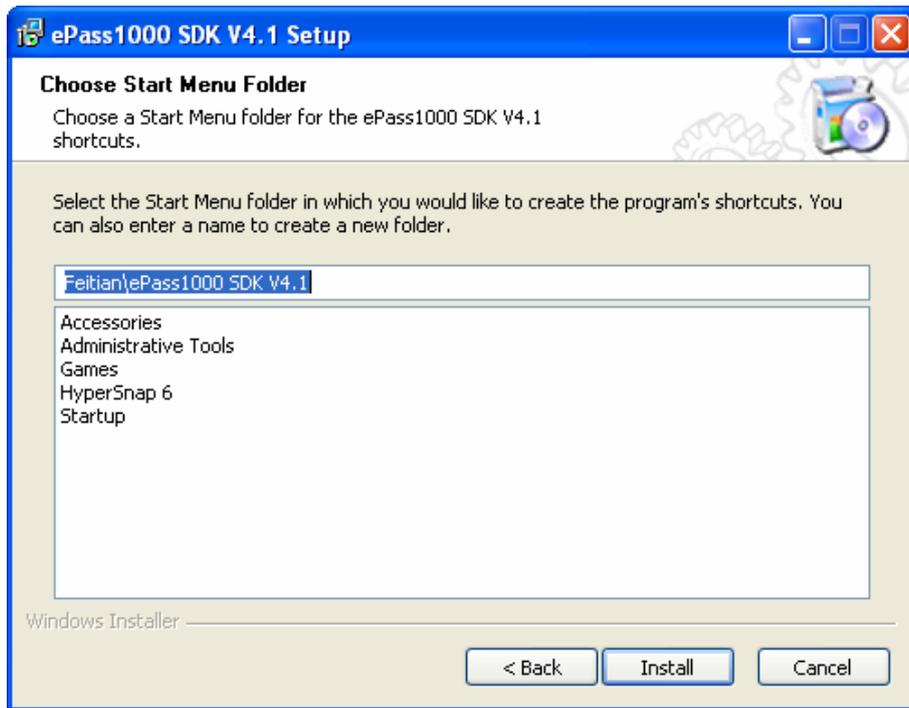


Figure 2.5

Click "Install" button, program will install the SDK to the computer. See Figure 2.6:

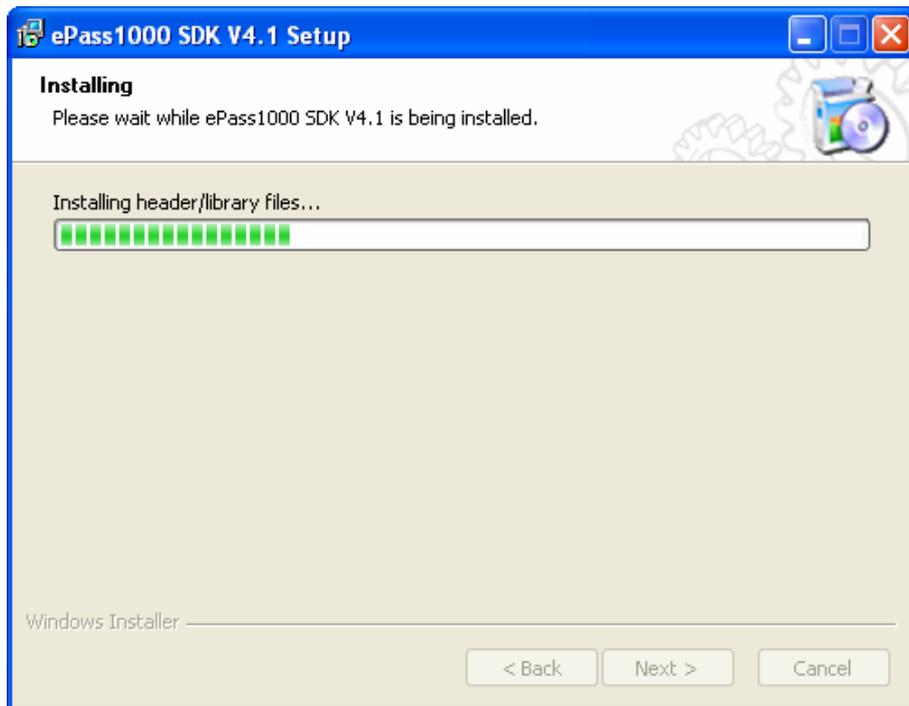


Figure 2.6

Installation finished. See Figure 2.7:

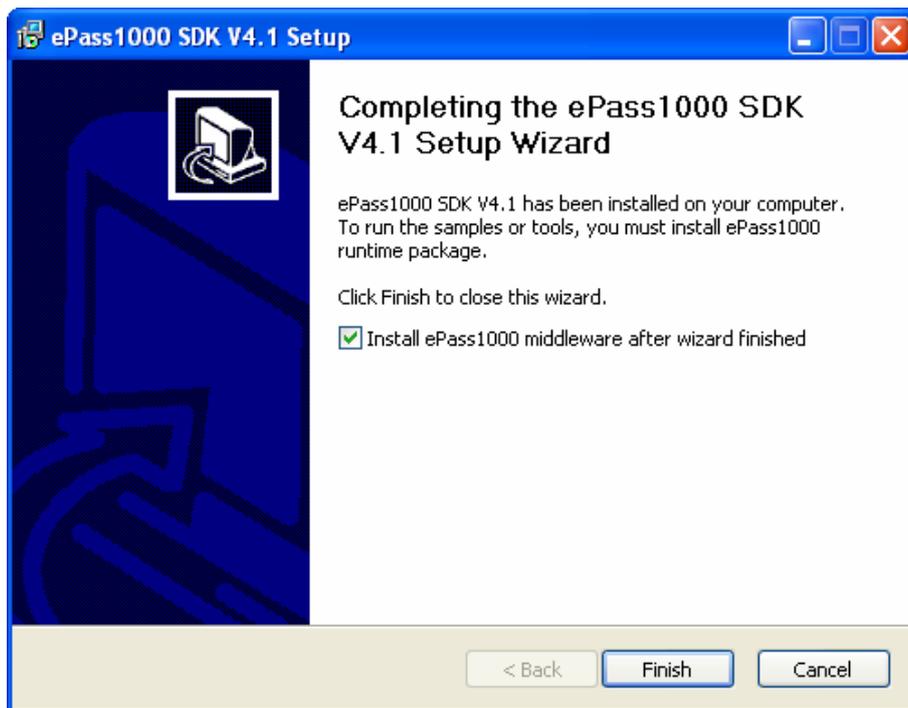


Figure 2.7

Click “Finish” to complete the ePass1000 SDK installation. After finished ePass1000 SDK installation, user could install the ePass1000 driver and runtime library from the ePass1000 SDK4.2 folder, or choose to install the middleware from the option in the last window. Section 2.3 will discuss the redistribution package installation in detail.

2.3 Installing ePass1000 Drivers and Run-time Library

You may choose to install the ePass1000 redistribution package after installing the ePass1000 SDK. User could start from the “Start Menu”. “Start” → “Programs” → “Feitian” → “ePass1000 SDK V4.2” → “Browser Installed Files”. User could find the installed components. For PKI user, double click “PKI” folder. For private user, double click “Private” folder. Here introduces as PKI user.

After entered the PKI folder, enter the Redist folder, user could find the folders CN and EN. In CN folder it is the simplified Chinese version middleware. In EN folder is English version. Here choose EN folder as example. Enter EN folder, double click on epslk_full.exe to start the driver and runtime package installation. The welcome windows will be prompted. See Figure 2.8



Figure 2.8

Click "Next" to start the installation. If user wants to support smartcard logon, please choose the option shown in Figure 2-9.

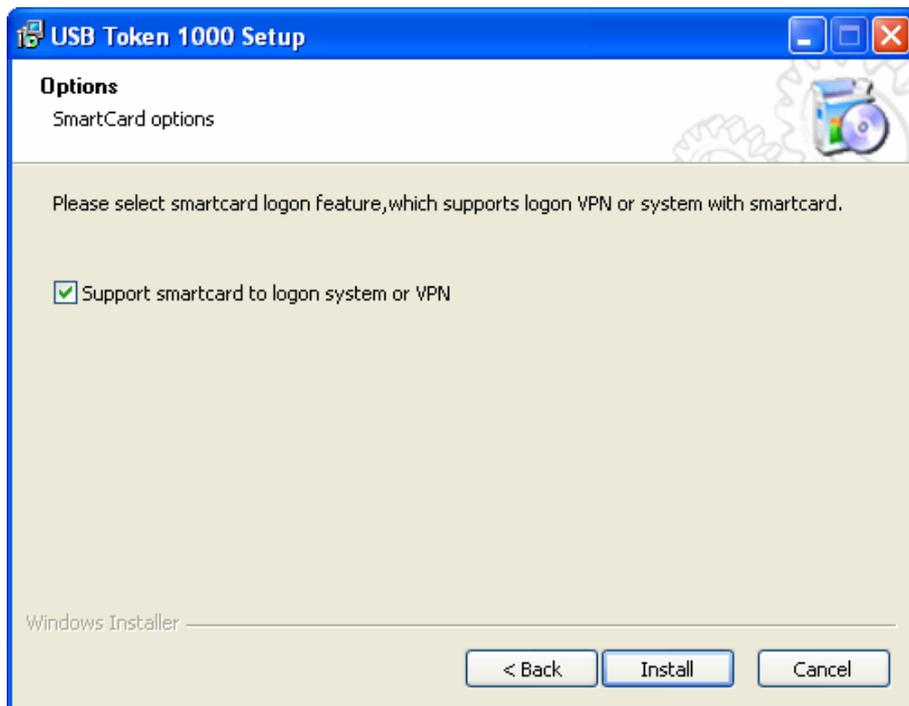


Figure 2.9

Click "Install" to perform the installation, after finished, see Figure 2.10:

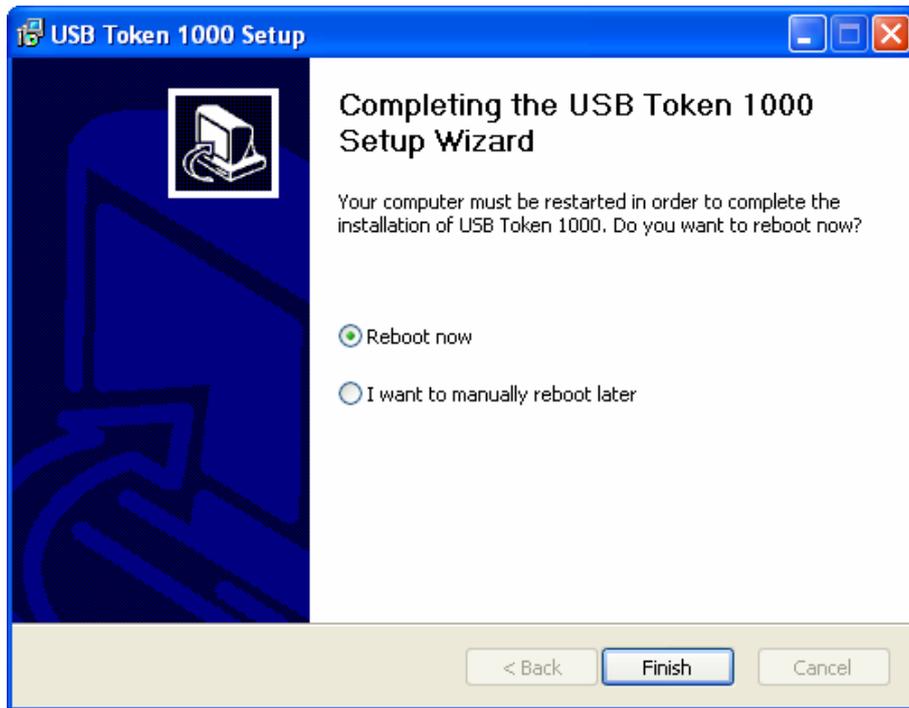


Figure 2.10

Click "Finish", ePass1000 driver and runtime library installation is completed.

If installation program detected Netscape Navigator is installed, it will register the required PKCS#11 components to Netscape Navigator.

2.4 Uninstalling ePass1000 SDK and Run-time Libraries

Uninstall ePass1000 SDK and its runtime library is similar to uninstalling other software. User could choose "Add or Remove Programs" in control panel. If user wants to uninstall the runtime library and keep the SDK, please choose the program from the "Add or Remove Programs" wizard. For example, to uninstall driver and runtime library, see Figure 2.11:

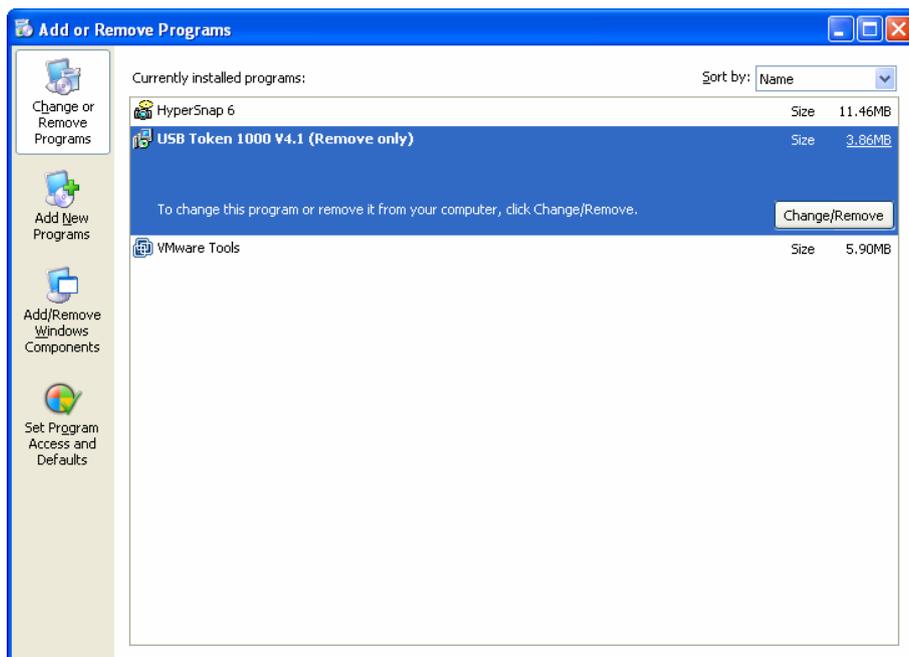


Figure 2.11

Then click “Change/Remove” to continue, a window will be prompted asking user to confirm, see Figure 2.12.



Figure 2.12

Confirm the uninstallation, the driver and runtime library will be uninstalled completely. See Figure 2-13

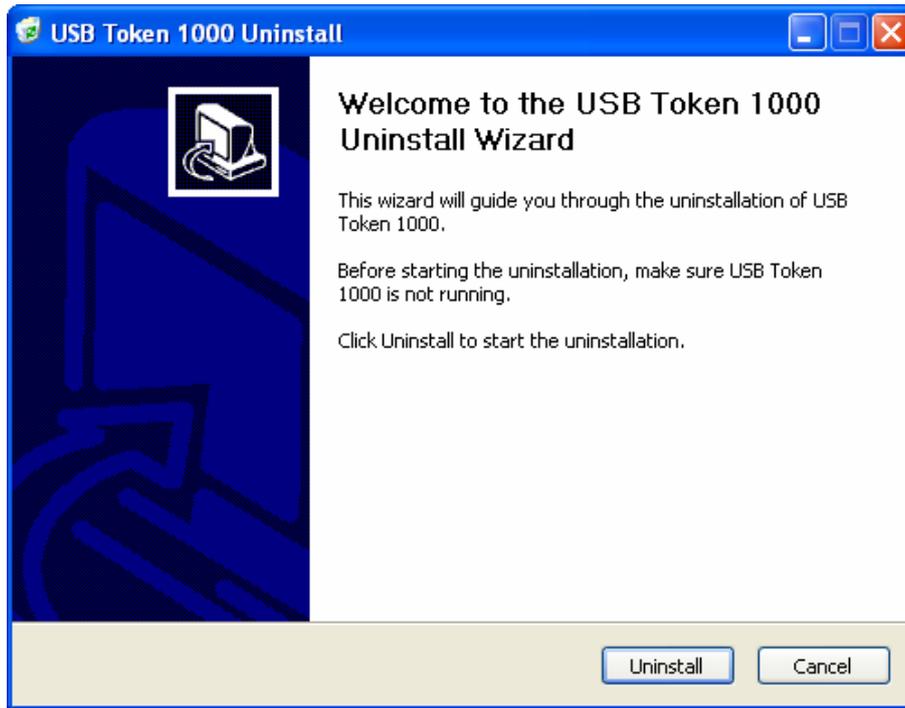


Figure 2.13

After finished, restarting the computer is required. See Figure 2.14:



Figure 2.14

Uninstalling ePass1000 SDK is similar to uninstalling ePass1000 runtime library. User could also start from “Start” → “Programs” → “Feitian” → “ePass1000 SDK V4.2” → “Uninstall ePass1000 SDK V4.2”. See Figure 2.15:

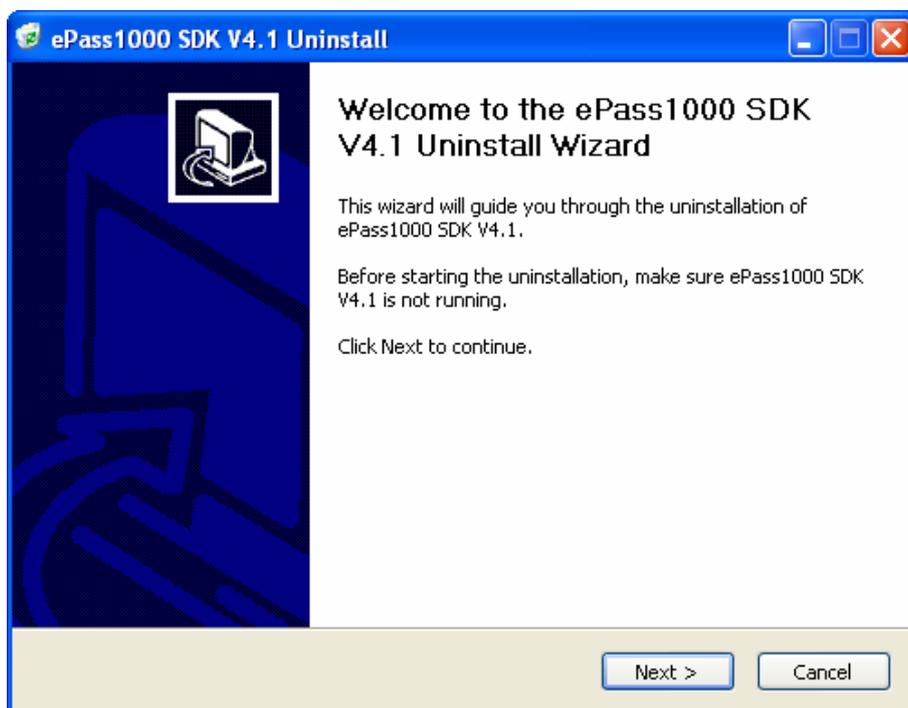


Figure 2.15

Click “Next” to continue, installed component and location will be displayed. See Figure 2.16:

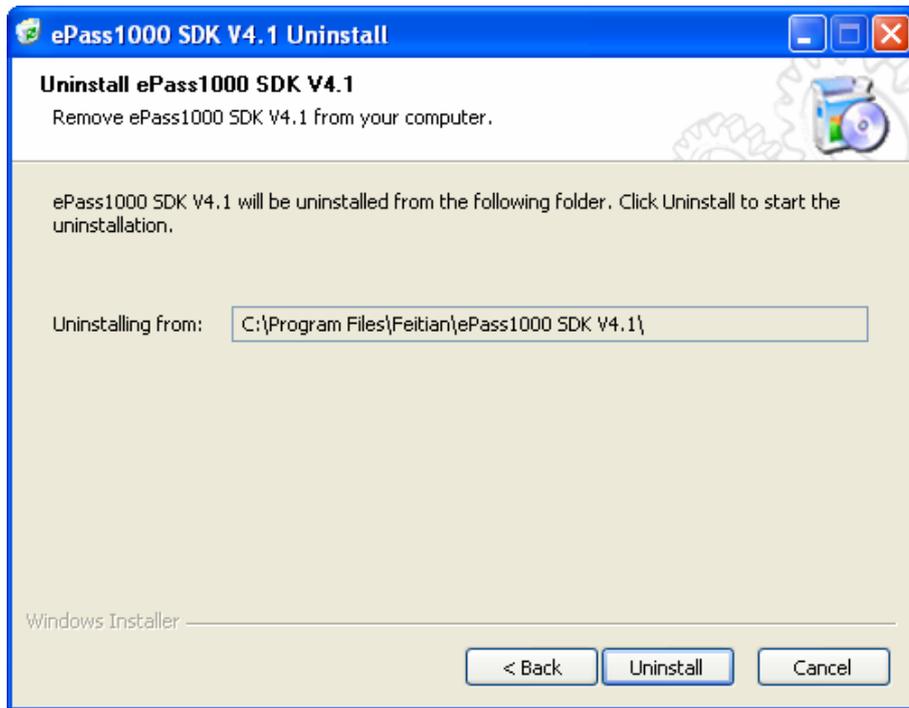


Figure 2.16

After clicking “Uninstall” ePass1000 SDK V4.2 will be uninstalled completely. After finished, program will show the uninstallation is finished. See Figure 2.17:

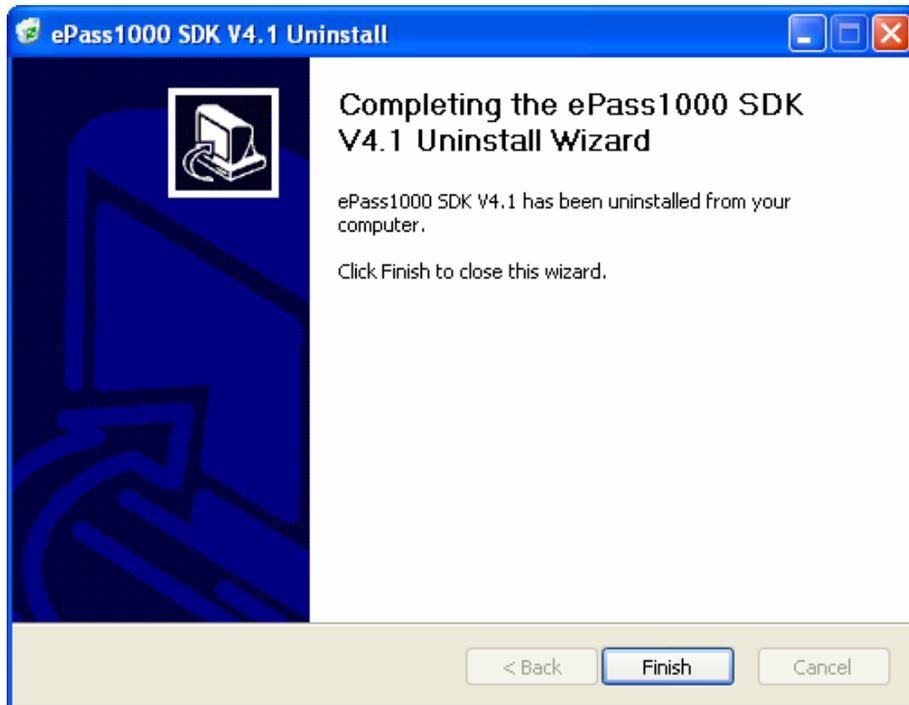


Figure 2.17

Click “Finish” to finish the program.

Chapter3

The ePass1000 Console Editor2

In this chapter we will cover some of the common ePass1000 functions and demonstrate the ePass1000 Console Editor. The ePass1000 Console Editor is programmed with ePass1000's C language API and may be used to manage many of the ePass1000 functions without the programming.

To know more about ePass1000 API, please refer to "ePass1000 Developer's Reference". This document describes ePass1000 API in detail.

Note: PKI developers may want to skip this chapter unless they want to improve their over-all familiarity with the product.

This chapter discusses:

- ✓ Using the ePass1000 Console Editor
- ✓ Opening ePass1000
- ✓ Turn on/off the LED
- ✓ Formatting ePass1000
- ✓ Get and Change Access Control Settings
- ✓ Managing the ePass1000 File System
- ✓ Management of the SO and User PIN
- ✓ Closing ePass1000

3.1 Using the ePass1000 Console Editor

The ePass1000 Console Editor was programmed with ePass1000' C language API set. It may be used to invoke all of the ePass1000 functions. Each command issued in the editor corresponds to one or more C language API functions. You may use it to explore all of the ePass1000 hardware functions and programming interfaces.

The most common ePass1000 programming interfaces are integrated to the Console Editor. You only need use the Console Editor unless your applications need low level control of ePass1000 (i.e. storage management, user authentication or cryptographic operations).

The following table lists the functions that correspond to each operation in the Console Editor:

ePass1000 Operations	ePass1000 C Language API Functions
Open the ePass1000	epas_OpenDevice : Open an ePass1000 that is connected to the computer. You may open the specified ePass1000 by system enumerating order, hardware serial number, etc.
Turn on/off the LED	epas_SetProperty : The LED can be used to indicate the operating state of ePass1000.
Format ePass1000	epas_DeleteDir : Delete all directories and files in ePass1000. epas_SetProperty : Set or change the property of the token.
Get and Change Access Control settings	epas_GetProperty : Retrieve access control settings of ePass1000. epas_SetProperty : Change access control settings of ePass1000.
Manage the ePass1000 File System	epas_CreateDir : Create directories. epas_DeleteDir : Delete directories. epas_ChangeDir : Change current directory. epas_CreateFile : Create files. epas_DeleteFile : Delete files. epas_OpenFile : Open file for further operation. epas_Read : Read data from files. epas_Write : Write data to files. epas_CloseFile : Close a file.
Manage SO and User PIN	epas_ChangeCode : Change User or SO PIN. epas_Verify : Verify User or Security Officer PIN.
Use Cryptographic functions of ePass1000	epas_GenRandom : Get a random number from the random number generator inside ePass1000. epas_HashToken : Perform an MD5 calculation. epas_MD5_HMAC : Perform an MD5 HMAC calculation.
Close ePass1000	epas_CloseDevice : Close ePass1000.

To invoke ePass1000 Console Editor, you may open the Windows Start menu >> Programs>>Feitian >>ePass1000 SDK V4.2>>Use Private API, and then select ePass1000 Editor. The user interface for the editor looks like the following, see Figure 3.1.

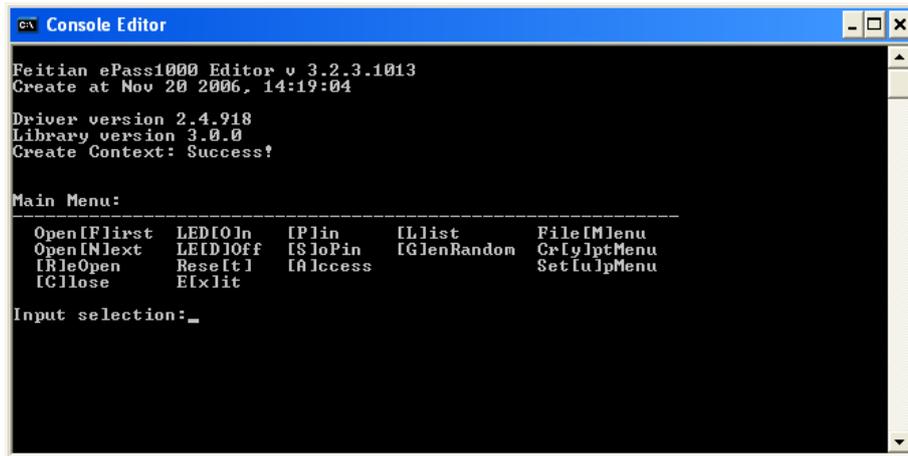


Figure 3.1

3.2 Opening ePass1000

ePass1000 must first be Opened before you can access its functions. ePass1000 is a USB device, and in theory there could be as many as 127 ePass1000 devices attached to a computer, but smart card subsystem of Windows platform supports at most 10 card readers. The C language API supports several ways to find and open ePass1000 devices. For information detail regarding the ePass1000 Open function, `epas_OpenDevice()`, please refer to the ePass1000 Developer's Reference.

Open the first ePass1000 device: After inserting ePass1000 into the USB port, type 'F' and press the 'Enter' key in the Console Editor. The ePass1000 Console Editor will attempt to open the first ePass1000 connected to the system. The error message, "Err: unit was not found! <0x6>" will be displayed if no ePass1000 is found. The Editor will display the screen seen below upon successful completion of the operation, see Figure 3.2.

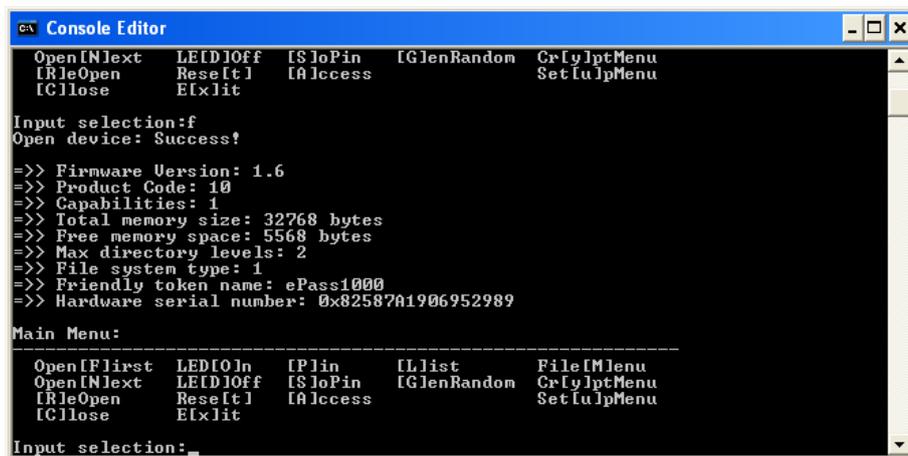


Figure 3.2

3.3 Turn on/off the LED

The ePass1000 LED is useful to indicate the state of ePass1000. The developer may use the LED to indicate when data is being sent to or retrieved from ePass1000. This could help prevent the unintended loss of data. ePass1000 should not be removed from the USB port during a data transfer.

To turn on the LED, input 'O' and 'Enter'. To turn off the LED, input 'D' and 'Enter'.

If the ePass1000 LED is not working properly, it may indicate bad contact, or that the driver was not successfully installed.

3.4 Formatting ePass1000

The formatting operation will empty ePass1000 storage space and zero-initialize it. Feitian recommends that ePass1000 be initialized before distribution. All information inside ePass1000 will be deleted during the formatting operation; this information cannot be recovered. Please perform this operation carefully.

The SO PIN must be verified before the format operation can be invoked. This requirement ensures that only Security Officer can format the key.

- ✓ **To verify the SO PIN:** input ‘S’ and press ‘Enter’, then input the SO PIN and press ‘Enter’. The message “Verify SO-PIN success!” will appear if the entered SO PIN was correct.
- ✓ **To format ePass1000:** input ‘U’ and press ‘Enter’, then input ‘D’ to format from the “Setup Menu”.

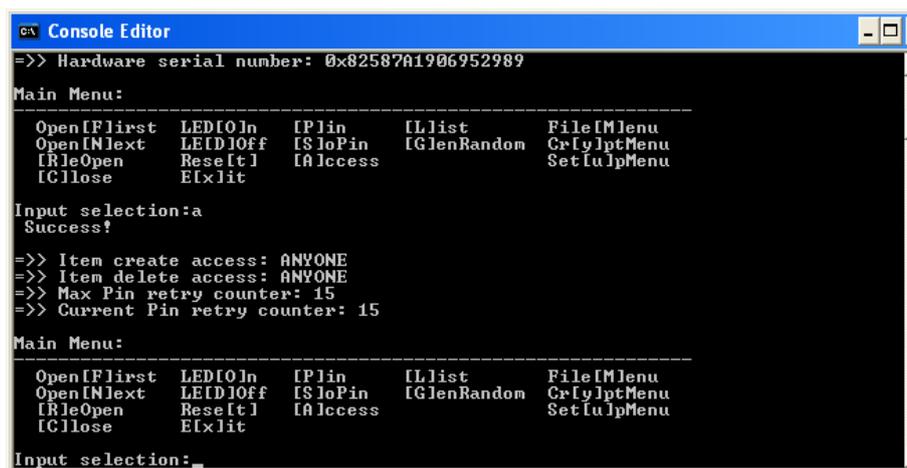
You may assign a friendly name to ePass1000, just like you would label a logical drive.

- ✓ **To set a token name:** access the “Setup Menu”, input ‘T’ and then enter the token name (Maximum of 31 characters).

3.5 Get and Change Access Control Settings

Access control settings are critical for working with ePass1000.

To retrieve current access control settings from ePass1000: from the “Main Menu” of ePass1000 Console Editor, input ‘A’ and ‘Enter’. See the example of an ePass1000 access control settings display in Figure 3.3.



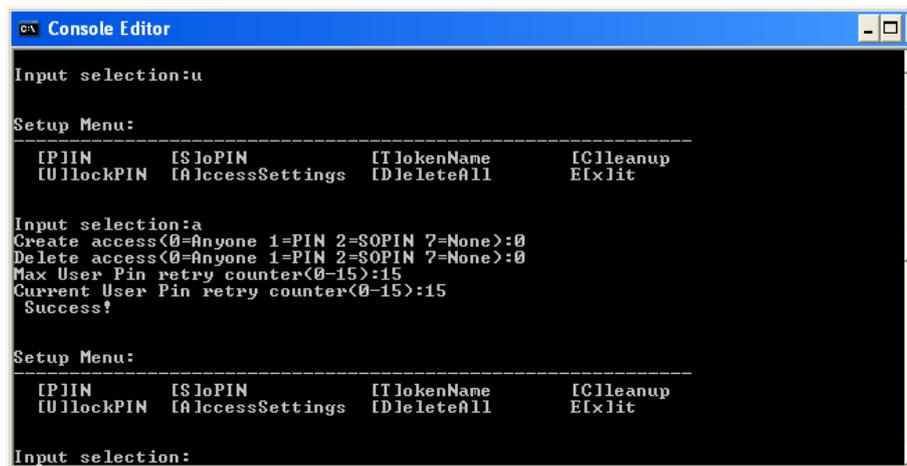
```
CA Console Editor
=>> Hardware serial number: 0x82587A1906952989
Main Menu:
-----
Open [F]irst  LED [O]n   [P]in   [L]ist   File [M]enu
Open [N]ext  LED [O]ff  [S]oPin [G]enRandom Cr [y]ptMenu
[R]eOpen     Rese [t]  [A]ccess Set [u]lpMenu
[C]lose      E[x]it

Input selection:a
Success!
=>> Item create access: ANYONE
=>> Item delete access: ANYONE
=>> Max Pin retry counter: 15
=>> Current Pin retry counter: 15
Main Menu:
-----
Open [F]irst  LED [O]n   [P]in   [L]ist   File [M]enu
Open [N]ext  LED [O]ff  [S]oPin [G]enRandom Cr [y]ptMenu
[R]eOpen     Rese [t]  [A]ccess Set [u]lpMenu
[C]lose      E[x]it

Input selection: _
```

Figure 3.3

To set the access control settings of ePass1000: from the “Setup Menu” of ePass1000 Console Editor, input ‘A’ and ‘Enter’. Then set the value of the access control settings. See Figure 3.4 below.



```
CA Console Editor
Input selection:u
Setup Menu:
-----
[P]IN        [S]oPIN     [T]okenName  [C]leanup
[U]llockPIN  [A]ccessSettings [D]eleteAll  E[x]it

Input selection:a
Create access(0=Anyone 1=PIN 2=SOPIN 7=None):0
Delete access(0=Anyone 1=PIN 2=SOPIN 7=None):0
Max User Pin retry counter(0-15):15
Current User Pin retry counter(0-15):15
Success!
Setup Menu:
-----
[P]IN        [S]oPIN     [T]okenName  [C]leanup
[U]llockPIN  [A]ccessSettings [D]eleteAll  E[x]it

Input selection: _
```

Figure 3.4

3.6 Managing the ePass1000 File System

As we discussed in previous chapter, ePass1000 has a two-level directory file system. You may create, delete, read and write files, and change access control settings to the files. To operate on directories or files, first access the “File Menu” of the ePass1000 Editor.

In the file system of ePass1000 a directory can be identified by the long ID (32 bits), short ID (16 bits), name (ASCII string) or by the GUID (128 bits). Your application may choose one or several of these four ways to identify a directory.

To create a directory with an ID: from the “File Menu” of ePass1000 Editor input ‘D’ and ‘Enter’. Then input the ID of the directory.

To create a directory with a name string or GUID: from the “File Menu” of ePass1000 Editor, input ‘A’ and ‘Enter’. Then input the name string or/and GUID.

To view the settings of all files and directories: input ‘L’ and ‘Enter’. See Figure3.5 below:

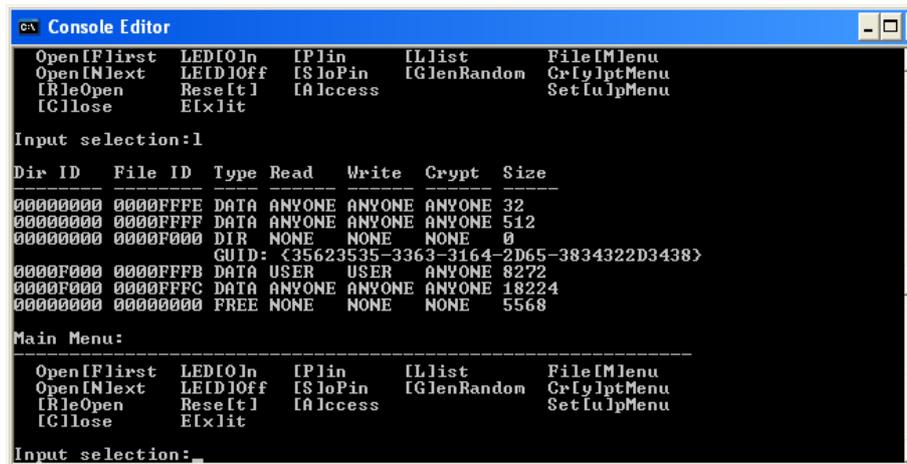


Figure 3.5

To change the current directory: from the “File Menu” of ePass1000 Editor, input ‘H’ and ‘Enter’. Input the ID of the directory.

Only one file may be opened or operated on at a time. The currently open file will close automatically if a new file is opened. There are three file access settings that may be applied to the binary or crypt file types: read access privilege, write access privilege and cryptography access privilege. (See the File Access Settings table in section 1.4.)

When a file is created its size must be specified. The size cannot be changed after the file is created. The file type must also be specified and cannot be changed later either.

To create a file: first enter the directory where you want the file stored. Then from the “File Menu” of the ePass1000 Editor, input ‘F’ and ‘Enter’ and then the file ID, file size, file type and file access settings. See Figure 3.6. Feitian recommends use of a short ID when creating a file. Some file IDs are reserved by Feitian, please refer to ePass1000 Developer's Reference for detail.

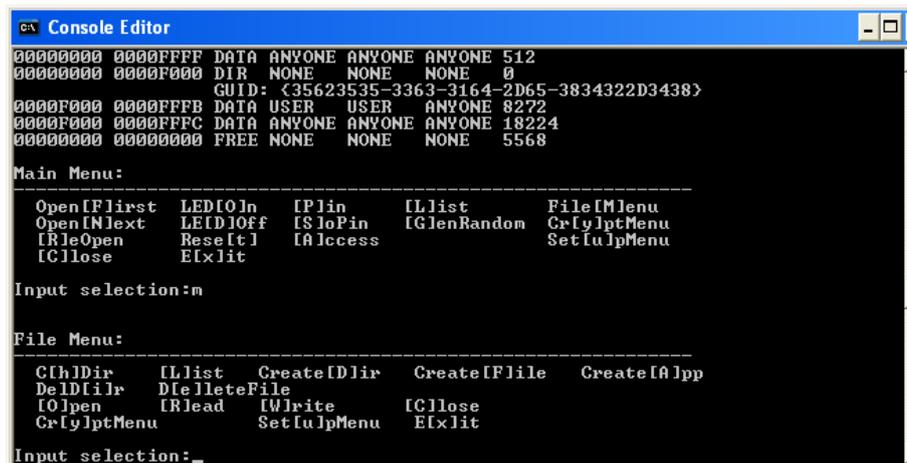


Figure 3.6

To delete a file: from the “File Menu” of the ePass1000 Editor, input ‘E’ and ‘Enter’. Then input the file ID.

3.7 Management of the SO and User PIN

The application should determine if the user will be required to enter an SO or User PIN. If security is not a concern or if the application has some other means of protecting the data, files may be set with an attribute that allows it to be accessed with no PIN.

To change the SO PIN: from the “Setup Menu” of ePass1000 Editor, input ‘S’ and ‘Enter’. Then input the current SO PIN and a new SO PIN.

The factory initialized SO PIN is “rockey”. Before distribution of ePass1000 you should change the SO PIN.

To change the User PIN: from the “Setup Menu” of ePass1000 Editor, input ‘P’ and ‘Enter’. Then input the current User PIN and a new User PIN. The factory initialized User PIN is “1234”. Before distribution of ePass1000 you should change the User PIN.

3.8 Closing ePass1000

ePass1000 is designed to be accessed by only one application at a time. If your application opens ePass1000, it cannot be accessed by any other application until your application closes ePass1000.

To close ePass1000: from the “Main Menu” of the ePass1000 Console Editor, input ‘C’ and ‘Enter’.

Chapter4

The ePass1000 Manager

The ePass1000 Manager integrates many of the ePass1000 administrative functions into a graphical interface. It is a useful utility for assisting PKI developers to use ePass1000. Figure 4.1 is the main interface of the ePass1000 Manager

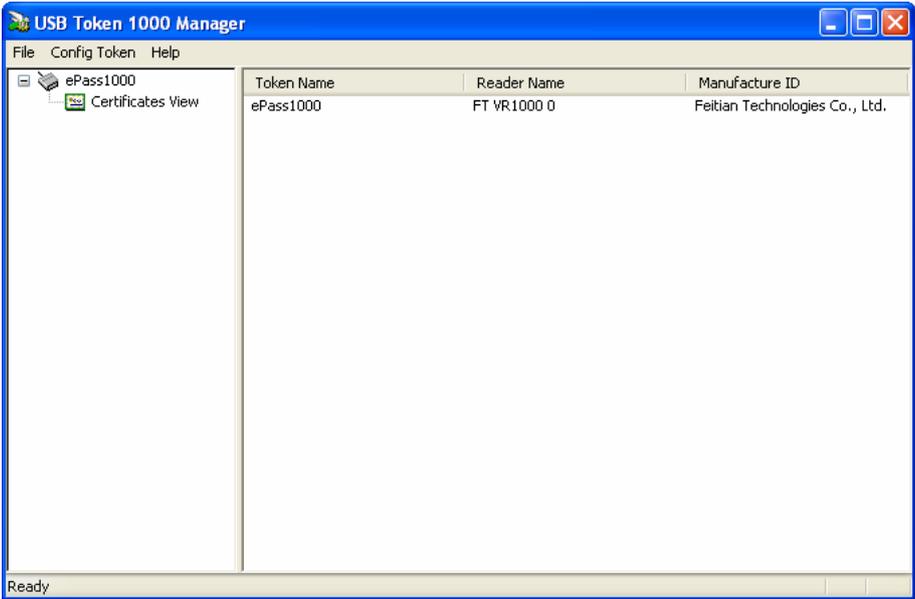


Figure 4.1

From Figure 4.1, it could be seen that ePass1000 Manager’s main functions are ePass1000 configuration and certificate management.

4.1 Configuring ePass1000

Single click the ePass1000 name in the ePass1000 Manager, multiple function tags will be displayed. See Figure 4.2. Choose different function tag can perform different operation.

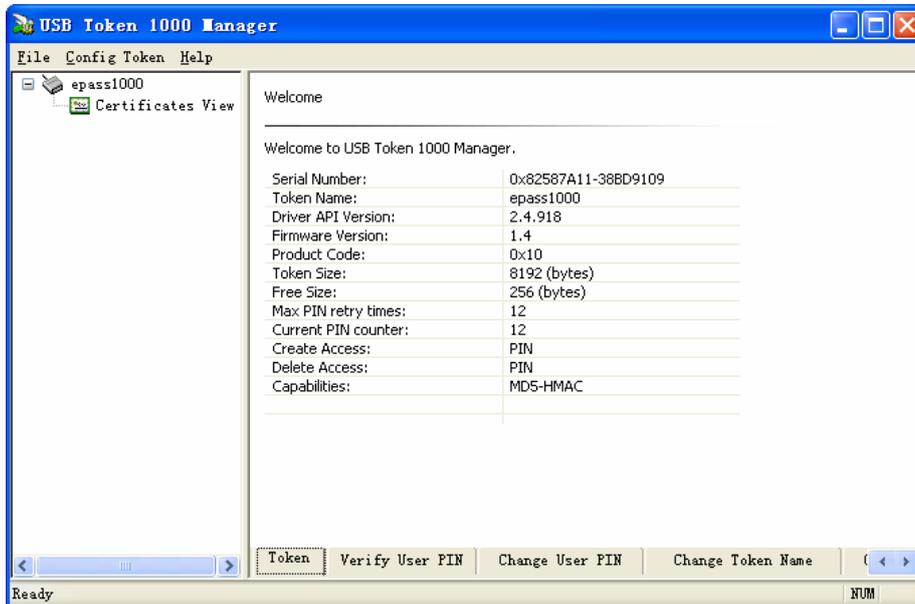


Figure 4.2

Let us discuss the management functions one by one.

4.2 Initializing ePass1000

The initialization function must be performed before ePass1000 can be recognized by an application. The initialization process will create private and public storage spaces. The ratio of the sizes of the two spaces must be well considered; if a storage space is too large it will be mostly wasted, if it is too small it may not be adequate for the application. The initialization function requires that you enter the SO PIN and reset the User PIN. You may also enter a “friendly” token name. See Figure 4.3 below.

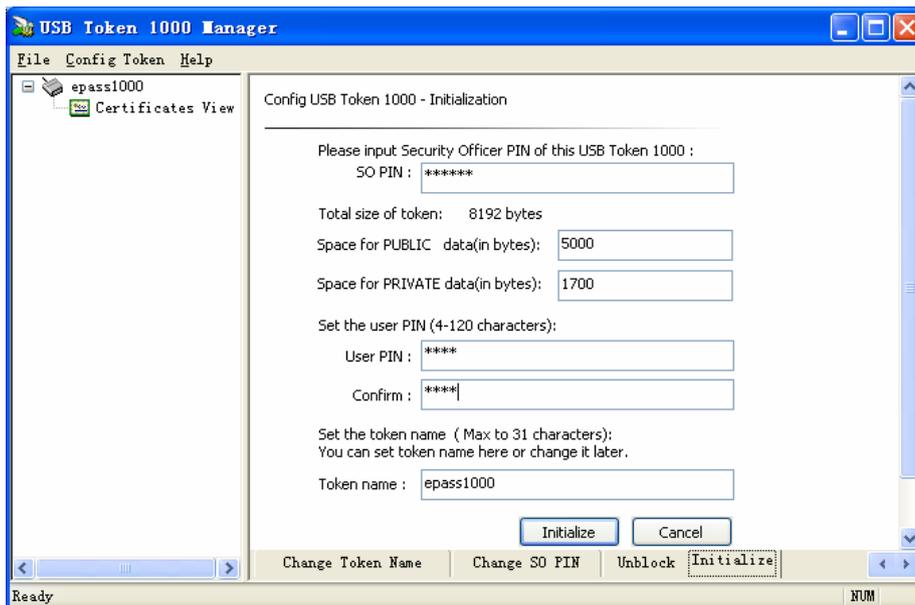


Figure 4.3

You must input SO PIN in the first edit box and the Initialize function will format the device and erase all data stored in ePass1000. Data cannot be recovered once it is erased by the Initialize function. Only the Security Officer is allowed to perform this operation.

You may set the ePass1000 public data and private data space in the second and third edit box. The default values are set by ePass1000 Manager. Please input the User PIN in the fourth and fifth edit box. Enter a name for the token in the last edit box. Be careful with the input field length.

The “Initialize” button will become available after you have properly input all the necessary fields. Click “Initialize” to start initializing the token.

4.3 Changing the User PIN

You may change the PIN code from time to time to enhance the security. This operation keeps all the existing data and just changes the ePass1000 User PIN code. Please click “Change User PIN” table. See Figure 4.4.

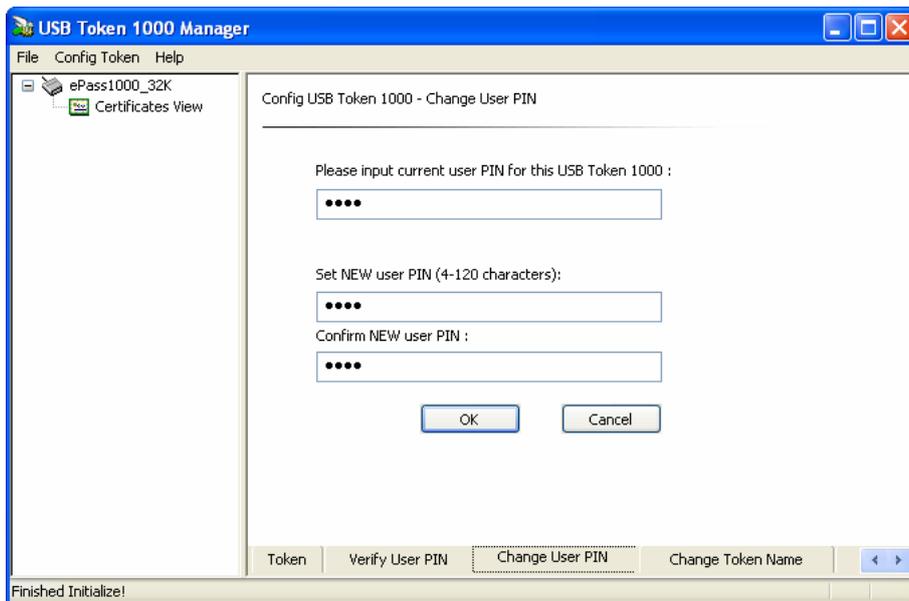


Figure 4.4

Enter the original User PIN in the first edit box and the new PIN in the second and third boxes. Click “OK” to commit the operation.

4.4 Changing the Token Name

Click “Change Token Name” table to invoke the operation window. See Figure 4.5 below.

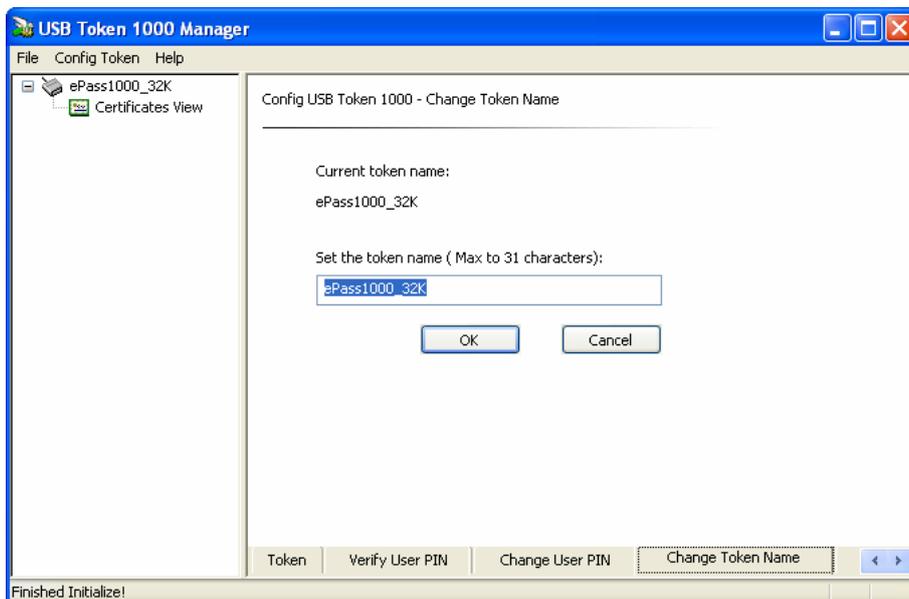


Figure 4.5

The “Change Token Name” window will display the current friendly name of the token. Enter the new token name and click “OK” to commit the operation.

4.5 Unblocking the User PIN

Private data stored in ePass1000 is protected by the User PIN. The PIN code retry number is limited by hardware. Once the preset maximum retry number is exceeded ePass1000 will be locked. Then even you have the correct User PIN you cannot perform certificate applications unless you unblock ePass1000 with SO PIN. The SO PIN is required for this function. Click “OK” to commit the operation, or “Cancel” to cancel the operation. See Figure 4.6.

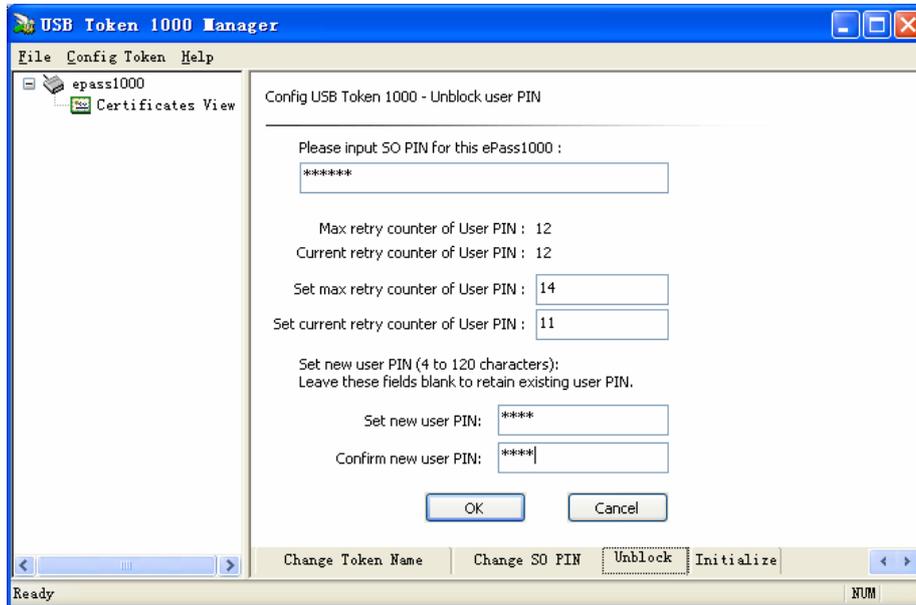


Figure 4.6

4.6 Changing the SO PIN

The SO PIN has the highest level of access to the ePass1000 token. In general it is set by whomever has ultimate control over the ePass1000 based application. You should be very careful to protect the SO PIN. ePass1000 Manager may be used to change the SO PIN. Click “Change SO PIN” table, then you will see one window similar to Figure 4.7.

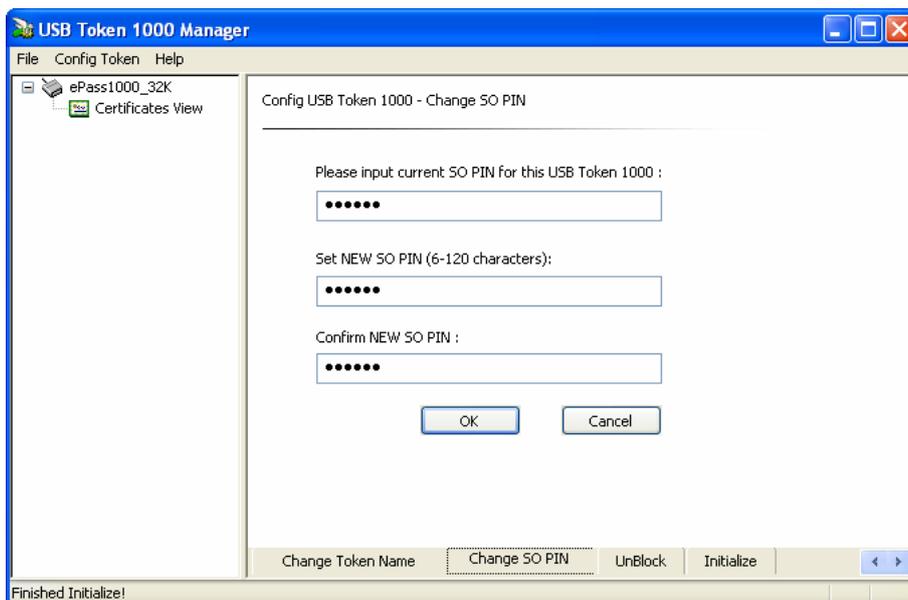


Figure 4.7

4.7 Managing Certificates

The following window will appear when you click “Certificates View” from the tree menu. See Figure 4.8.

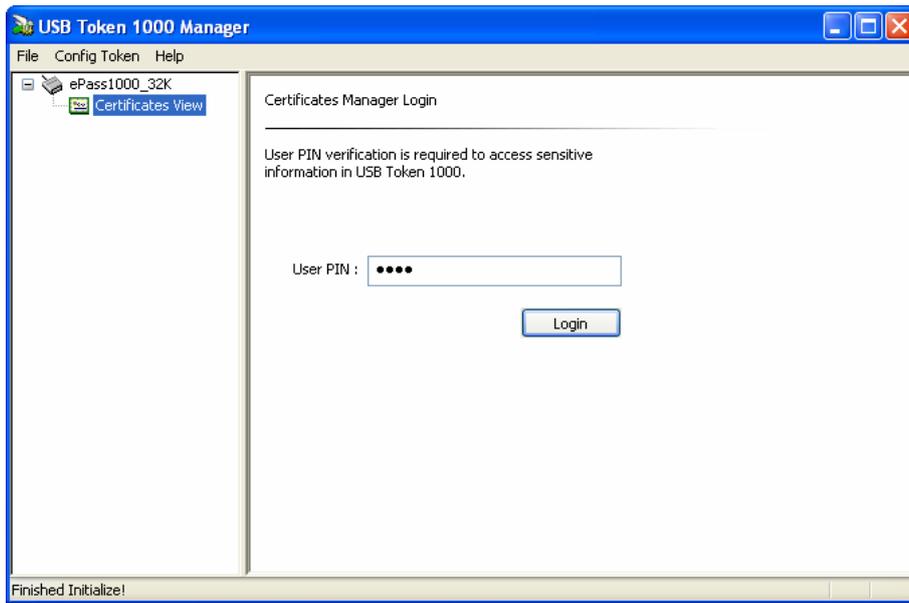


Figure 4.8

To view the certificate information you must enter the User PIN. The window below pictures the screen after entry of the User PIN and “Login”.

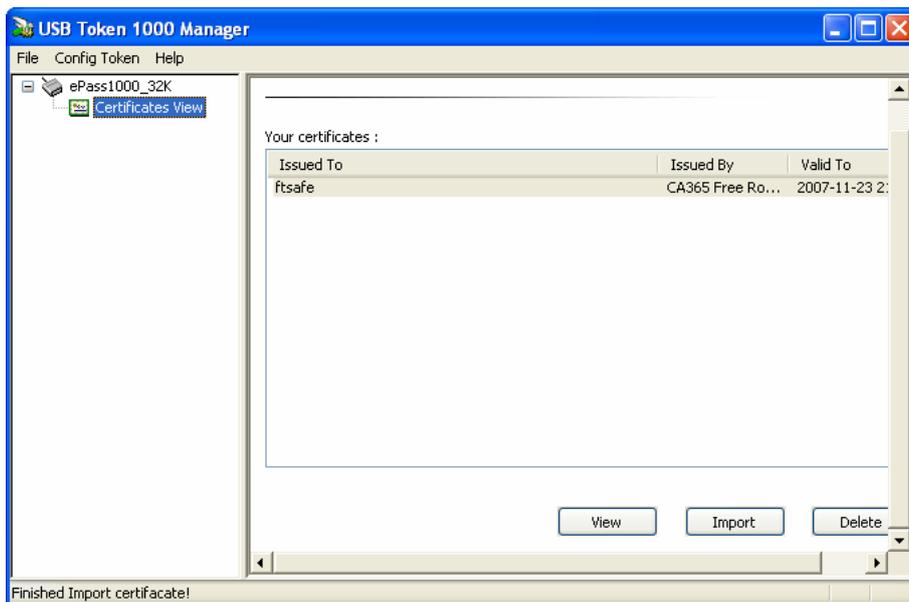


Figure 4.9

You may click “Import” button to import the certificate file from the disk to ePass1000 token; click “Delete” button to delete the invalid certificate; click “View” button or double click the certificate to see the detailed certificate information, if the data is invalid the system will prompt whether you want to delete the current invalid data.

Once the “Import” button is clicked, the following window will appear:

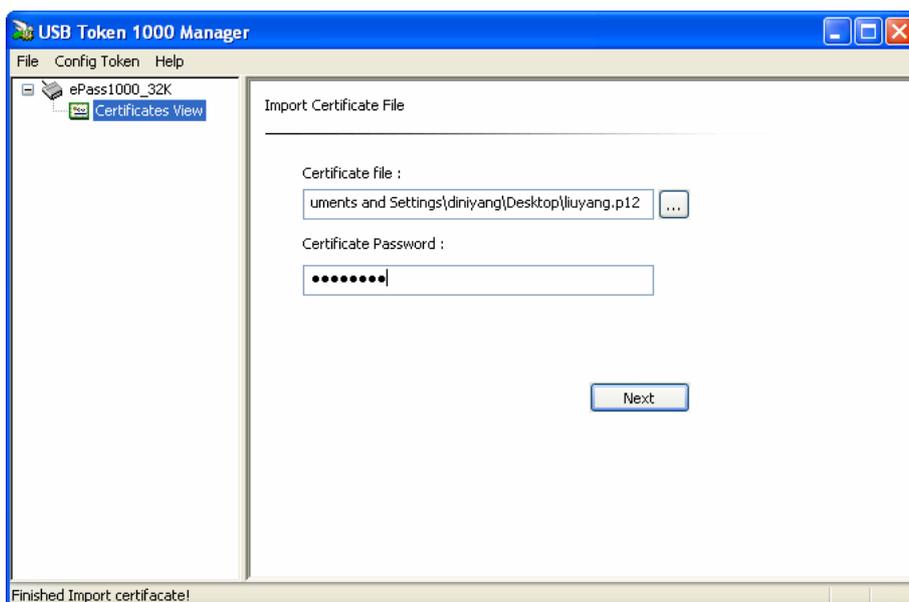


Figure 4.10

You may enter the complete path of the certificate file in the “Certificate file:” edit box. Or you may select a file by clicking the “...” button to the right of the “Certificate file:” edit box. You must also enter the password of the certificate file in the “Certificate Password” edit box. Then click “Next” to continue. The certificate file that you plan to import to ePass1000 must be *.cer, *.p7b, *.pfx or *.p12

The “Delete” button may be used to delete the selected certificate and corresponding private key. First click on the object label of the certificate/private key pair that you want deleted, then click the “Delete” button. Please see the figure below:

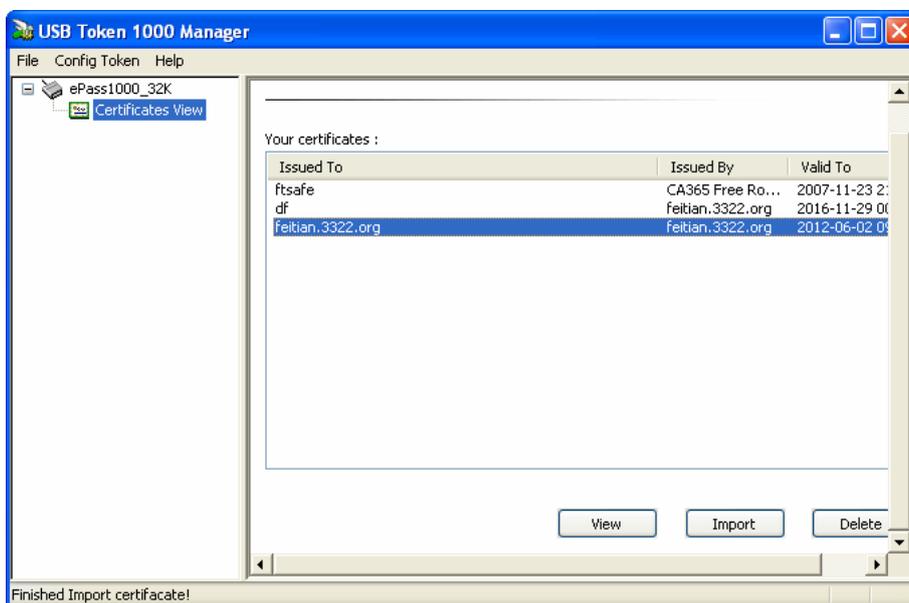


Figure 4.11

Chapter5

Integrating ePass1000 into Your PKI Application

ePass1000 security features can be of benefit to PKI developers that use Public Key cryptography to control access to enterprise intranets, Virtual Private Networks, SSL secured web sites and encrypted E-mail applications.

Note: PKI support of ePass1000 is currently only implemented for the Win32 platform.

In the following section we will discuss:

- ✓ The ePass1000 PKI Architecture
- ✓ The ePass1000 PKCS#11 Module
- ✓ The ePass1000 CSP for MS CAPI
- ✓ Multi-Token Application
- ✓ The ePass1000 PKI Application Guide
- ✓ Other Applications

5.1 The ePass1000 PKI Architecture

ePass1000 supports two industry PKI standards: PKCS#11 from RSA Labs and Microsoft's CAPI. Any application that is based on either of these standards may be integrated with ePass1000 without modification. Applications that comply with MS CAPI or PKCS #11 can use ePass1000 to securely store certificates, generate RSA/DSA key pairs, sign and verify digital signatures and encrypt or decrypt data. Figure 5.1 illustrates the architecture of ePass1000 PKI interface.

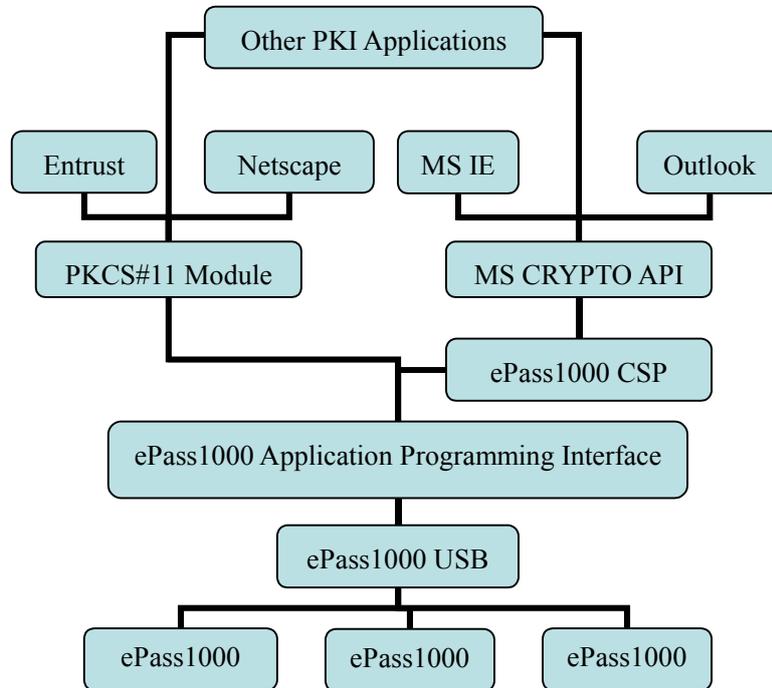


Figure 5.1

The ePass1000 driver manages the communication between ePass1000 and the computer. Applications communicate with the ePass1000 Application Interface. The API layer communicates with ePass1000 USB driver. The requirements of your application will determine into which layer you will program. Higher layers provide more complex functions and lower layers provide greater control of ePass1000. We do not suggest using several layers in one application.

5.2 The ePass1000 PKCS#11 Module

The exponential growth of the Internet fueled the demand for applications to secure transactions and communications over the public networks. With the growth in cryptographic applications came a corresponding need for these new applications to interoperate, to communicate with one another. RSA Laboratories created the Public Key Cryptography Standard (PKCS) with this purpose in mind.

The PKCS#11 standard is one of the PKCS standard families. The PKCS#11 standard (also known as "Cryptoki") is designed to ensure compatibility and interoperability between Public Key Cryptography implementations developed by different software or token vendors. It defines a common programming interface for Cryptoki tokens, such as ePass1000, and other cryptographic hardware.

Before developing applications with the ePass1000 PKCS#11 module, you need to be familiar with the PKCS#11 standard. You may find the latest version of the standard at <http://www.rsasecurity.com/rsalabs/pkcs/pkcs-11/index.html>.

The ePass1000 Cryptoki module supports PKCS#11 version 2.10. The interface is provided as a set of C language functions in a Win32 dll. The following files are required to develop the ePass1000 PKCS#11 applications:

File	SDK Path
cryptoki.h	\PKI\Include (Provided by RSA)
pkcs11.h	\PKI \Include (Provided by RSA)
pkcs11f.h	\PKI \Include (Provided by RSA)
pkcs11t.h	\PKI \Include (Provided by RSA)

ep1pk111.lib	\PKI \Lib
ep1pk111.dll	\PKI \Lib (should be under the system directory at run-time)
ep1kdl20.dll	\Private \Lib (should be under the system directory at run-time)

The ep1pk111.dll is the core dynamic library of ePass1000, which contains implementation of interface defined by RSA PKCS#11 specification. Include the cryptoki.h header file into your project if you need use this interface.

5.2.1 Supported PKCS#11 Object Class

The ePass1000 PKCS#11 implementation creates the following PKCS#11 objects:

Object Class	Description
CKO_DATA	For data structures defined by applications
CKO_SECRET_KEY	For symmetric keys
CKO_CERTIFICATE	For X.509 V3 certificates
CKO_PUBLIC_KEY	For RSA/ DSA public key.
CKO_PRIVATE_KEY	For RSA/DSA private key.

Applications may create and store inside ePass1000 all of the objects listed in the table above. Or the developer may choose to just create the objects at run-time. Developers should consider memory capacity limitations of ePass1000 as they select the objects for storage. Only the key and perpetual objects need be stored in ePass1000.

5.2.2 PKCS#11 Mechanisms Supported by ePass1000

The following table illustrates all mechanisms implemented by ePass1000 PKCS#11 module:

Mechanisms	Encrypt/ Decrypt	Sign/ Verify	Digest	Generate Key/Pair	Wrap/ Unwrap
CKM_RSA_PKCS_KEY_PAIR_GEN				√	
CKM_RSA_PKCS	√	√			
CKM_DSA_KEY_PAIR_GEN				√	
CKM_DSA		√			
CKM_RC2_KEY_GEN				√	
CKM_RC2_ECB	√				
CKM_RC2_CBC	√				
CKM_RC2_CBC_PAD	√				
CKM_RC4_KEY_GEN				√	
CKM_RC4	√				
CKM_DES_KEY_GEN				√	
CKM_DES_ECB	√				
CKM_DES_CBC	√				
CKM_DES_CBC_PAD	√				
CKM_DES3_KEY_GEN				√	
CKM_DES3_ECB	√				

CKM_DES3_CBC	√				
CKM_DES3_CBC_PAD	√				
CKM_MD2			√		
CKM_MD5			√		
CKM_SHA_1			√		
CKM_DH_PKCS_KEY_PAIR_GEN				√	
CKM_DH_PKCS_DERIVE				√	

The following table shows key sizes supported by ePass1000 PKCS#11 library.

Mechanisms	Key Sizes
CKM_RSA_KEY_PAIR_GEN	512-2048 bits
CKM_DSA_KEY_PAIR_GEN	508-1024 bits
CKM_RC2_KEY_GEN	1-128 bytes
CKM_RC4_KEY_GEN	1-256 bytes
CKM_DES_KEY_GEN	8 bytes
CKM_DES3_KEY_GEN	24 bytes
CKM_DH_PKCS_KEY_PAIR_GEN	128-2048 bits
CKM_DH_PKCS_DERIVE	1-128 bytes

5.2.3 ePass1000 PKCS#11 Function Library

Developers and hardware vendors should familiarize themselves with the PKCS #11 standards because they will need to call these functions to integrate ePass1000 PKCS #11 modules to their applications (See the RSA PKCS#11 specification at <http://www.rsasecurity.com>.)

The PKCS#11 specification is for a general case of Cryptoki hardware. Specific Cryptoki hardware implementation will vary depending on the characteristics of the hardware itself.

The ePass1000 PKCS#11 library also has some hardware specific variances from the standard:

- ✓ There are some functions defined in the PKCS #11 standard that are not implemented for the ePass1000 interface. If a non-implemented function is called, it will generate a return code CKR_FUNCTION_NOT_SUPPORT.

Note: ePass1000 is the “token” referred to in the PKCS#11 standard.

The PKCS #11 standard will refer to a “slot” for a card reader. Since the card reader function is integrated into ePass1000, this “slot” is a virtual device.

The following table lists functions of PKCS#11 specification:

Name	Description
General Purpose Functions	
C_Initialize	This function will initialize the library. The library must be initialized before another function may be called. The only exception is the C_GetFunctionList function.
C_Finalize	Call this function when the application is finished with the library.
C_GetInfo	Get information about the cryptoki library.

C_GetFunctionList	Get a list of pointers for all functions exported by the library.
Slot and Token Management Functions	
C_GetSlotList	Get a list of the slots.
C_GetSlotInfo	Get slot information.
C_GetTokenInfo	Get information about the token in the slot.
C_WaitForSlotEvent	Wait for a state change event in the slot, such as token insertion or removal.
C_GetMechanismList	Get a list of mechanism types supported by the library.
C_GetMechanismInfo	Get information on a mechanism supported by the library.
C_InitToken	Initialize the token.
C_InitPIN	Initialize the USER PIN.
C_SetPIN	Modify the PIN of the user who is currently logged into the token.
Session Management Functions	
C_OpenSession	Open a session between the application and the token.
C_CloseSession	Close a session between the application and the token.
C_CloseAllSessions	Close all sessions opened by the application.
C_GetSessionInfo	Get Information about a session.
C_GetOperationState	Obtain a copy of the cryptographic state of a session.
C_SetOperationState	Restore the state of a session from the bytes retrieved with the C_GetOperationState function.
C_Login	Log a user into the token.
C_Logout	Log a user out of the token.
Object Management Functions	
C_CreateObject	Create a new Cryptoki object.
C_CopyObject	Make a copy of the object.
C_DestroyObject	Erase an object.
C_GetObjectSize	Get size of an object in bytes.
C_GetAttributeValue	Get one or more attributes of an object.
C_SetAttributeValue	Modify one or more attributes of an object.
C_FindObjectsInit	Initialize an object search operation.
C_FindObjects	Continue an object search operation and return an object handle which matches the specified template.
C_FindObjectsFinal	Finish an object search operation.
Encryption Functions	
C_EncryptInit	Initialize an encryption operation.
C_Encrypt	Encrypt input data.
C_EncryptUpdate	Continue an encryption operation.
C_EncryptFinal	Finish an encryption operation.

Decryption Functions	
C_DecryptInit	Initialize a decryption operation.
C_Decrypt	Decrypt input data.
C_DecryptUpdate	Continue a decryption operation.
C_DecryptFinal	Finish a decryption operation.
Message Digesting Functions	
C_DigestInit	Initialize a message digesting operation.
C_Digest	Digest the input data.
C_DigestUpdate	Continue a message digesting operation.
C_DigestKey	Continue a message digesting operation by digesting a secret key.
C_DigestFinal	Finish a message digesting operation and return the digest of the message.
Signing and MACing Functions	
C_SignInit	Initialize a signature operation.
C_Sign	Sign the input data.
C_SignUpdate	Continue a data signing operation.
C_SignFinal	Finish a data signing operation.
C_SignRecoverInit	Initialize a signature operation where the data can be recovered from the signature.
C_SignRecover	Sign data in a single operation where the data can be recovered from the signature.
Verifying Signatures and MAC functions	
C_VerifyInit	Initialize a verification operation.
C_Verify	Verify a signature in a single operation.
C_VerifyUpdate	Continue a multi-part verification.
C_VerifyFinal	Finish a verification operation.
C_VerifyRecoverInit	Initialize signature verification where the data is recovered from the signature.
C_VerifyRecover	Verify a signature in a single-part operation where the data is recovered from the signature.
Dual-Function Cryptographic Functions	
C_DigestEncryptUpdate	Continue a multi-part digesting and encryption operation.
C_DecryptDigestUpdate	Continue a multi-part decryption and digesting operation.
C_SignEncryptUpdate	Continue a multi-part signing and encryption operation.
C_DecryptVerifyUpdate	Continue a multi-part decryption and verification operation.
Key Management Functions	
C_GenerateKey	Generate a secret key and create a new key object.
C_GenerateKeyPair	Generate a pair of asymmetric keys and create new public and private key objects.

C_DeriveKey	Derive a private or secret key.
C_WrapKey	Wrap a private or secret key.
C_UnwrapKey	Unwrap a wrapped key and create a new key object.
Random Number Generation Functions	
C_SeedRandom	Add a seed into the token's random number generator.
C_GenerateRandom	Generate random strings with the token's random number generator.
Parallel Function Management Functions	
C_GetFunctionStatus	It is a legacy function and does nothing.
C_CancelFunction	It is a legacy function and does nothing.

5.3 The ePass1000 CSP for MS CAPI

CryptoAPI from Microsoft is a common Application Programming Interface (API) to add cryptographic security to applications. The standard includes functions for encoding and decoding, hashing, encrypting and decrypting data, digital certificate based authentication and certificate management. Encrypting and decrypting are provided with both session and public/private key pairs.

The following figure illustrates the CAPI architecture:

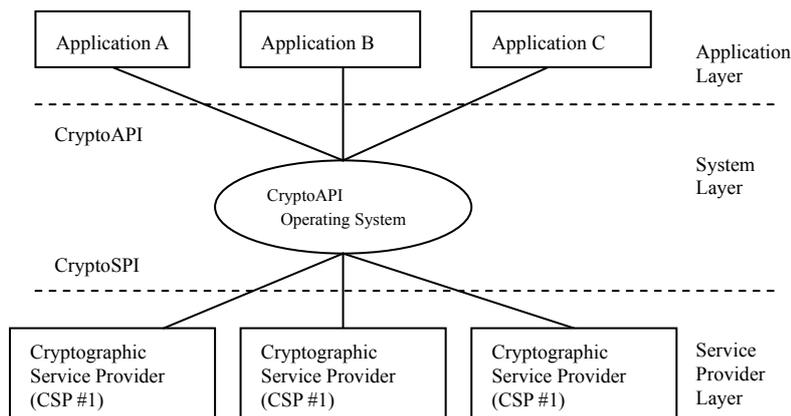


Figure 5.2

The above figure shows that in the MS CAPI architecture the CSP provides the cryptographic functions. Applications do not call the CSP functions directly. The CryptoAPI coordinates the interaction between the applications and the CSP. In a CryptoAPI environment, applications can access ePass1000 via the ePass1000 Cryptographic Service Provider (CSP).

For more information on MS CAPI, please visit the Microsoft Software Developer Network (MSDN) at <http://www.microsoft.com>.

The ePass1000 CSP is a PROV_RSA_FULL type provider. Applications should invoke CAPI to establish communication with ePass1000. CAPI will interact with the ePass1000 CSP for your application. Each vendor's CSP implementation will vary depending on the characteristics of the specific token.

Below is a CSP function list for ePass1000:

Names	Descriptions
Connection Functions	
CPAcquireContext	Create a context and initialize the access to CSP. You may and must specify the CSP here. This function acquires a handle to the key container.
CPGetProvParam	This function returns information about the CSP.

CPReleaseContext	This function releases a context created by CPAcquireContext.
CPSetProvParam	This function customizes the operations of a CSP.
Key Generation and Exchange Functions	
CPDeriveKey	This function generates a cryptographic session key using a base data hash. It guarantees that all keys will be identical if they are generated from the same base data and algorithms. The base data can be a password or other user-supplied data.
CPDestroyKey	This function releases the key handle. After it is released, the key handle becomes invalid and the key can no longer be used.
CPDuplicateKey	This function makes an exact copy of a key.
CPExportKey	This function exports cryptographic keys from the container of the CSP.
CPGenKey	This function generates a random cryptographic key or key pair.
CPGenRandom	This function fills a buffer with random bytes.
CPGetKeyParam	This function gets the attributes of an encryption key.
CPGetUserKey	This function retrieves the handle of one of the permanent key pairs in the CSP container.
CPImportKey	This function transfers a cryptographic key from a key blob to a CSP key container.
CPSetKeyParam	This function customizes the operations of a key.
Data Encryption Functions	
CPDecrypt	This function decrypts data previously encrypted with the CPEncrypt function. Optionally, the application can specify that the decrypted data be hashed.
CPEncrypt	This function encrypts data. Optionally, the application can specify that a hash of the plaintext data is to be generated.
Hashing and Digital Signature Functions	
CPCreateHash	This function creates a hash object and initiates the hashing of a stream of data.
CPDestroyHash	This function destroys the hash object handle.
CPDuplicateHash	This function makes an exact copy of a hash and its state.
CPGetHashParam	This function retrieves data about the operations of a hash object. The actual hash value can be obtained by using this function.
CPHashData	This function feeds data into a specified hash object.
CPHashSessionKey	This function feeds a cryptographic key to a specified hash object. This allows a key to be hashed without the application having to access the key material
CPSetHashParam	This function customizes the operations of a hash object.

CPSignHash	This function signs a hash object.
CPVerifySignature	This function verifies the digital signature.

5.4 Multi-Token Application

ePass1000 supports multi-token applications – applications that require simultaneous connection of two or more tokens. You may select a single ePass1000 token to store certificate information, manage certificates or store other data used by PKCS#11 or MS CSP PKI applications. For example, you can apply certificate information to specific attached token, or you can allow the browser to view certificate information from several attached tokens.

Note: Smart card subsystem of Windows platform supports at most 10 card readers.

5.5 The ePass1000 PKI Application Guide

The ePass1000 product was designed to be seamlessly integrated with existing PKI applications. PKI application developers can integrate ePass1000 system into PKI applications by configuring the relevant server, without any programming. This chapter will discuss following topics:

- ✓ Configure Certificate Authority
- ✓ Configure SSL Encrypted Web
- ✓ Apply for Digital Certificate with ePass1000 token
- ✓ Access SSL Encrypted Web with ePass1000 token
- ✓ Receive/Send Digital Signed and Encrypted E-mail with ePass1000
- ✓ Windows2000 Smart Card Logon with ePass1000
- ✓ VPN Remote Logon with ePass1000

5.5.1 Configure Certificate Authority

CA stands for Certificate Authority. The CA is the core of PKI Applications and all PKI applications are based on the CA. Windows2000 includes built-in support for PKI applications. After proper configuration you can perform smart card log on, workstation lock, VPN remote access, access SSL encrypted web and other operations with Windows2000. We will take the CA of Windows2000 server or advanced server as an example to discuss how to configure CA.

Install Certification Authority

Windows 2000 Server does not install certificate service by default because Windows2000 Server cannot change the name of computers after installing the certificate service. When the user needs to install certificate service to Windows2000 Server, he should choose **Windows Components** from **Add/Remove programs** to install certificate service.

Install Certificate Service to Windows2000 Server:

1. Login Windows2000 system with the administrative privilege.
2. From **Start** menu, select **Settings** then **Control Panel**.
3. Double click **Add/Remove Programs** icon. See Figure 5.3.

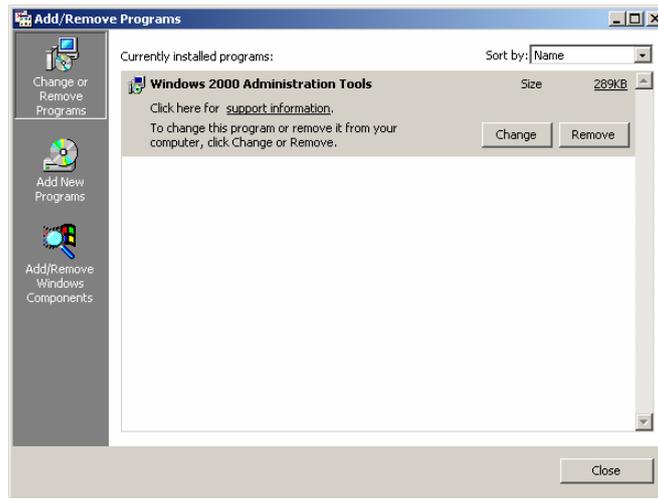


Figure 5.3

4. Click **Add/Remove Windows Components** button. The Windows Component Wizard will start. Choose your targeted service or component. See Figure 5.4.

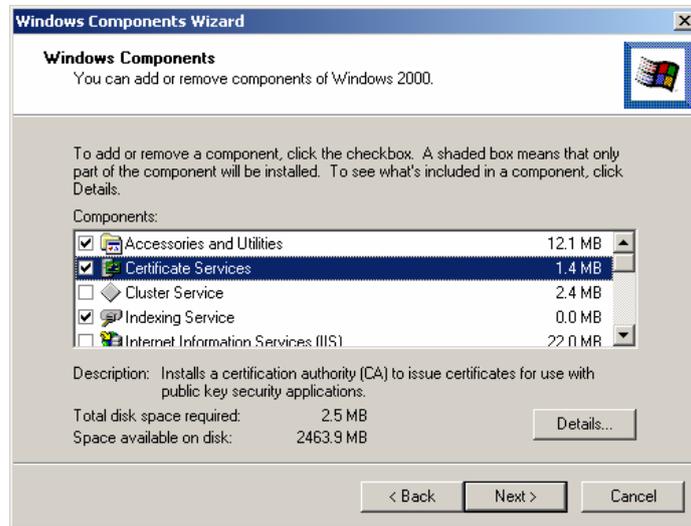


Figure 5.4

5. Select **Certificate Services** button from the component list. After installing certificate service, this computer becomes the CA server and you cannot change its name, add it to other domain, or delete it from the current domain. So you must make sure of the stability of this computer before you install the certificate service.
6. Click **Next** button to set Certification Authority Type. Choose the CA type you require. Then click **Next**. The figure below lists the optional CA types: (See Figure 5.5)

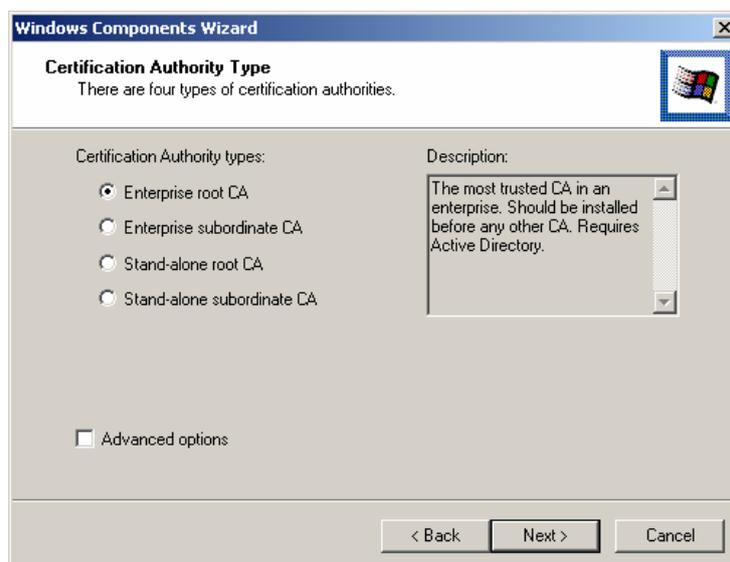


Figure 5.5

Enterprise Root CA: If this CA will issue the certificates to the entity in enterprise Active Directory domain you must install this CA. The root CA will be registered in the directory, and all computers in your enterprise using that directory will automatically trust the root CA. It is good security practice to limit the root CA to issue certificates to subordinate CAs only, or to issue only a few special purpose certificates if the hardware space is enough. This means you want to install an enterprise subordinate after you finish installing the root. However, you can choose only the root CA. If there is no CA inside the enterprise, you must install Root CA.

Enterprise Subordinate CA: If there is a Root CA already in the enterprise, you should install Enterprise Subordinate CA to issue the certificates to each entity in the enterprise. It requires Active Directory as well.

Stand-alone Root CA: You should install a stand-alone root CA if you will issue certificates outside of the enterprise network. For example, you can issue certificates to your customers so they can access your Web site from outside your enterprise. A stand-alone CA is the root of a CA trust hierarchy.

Stand-alone Subordinate CA: It operates with an existing CA trust hierarchy. This could be an external, commercial CA or a Stand-alone Root CA. You should set up a stand-alone subordinate CA when you will issue certificates to entities outside a corporation.

For Windows 2000 smart card logon, you should select an enterprise CA. In Figure 5.5 we selected Enterprise Root CA.

The Windows 2000 Certificate Services use the default encryption system to provide the security policy. If you need to configure the advanced CA settings (such as CSP, digital signature, the Hash function for information integrity, the length of key size and key types, etc.), enable **Advanced options**. Then click **Next** to set Public/Private Key. See Figure 5.6.

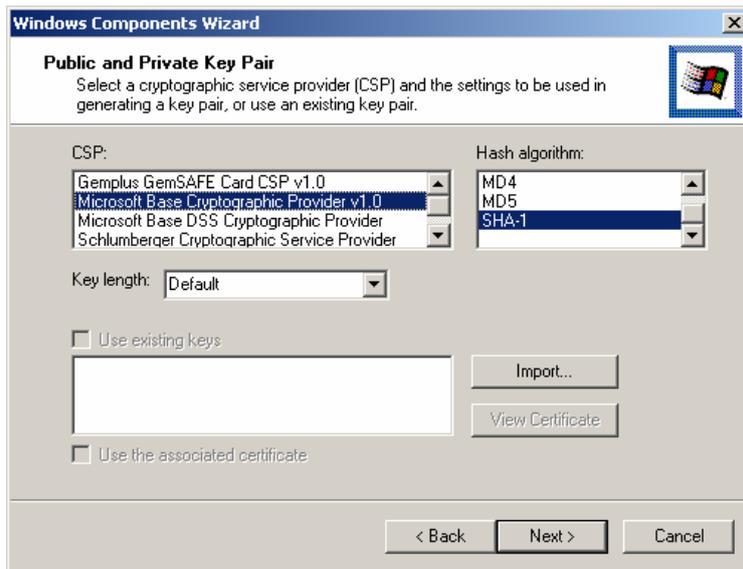


Figure 5.6

You can change the default encryption function here, such as CSP and Hash algorithm. The listed encryption functions are supported by the software and hardware of your computer.

You can change the key length from the option list. In general, longer the keys will mean more secure data. Longer keys also mean that the encryption/decryption processes will take longer than for shorter keys. If you choose “**Default**” for Key length, the system will set the key length automatically according to the selected CSP. We recommend that users select the longest key length possible. Some hardware will not support longer key lengths.

If you want to use the existing keys in the system, choose **Use existing keys** and click **Import** button.

Then click **Next** to continue.

7. Enter the appropriate identifying information for your CA, see Figure 5.7:

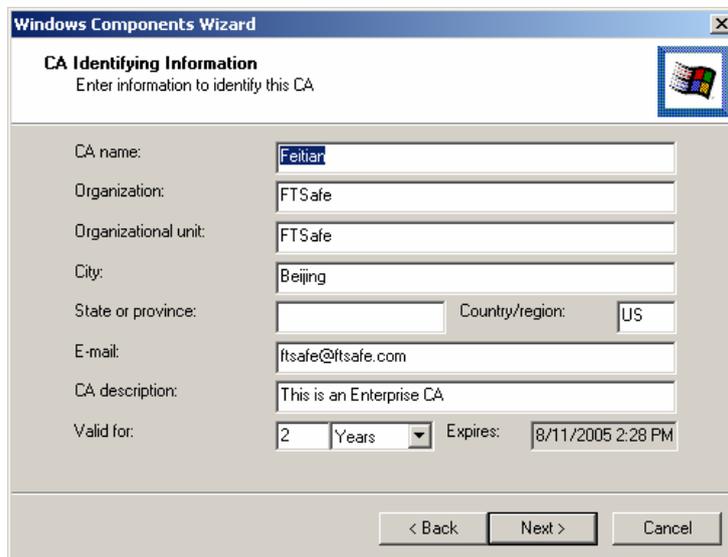


Figure 5.7

You must input a name for the CA in the **CA name** field. You will use this name later when you identify the CA to the Certificate Services.

The CA name will identify the Enterprise CA to Active directory and any stand-alone CA. The expiration time for a root CA should be long enough, at least greater than that for any subordinate CA. The administrator should take into account the security and work required to manage the system when setting the expiration time. When root CA server expires the administrator must refresh all the trust relationship.

Then click “Next” to continue.

8. Specify the storage location of certificate database, certificate services setting information, certificate revocation list and certificate database log file. See Figure 5.8.

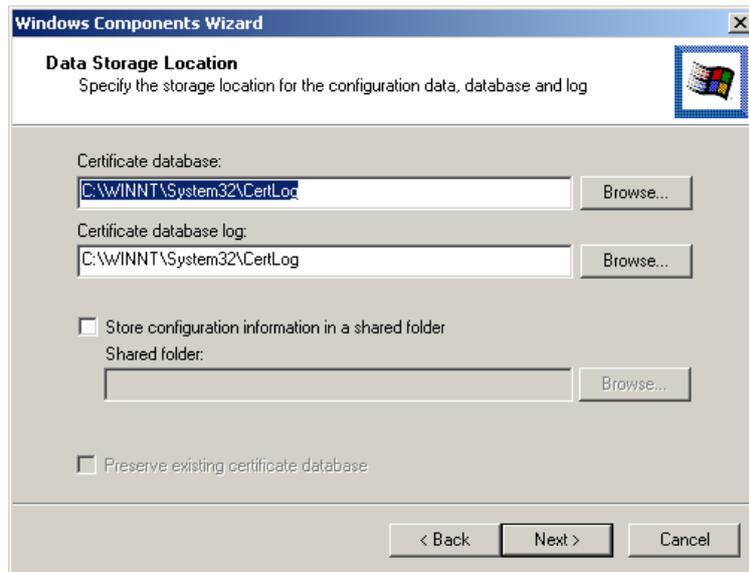


Figure 5.8

If it is an enterprise CA, it will keep some of its setting and attribute information in the domain (domain controller).

If you did not set Certificate Services in the domain controller computer, choose the **shared folder** option and input a shared folder path to specify the storage place for the CA (users can specify a shared folder – this way a client outside of the domain can access the certificate revocation list).

Click **Next** to continue.

9. If you are installing a subordinate CA, you will see “CA Certificate Request” Window (If you are not installing subordinate CA, please skip to step 10). Subordinate CA requests certificate information directly from root CA, here we need to specify the subordinate CA should request the certificate information from the root CA on which Windows2000 computer. You may send the request online or save your request to a file to get the certification information from the CA. If you choose **Send the request directly to a CA already on the network** you need to specify the CA computer name and the parent CA. If you choose **Save the request to a file** please specify the file path. See 5.9.

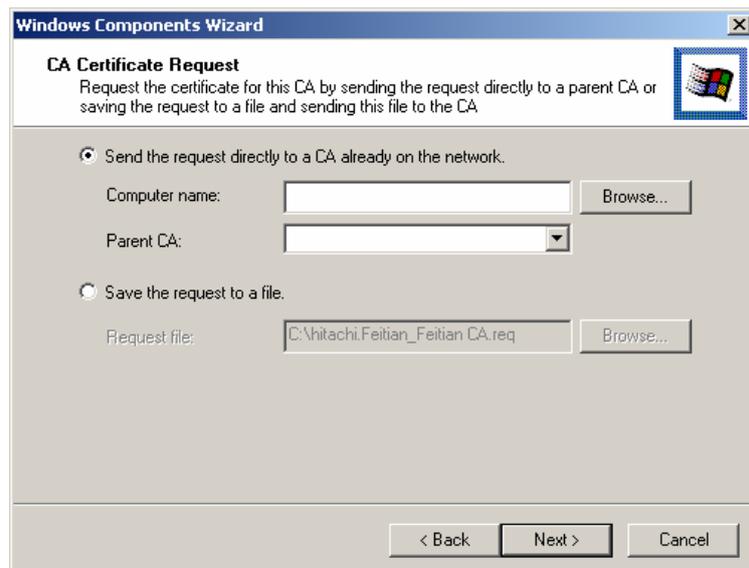


Figure 5.9

Click “Next” to continue.

10. If IIS is running, a message will prompt you to stop the service. Click **OK** to stop IIS. You must stop IIS to install

the Web components. If you do not have IIS installed, you will not see this message. See Figure 5.10.



Figure 5.10

11. The system begins to install the related components and programs, see Figure 5.11.

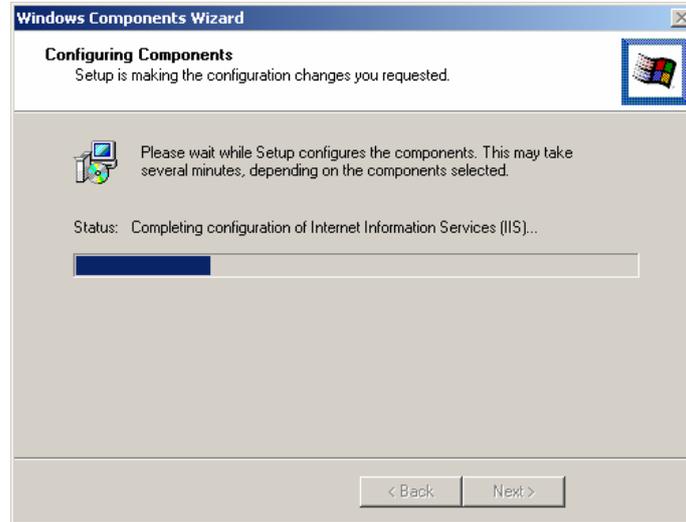


Figure 5.11

The file folder `%SystemRoot%\system32\CerSrv\CertEnroll` is shared to allow the clients to access the information under this directory and check revocation list. If this folder is not shared, the client computer may not work properly.

12. Now the Certificate Services is successfully installed to your Windows 2000 computer. You may start and manage Certificate Services from **Start Menu > Program > Administrative Tools > Certificate Authority**.

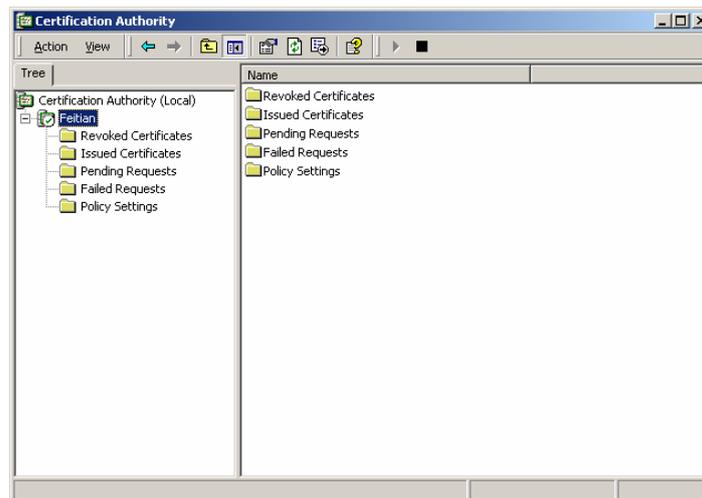


Figure 5.12

5.5.2 Install Root Certificate

Before you request certificate from one CA you must install the root certificate of this CA. Without the root certificate it is impossible to verify the certificate issued by this CA, and you cannot request certificate from this CA either.

The user must get his root certificate from the enterprise root CA, then the system can recognize the certificates from this CA. Below is a sample to show how to install the root certificate.

1. Type the certificate server path in Internet Explorer. (Such as <http://epassca/certsrv> where ePassCA is the CA certificate server computer name in the enterprise.) Then press “Enter” key.

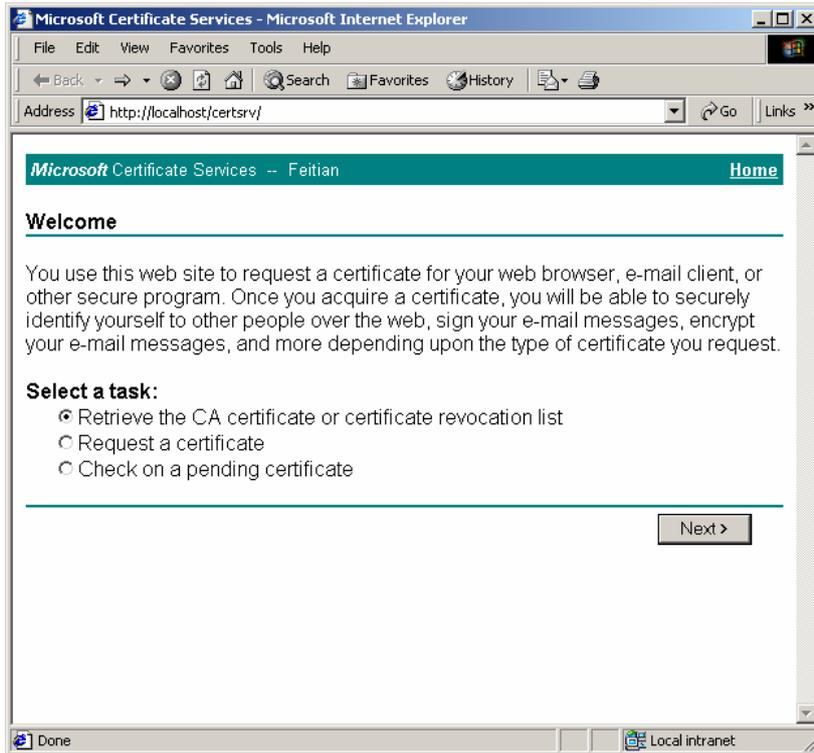


Figure 5.13

To get the root certificate of this CA, please select **Retrieve the CA certificate or certificate revocation list**, see Figure 5.13. Then click **Next** to continue.

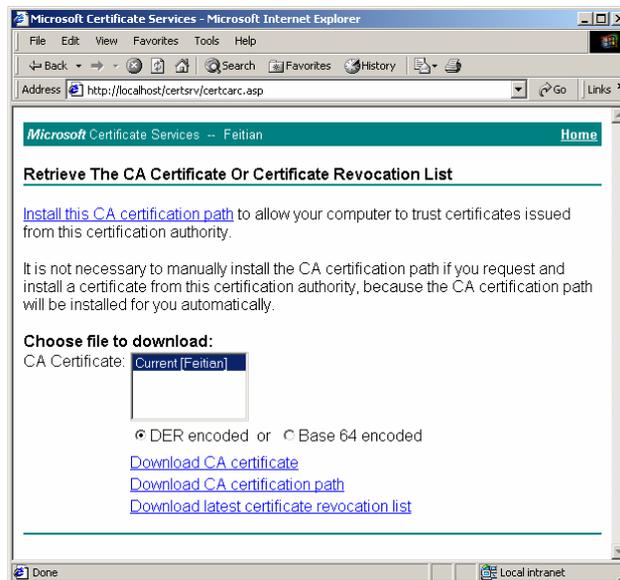


Figure 5.14

2. You will see some links to install CA certificate or download CA certificate on the webpage.

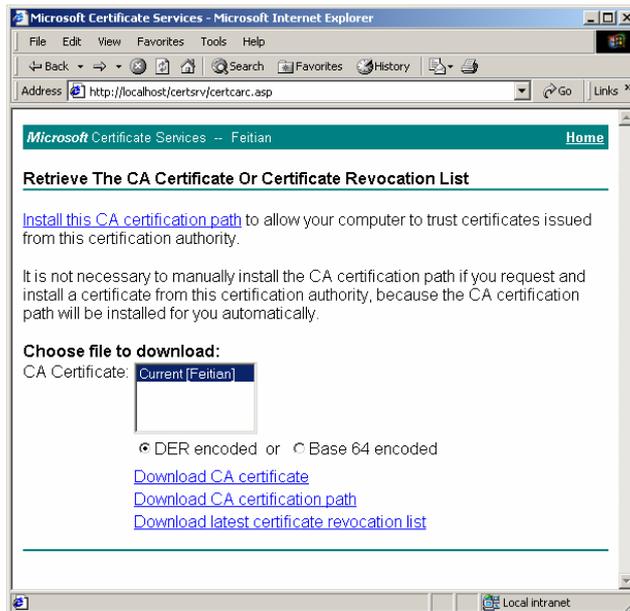


Figure 5.15

If you want to trust **all** the certificates issued by this CA, click **Install this CA certification path**. Then this Windows 2000 computer may perform identity verification and other security functions with the certificates issued from this CA. See Figure 5.16.

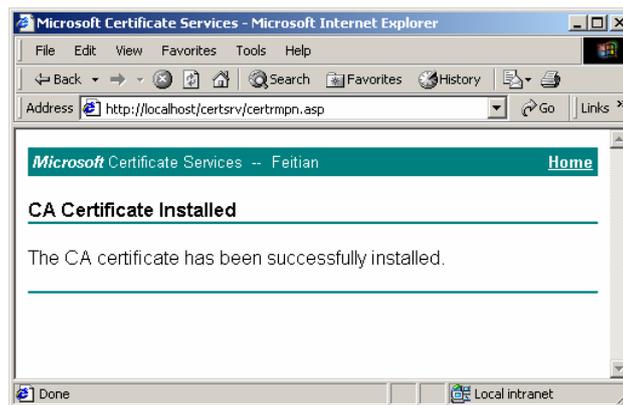


Figure 5.16

You may choose the link **Download CA certificate** to get the certificate information manually. You may ask the CA to package the certificate information in DER encoded format or Base 64 encoded format file. Then you may download the certificate, certificate path or certificate revocation list with IE and import the file at a later date.

3. After you choose the encode format and click the **Download CA certificate** link, the system will download the CA certificate to your Windows2000 computer in your chosen format. See Figure 5.17.

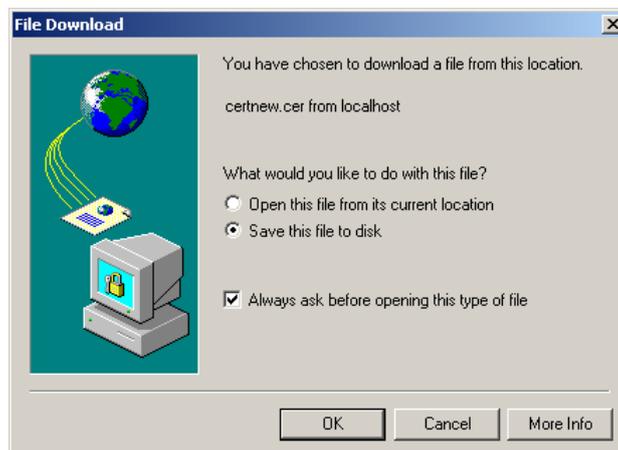


Figure 5.17

4. If you want to view this file certnew.cer (the certificate export file), choose **Open this file from its current location**, and then click **OK**. See Figure 5.18.



Figure 5.18

5. You may view the certificate information here, if it is correct, press **Install Certificate...** button in table General. The system will invoke the certificate import wizard to install the certificate to your computer.



Figure 5.19

In the Certificate Import wizard, follow the instruction to finish installation.

You may download the certificate revocation list in the same way.

5.5.3 Configuring SSL Encrypted Web

Internet Information Services (IIS) is installed on Windows 2000 Server by default. IIS is one of the services offered by the Windows 2000 system. IIS provides several Internet services, such as WWW, FTP, Gopher, etc. You can install IIS

from **Control Panel > Add/Remove Programs > Add/Remove Windows Components**, then select **IIS**. Then follow the on-screen instructions to install. See Figure 5.20.

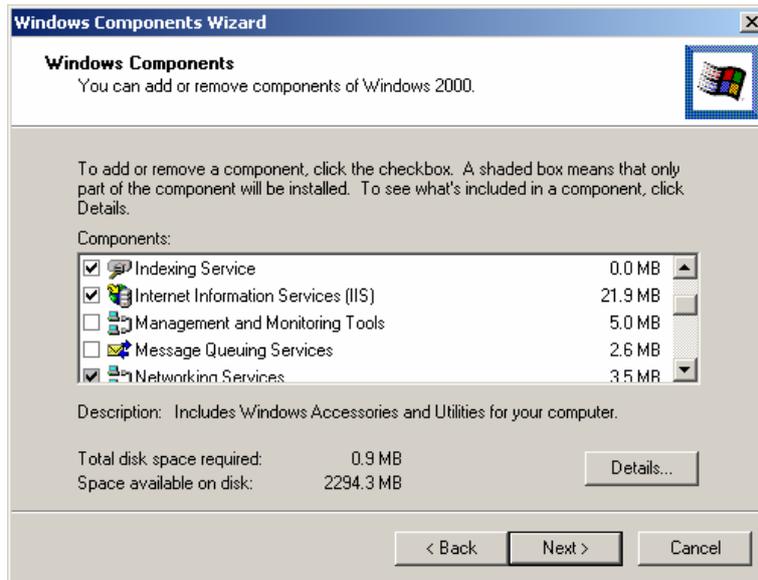


Figure 5.20

If IIS has been installed already and activated, you can run IIS tool from **Start Menu > Program > Administrative Tool > Internet service administrative tool**. See Figure 5.21.

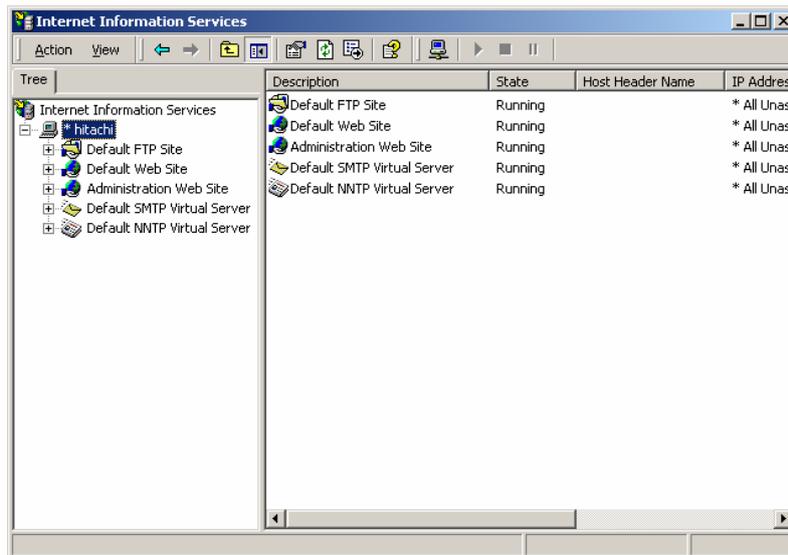


Figure 5.21

We need to set IIS system to enable the SSL function. See the following steps:

1. Login Windows 2000 Server computer with administrative privilege.
2. From **Start menu > Programs > Administrative Tools > Internet Services Manager** to start IIS service managing tool.

Expand “Internet Information Services” node in the console, then right click the computer name, select **Properties** (Please do not select the Web station or other IIS node).

Then the system will activate the IIS Properties Window, see Figure 5.22.

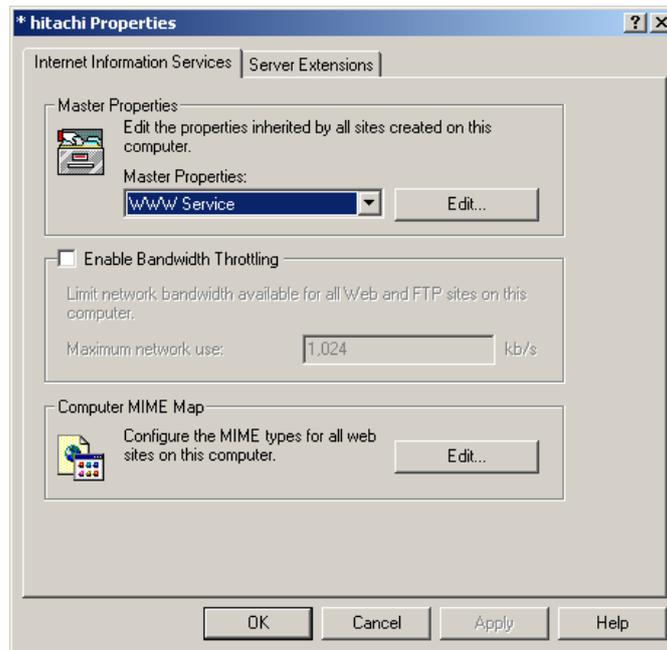


Figure 5.22

3. Then select **WWW Service** for “Master Properties”, and press **Edit** button to the right of the list box.
4. See the “WWW Service Master Properties” window in Figure 5.23, and select “Directory Security” page.

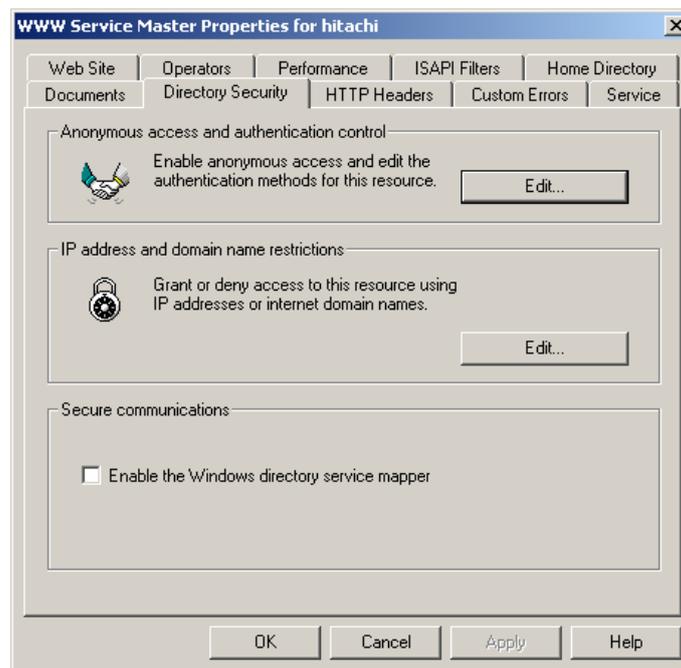


Figure 5.23

Click **Enable the Windows directory service mapper** for Secure communications.

If we **Enable the Windows directory service mapper**, IIS will ask the Active Directory domain controller to deal the mapping relationship between the certificate and the account. You can set this function only in IIS Master Properties window.

If you let Active Directory domain controller to deal the mapping you may use the certificate issued by your enterprise CA to log into the enterprise web site. By default Windows 2000 may automatically one-to-one map the certificate and account. You may access the web with this map.

5. Here we will discuss how to set the security functions for one single Web site. If you do not want to use Windows 2000 Active Directory mapping function (That is to say you do not want to enable the Windows directory service

mapper in last step), you may jump to this step.

In IIS we may set to manage multiple IIS servers (including several WWW Servers, multiple FTP Servers, or other IIS servers). In the above we set the secure communication for the whole IIS (Home Directory Security Settings). Now let us see how to set and manage the security features of one web site inside the IIS.

Back to the console and right click the service node (such as Default Web Site) that you want to set for and choose **Property**.

The system will open the IIS Properties window, please click table “Directory Security”, see Figure 5.24.

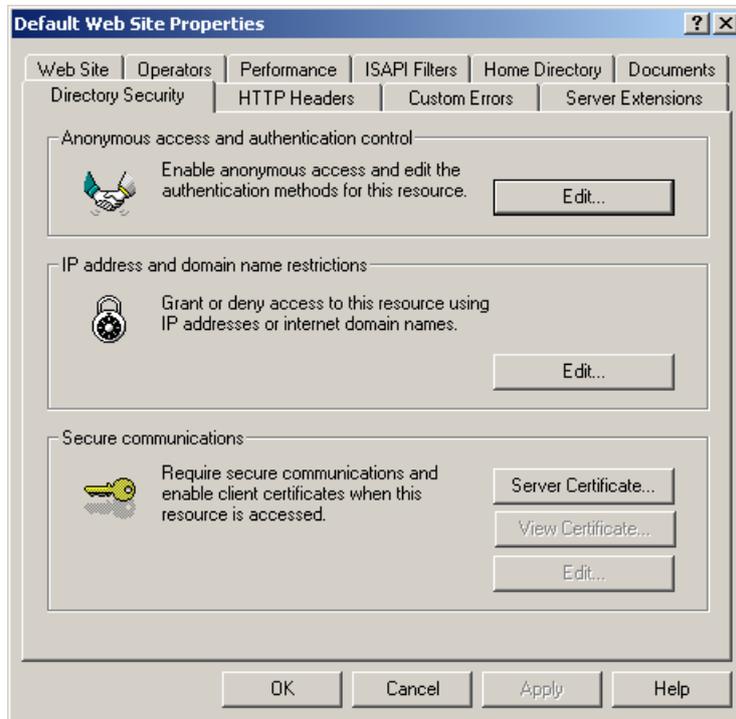


Figure 5.24

To activate IIS function you must request one server certificate first to offer the certificate verification service.

The **Edit** button under **Secure communications** is unavailable. This is the case until you have installed a Web server certificate. To install the Web server certificate please press “Server Certificate” button.

6. Then Web Server Certificate Wizard will start, and guide you to install the server certificate. See Figure 5.25.



Figure 5.25

7. Click “Next” to continue. You need to specify the server certificate, if you have never installed server certificate before, you must **Create a new certificate**. If you once requested a sever certificate and want to use the existing

certificate you may **Assign an existing certificate** or **Import a certificate from a Key Manager backup file** to install the existing certificate.

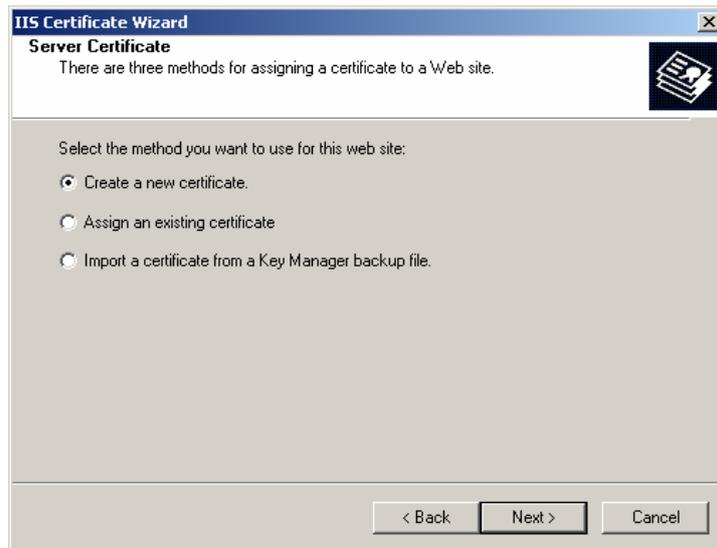


Figure 5.26

Here we choose “Create a new certificate” and click **Next** to continue.

8. The system will ask you to decide whether you plan to prepare the certificate request first, and send this request to the CA later or send the certificate request to the CA immediately to get the certificate information.

You send the certificate request directly to the CA and get the certificate information by **Send the request directly to a CA on the network**; or save the certificate request to a file and send the request to the CA later by **Save the request to a file**.

9. If the certificate should be issued by a commercial CA outside of the enterprise, you need to save your request to a file, and the commercial CA will verify the request file and issue one certificate to you. In general the CAs that you send the request directly to on the network are local CAs, or enterprise CAs.

We **Save the request to a file** here. And click **Next** to continue.

10. In the **Name and Security Settings** window set the name for the certificate and specify the key bit length. Here we set **Bit Length** to 1024.

And you may set this certificate to Server Gated Cryptography (SGC) certificate. SGC certificate is for export versions only. See Figure 5.27.



Figure 5.27

Click **Next** to continue.

11. Now you need to input the organization information for your certificate. See Figure 5.28. Then **Next** to continue.

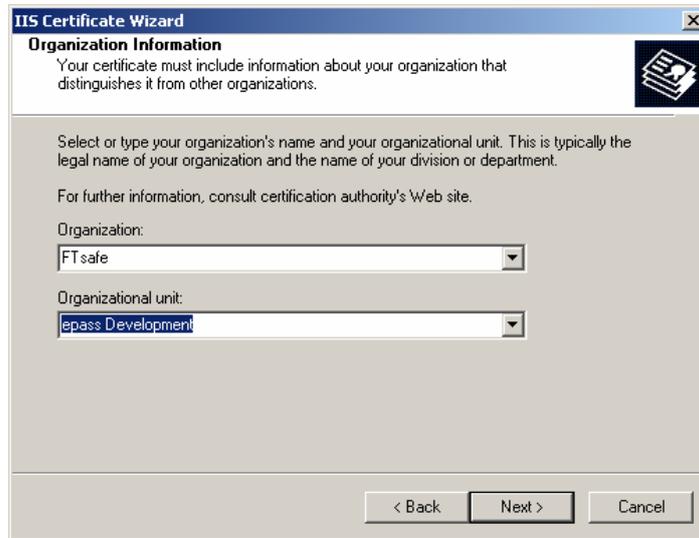


Figure 5.28

12. Input your site's common name. If the server is on the Internet, use a valid DNS name; if the server is on the intranet, you may prefer to use the computer's NetBIOS name, see Figure 5.29. Click **Next** to continue.

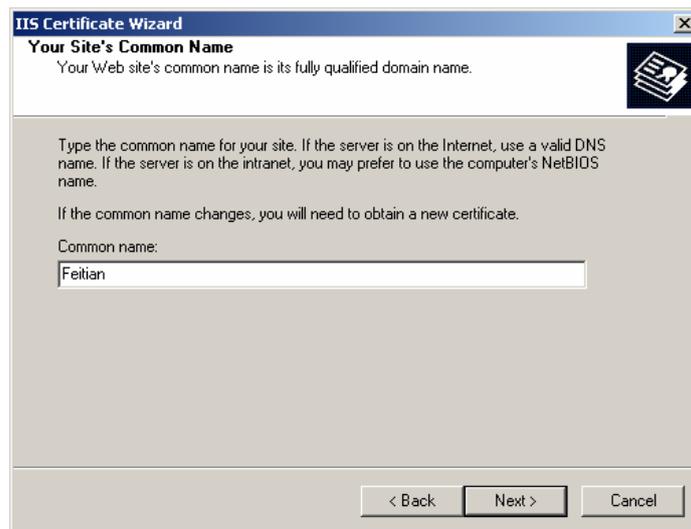


Figure 5.29

13. Enter your location information, and click **Next**. See Figure 5.30.

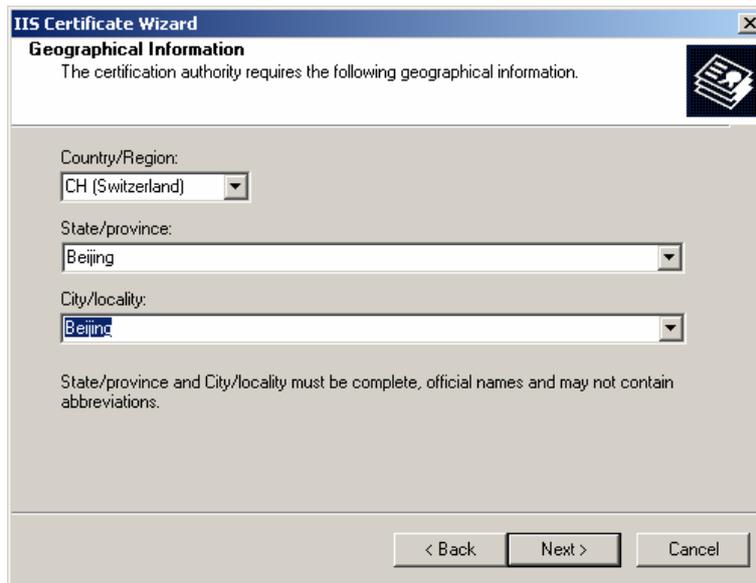


Figure 5.30

14. Enter a file name for the certificate request file. See Figure 5.31.

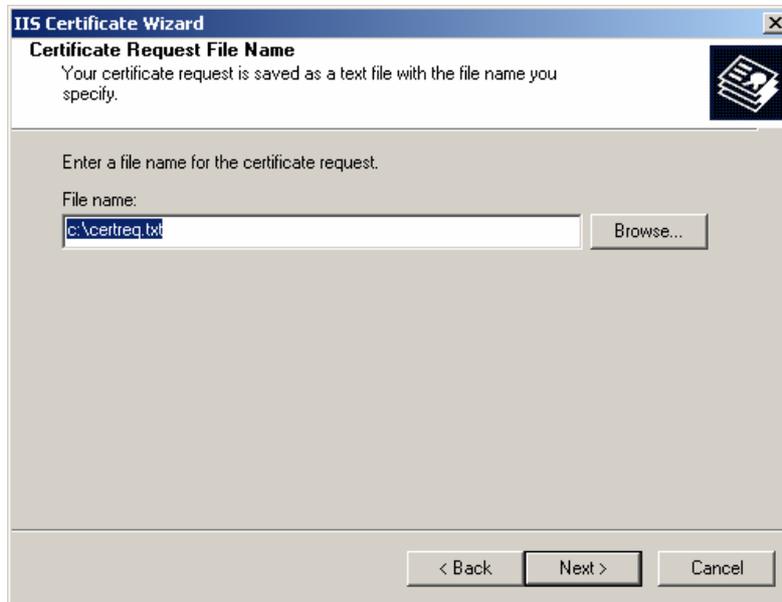


Figure 5.31

15. The system will display the request file summary. Please check the information. Click **Next** to continue. See Figure 5.32.

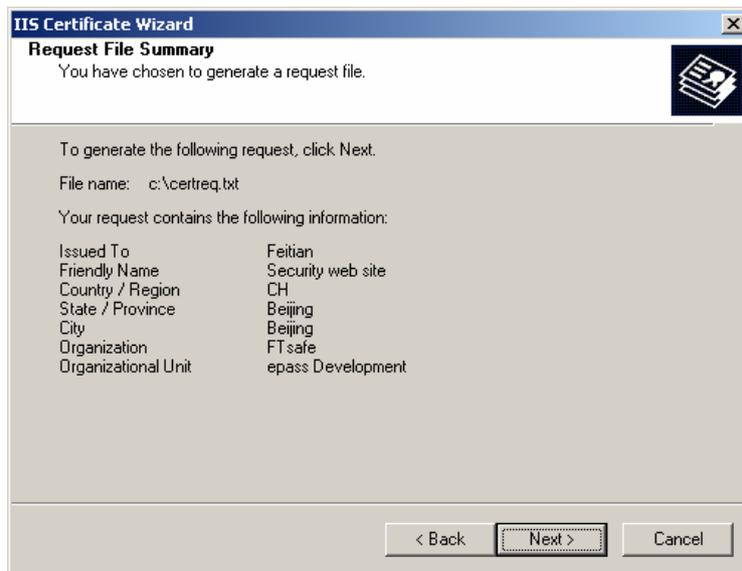


Figure 5.32

16. Click the **Finish** button. The computer has already saved the certificate request. You may take the certificate request file to the CA and obtain your certificate.
17. Open a browser and go to `http:// YourWebServerName /certsrv/`, select **Request a Certificate**. Click **Next**. See Figure 5.33.

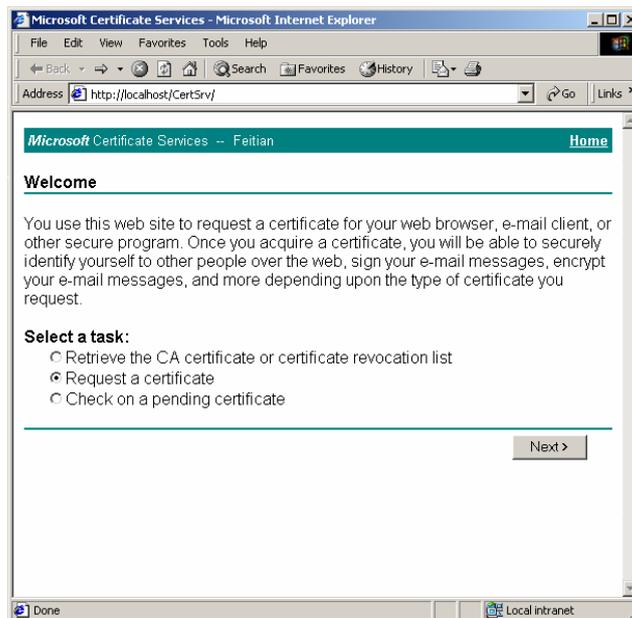


Figure 5.33

18. Select **Advanced Request**. Click **Next**.

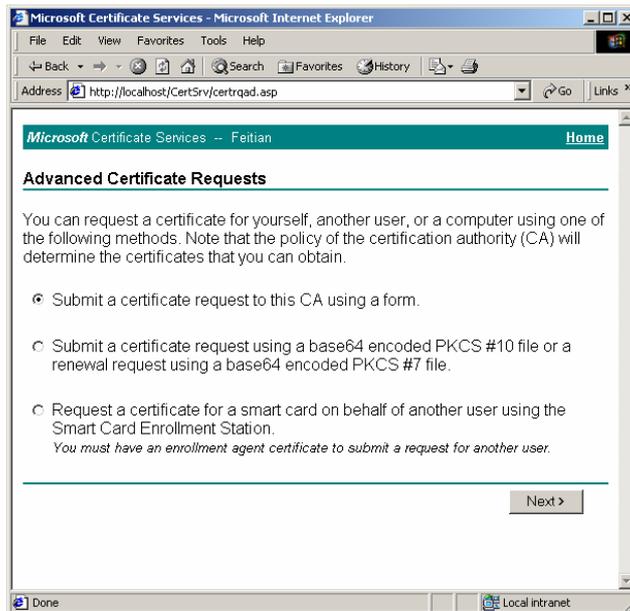


Figure 5.34

19. Select **Submit a Certificate Request using a Base64 encoded PKCS#10 file or a renewal request using a Based64 encoded PKCS#7 file**, and click **Next** as in Figure 5.34.
20. Open the request file that you created in the former step in Notepad. Copy the contents of the document and paste into the Web form's **Base64 Encoded Certificate Request** text box. See Figure 5.35.

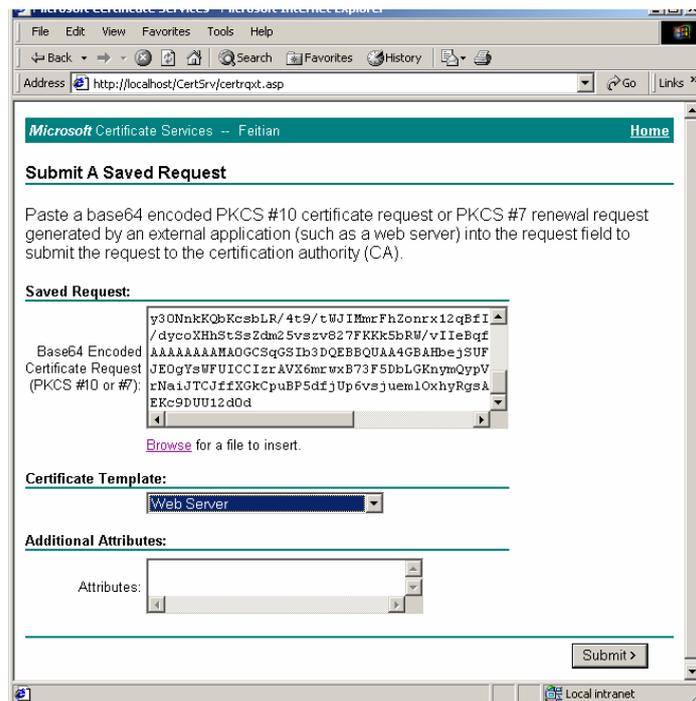


Figure 5.35

21. The certificate request is pending. Please come back later to obtain the certificate.

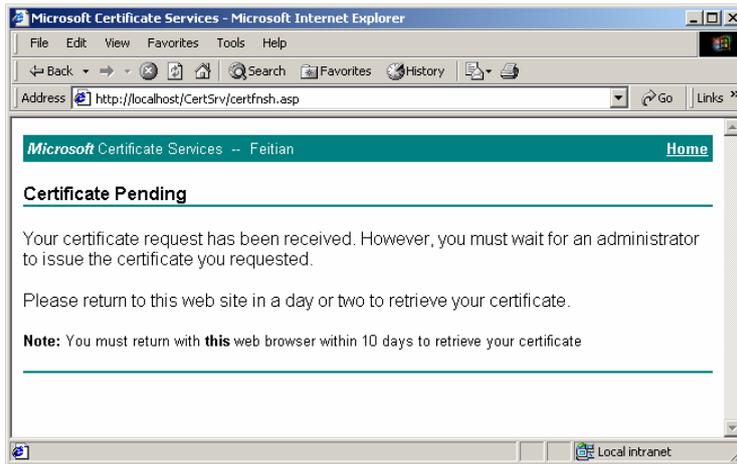


Figure 5.36

22. When you are informed that your certificate is issued, please come back to the certificate server to **Check on a pending certificate**. See Figure 5.37.

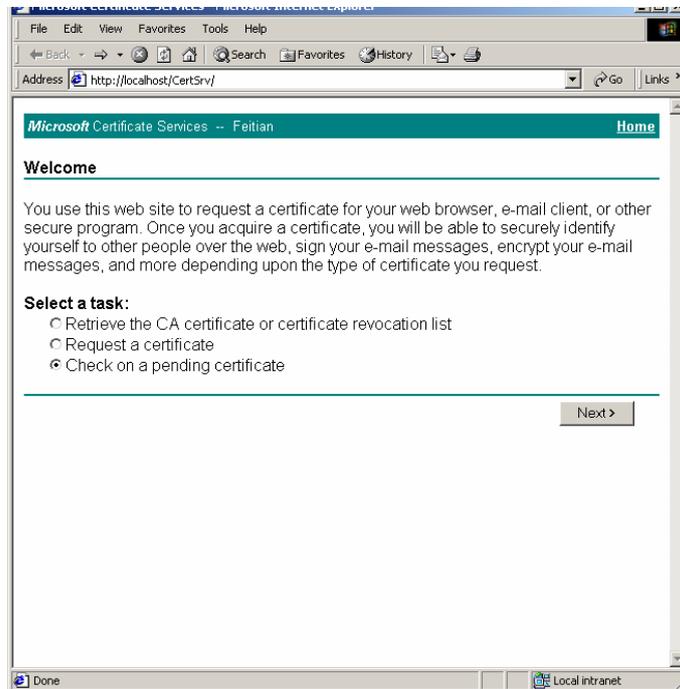


Figure 5.37

23. Select the certificate request according to the request time, and click **Next** to continue. See Figure 5.38.

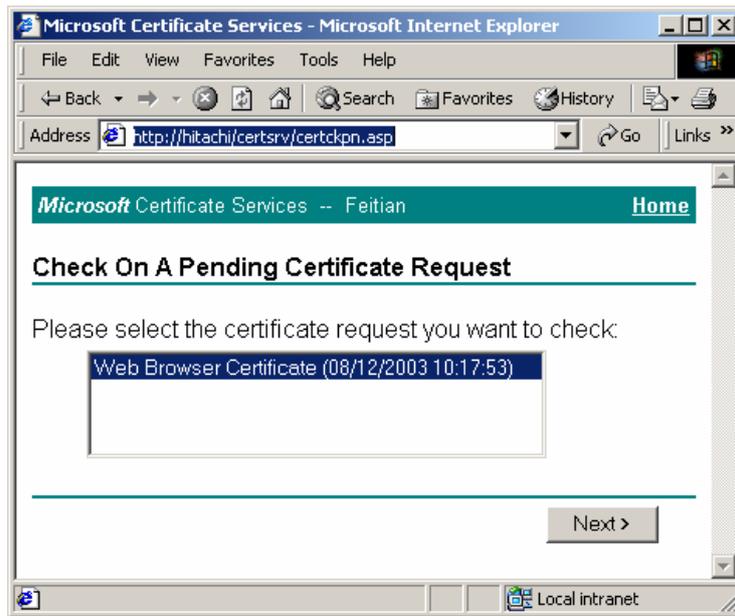


Figure 5.38

24. Click **Download CA certificate** to download your certificate. See Figure 5.39.

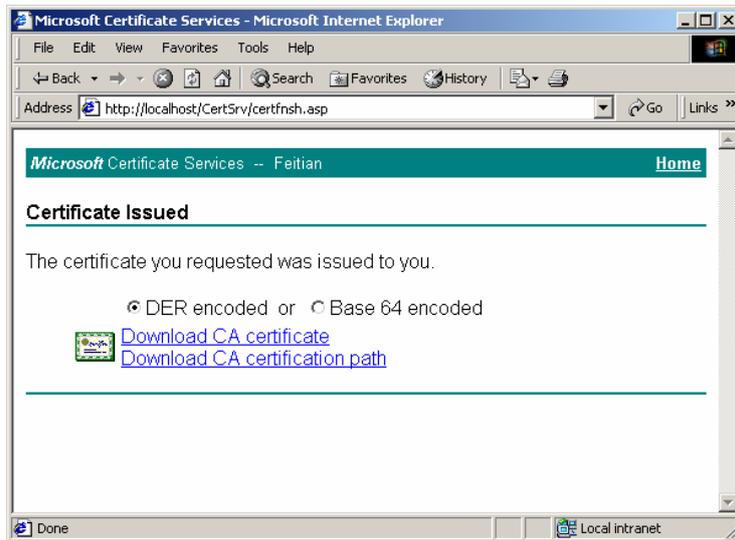


Figure 5.39

25. After you download the certificate, you must install it with the Certificate Installation wizard. See Figure 5.40.



Figure 5.40

If you send the request directly to a CA already on the network, the wizard will send a request to the CA. After the CA verifies your identity it will issue the certificate to you and it will be automatically installed on your WEB server.

After you have installed the certificate for the server, you may come back to the website property setting window to set SSL port. The default SSL port is 443.

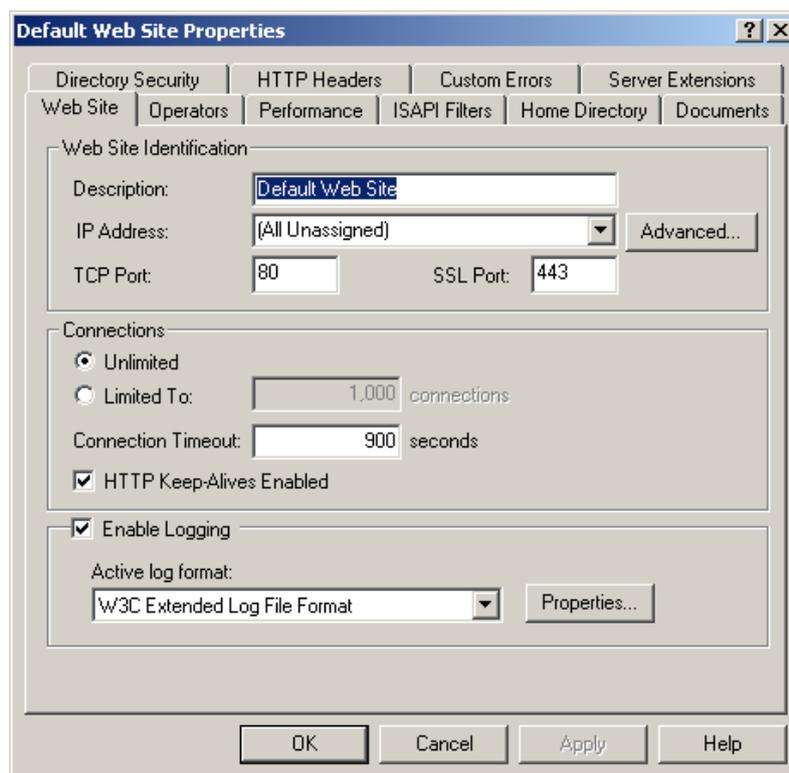


Figure 5.41

Click table “Directory Security” you will see that the “View Certificate...” button and “Edit...” button under **Secure communication** are available. You may now set the SSL applications:

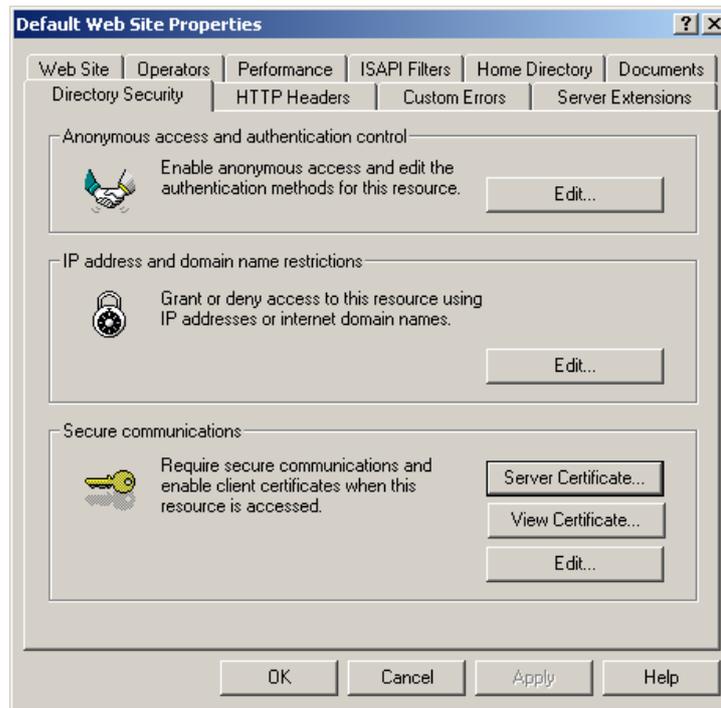


Figure 5.42

Set the SSL applications:

1. Click **Directory Security** tab. See Figure 5.42.
2. Click **Edit** on the **Secure communications** dialog box.
3. Activate **Require secure channel (SSL)** and **Require client certificates**. Then click **OK**. See Figure 5.43.

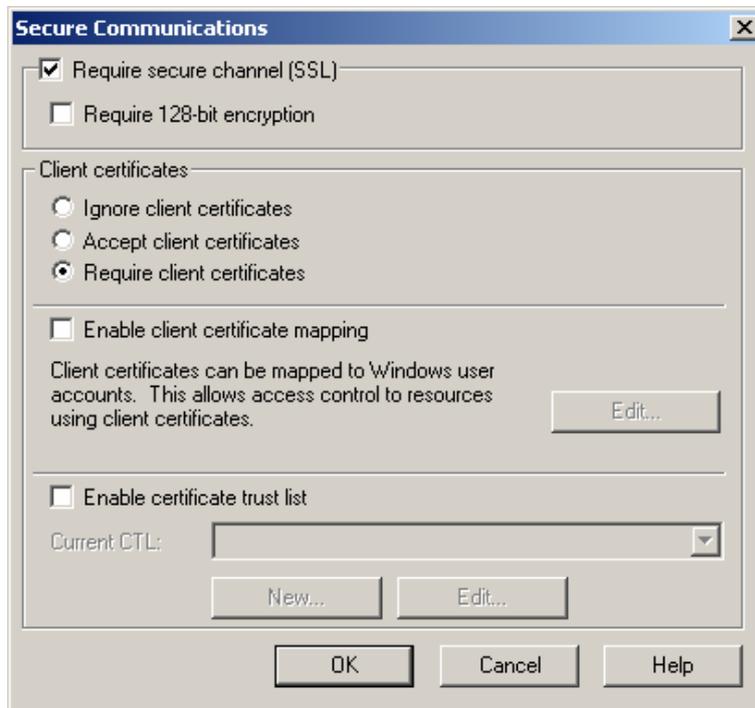


Figure 5.43

- **Require secure channel (SSL)**: If we do not activate this option the web server will offer the WWW service in HTTP protocol by default. If we activate this option IIS system will force the WWW client browser to use WWW service through SSL communication protocol. That means when we activate this option the system will close all *http:* links, and only *https:* links are allowed to connect to the web server.

- **Require client certificates:** The access is enabled with a certificate. This guarantees a higher security level.

Click “OK” to enable the settings.

We have finished the settings for secure website and activated the secure communication. An error will be issued if you load the page with “http:”.

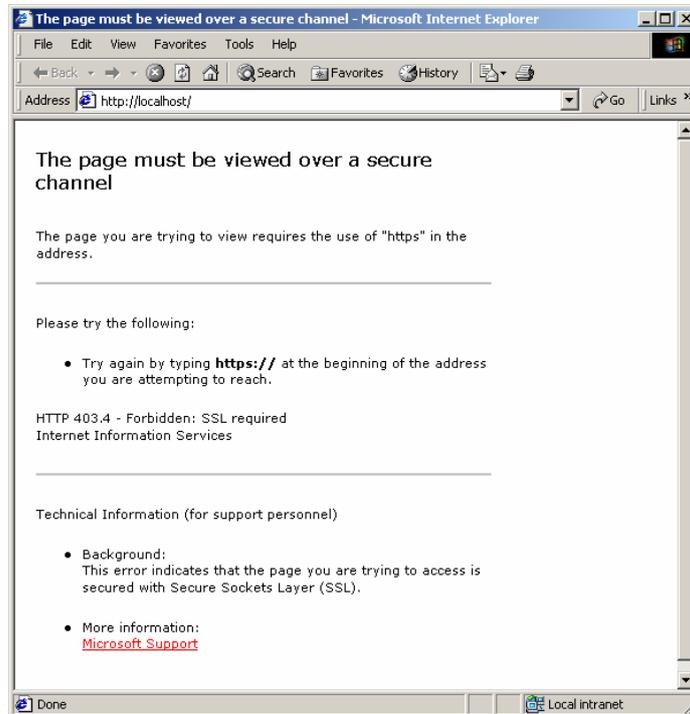


Figure 5.44

The system prompts that you need to load the web page with “https:”, and when you load the secure web page with “https:” you will be prompted with a warning message.



Figure 5.45

Click “OK” and you are asked to select the certificate for identification. At present we have no client certificate, so the certificate list is blank.

Then let us discuss how to request client certificate with ePass1000.

5.5.4 Request Digital Certificate with ePass1000

First insert an ePass1000 token that has been initialized for PKI applications. Then connect to a CA website with IE and request a certificate. We will use the Verisign certificate server as an example.



Figure 5.46

Fill the required information as prompted and select **FTsafe ePass1000 RSA Cryptographic Service Provider** for

CSP.



Figure 5.47

This means that the key pair of the certificate will be created by ePass1000. When prompted, type the user PIN associated with the ePass1000 token. Click **Login**.



Figure 5.48

Then continue the request operation

You may view the certificate with ePass1000 Manager, please refer to Chapter 4 ePass1000 Manager.

5.5.5 Access SSL Secure Website with ePass1000

Since the SSL secure website is ready, let us access this secure website with ePass1000 now.

Request a Web browser certificate or a similar digital certificate with ePass1000, and the CSP is "FTsafe ePass1000 RSA Cryptographic Service Provider". This process is similar to "Request Digital Certificate with ePass1000".

You may access the SSL secured website when the Web browser certificate is stored. in ePass1000.

5.5.6 Receive/Send Signed or Encrypted Email with ePass1000

This example assumes you have an Outlook Express email account and a digital ID (Certificate).

You may refer to "Request Digital Certificate with ePass1000" to request one Email security certificate. Be careful that the certificate type must be Email security certificate, and the Email address you input for request the certificate must be the one that you will use in Outlook Express to send and receive secure Email. If you have one Email security

certificate already (and this certificate contains the private key), and this certificate is valid, you may import this certificate with ePass1000 Manager.

Next we need to set the properties for the Outlook Express account.

Open Outlook Express, and choose Accounts from the Tools menu. See Figure 5.49.



Figure 5.49

Select **Mail** tab. Choose which mail account to define certificates for and click **Properties**. Then click the **General** tab, to check if any information is not correct. See Figure 5.50.

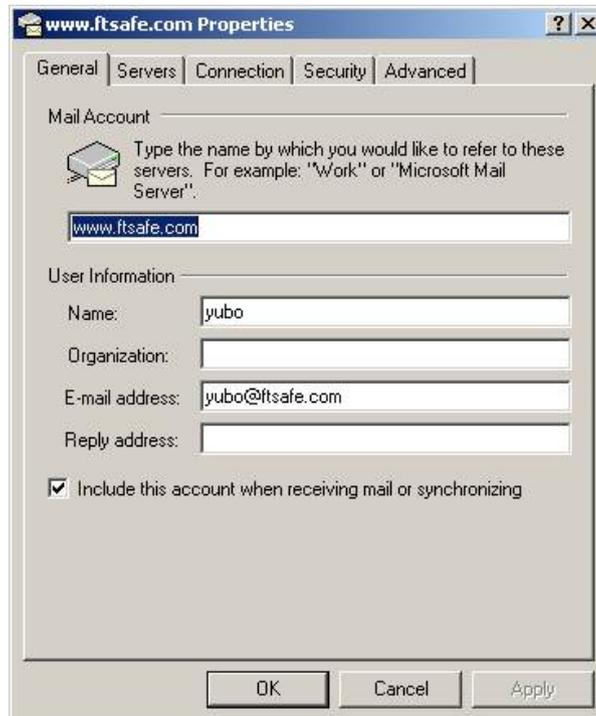


Figure 5.50

Click “Security” tab, to see its security settings. Click the **Select...** button for **Signing certificate** and select the certificate you want to use for signing and click **OK**, to enable you sign the Email sent from this account. Click the **Select...** button for Encryption preferences, and select the certificate you want to use for encryption and click **OK**, to enable you encrypt the Email sent from this account. You may also select the encryption algorithm. See Figure 5.51.



Figure 5.51

Click “OK”, then click on the digital ID that you will use for this Email account in the selection list. Outlook Express will only use your specified digital ID to perform the security functions.



Figure 5.52

Note: The certificates stored in ePass1000 will be enumerated if ePass1000 is already attached to your PC.

Click “OK” to complete the settings and go back to Outlook Express main window.

Select “Options” from “Tools”. Then click table “Security”. See Figure 5.53.

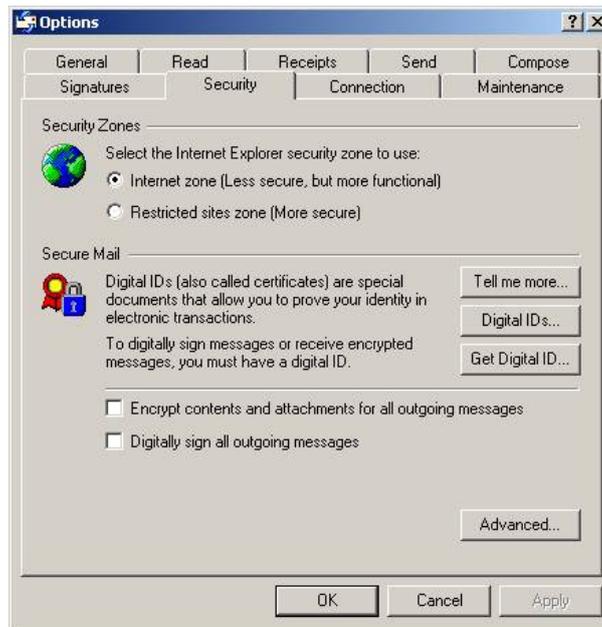


Figure 5.53

If you want to sign all outgoing Email, please enable **Digitally sign all outgoing messages**. See the figure above. We will discuss how to sign the individual outgoing Email later.

If you want to encrypt all outgoing email, please enable **Encrypt contents and attachments for all outgoing messages**. See the figure above. We will discuss how to encrypt the content and attachment of individual outgoing Email later.

Click **Advanced...** button.



Figure 5.54

Click check box **Always encrypt to myself when sending encrypted mail** under Encrypted messages.

Click check box **Include my digital ID when sending signed messages** and **Add senders' certificates to my address book** under Digitally Signed messages. The recipient must get the sender's certificate to verify the signature, and only when the recipient has the sender's certificate he can send encrypted mail to the sender.

You may modify other settings such as the key length. We have finished the settings for Outlook Express. When we send email it will be automatically encrypted and signed.

Obtain the Receiver' Public Key and Certificate

To send an encrypted and digitally signed email message you must first obtain the receiver's public key and certificate

information. You use the public key to encrypt the email so that only the receiver, who possesses the corresponding private key, can decrypt and read the message, and verify the digital signature.

To obtain other person's public key or certificate you must ask him to send you a signed email. Then you should store the digital ID. See the detailed steps below:

1. Ask the receiver to send an email to you with his digital signature.
2. Open the signed email sent to you in Outlook Express.
3. Right click "Sender" field and select "Add to Address Book". Then click "OK" to automatically import the person's certificate to the Outlook Express address book.

Send Encrypted Email with Outlook Express

To send an encrypted email you must first have the receiver's public key or certificate and have added the certificate information to your Outlook Express address book.

1. Create a new message.
2. Select recipient from address book that sent you a signed email. A red mark on the recipient's icon will indicate that the digital ID is attached.

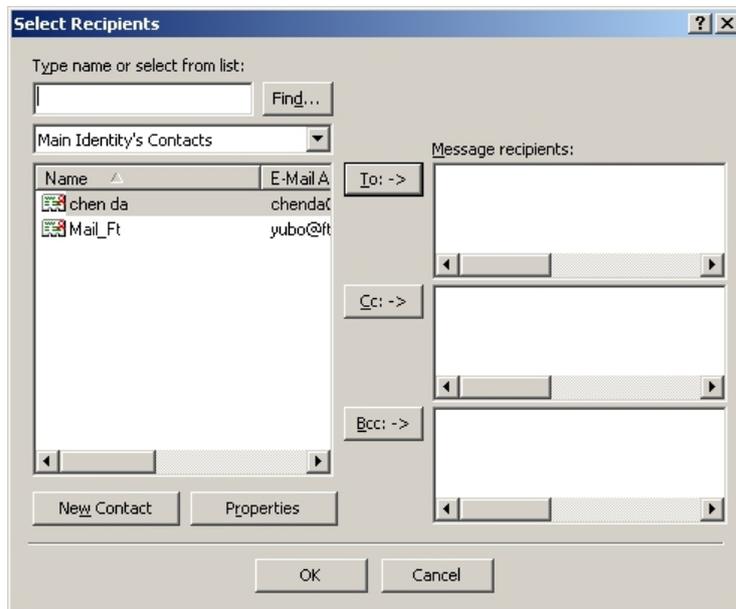


Figure 5.55

3. Fill in the message as you normally would.
4. Click **Encrypt** button, see Figure 5.56.

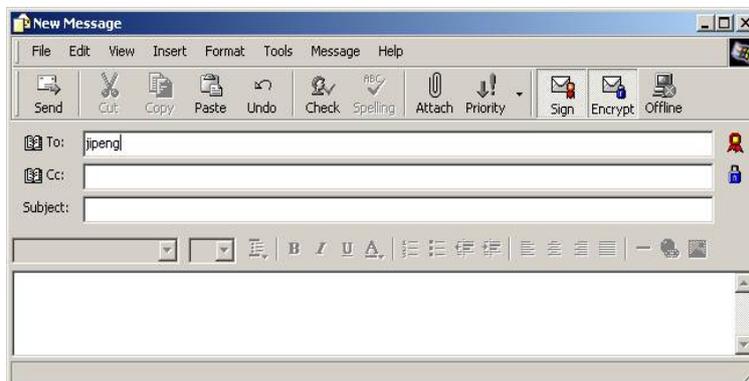


Figure 5.56

5. Click **Send** to send this email.

5.5.7 Win2000 Smart Card Logon with ePass1000

Windows2000 and later systems are designed to support a smart card logon function. Computer users may logon to the computer system in the traditional user name + password method, or logon with a smart card. Smart card logon is much more secure and convenient than the traditional method. Users only need to keep the smart card User PIN code in mind.

To logon Windows2000 station with a smart card requires that we set the station to issue smart card certificates to the users. A smart card certificate is a certificate stored in the smart card.

Please refer to Microsoft documents to learn how to set the CA to issue smart card certificates for the users.

See below how to request smart card logon certificate with ePass1000.

Suppose that there is a website available for us to request the smart card certificate. We would need to fill in the user information and pay attention to the following issues:

1. Select **Smartcard User** for **Certificate Templates**.
2. Select **FTSafe ePass1000 RSA Cryptographic Service Provider** for **Cryptographic Service Provider**.
3. Select the requested enrollment agent certificate for **Administrator Signing Certificate**.
4. Select the user who is being enrolled for the certificate by clicking **Select User**. Then Click **Enroll** button.
5. Type the ePass1000 token user PIN code in the pop up authentication dialogue box, and then the certificate will be generated.

The above is just an example and different Websites may look differently. Requesting smart card certificates is quite similar to requesting a common digital certificate.

You may now attach your ePass1000 with smart card certificate inside to perform Windows2000 smart card logon.

5.5.8 VPN Remote Logon with ePass1000

Virtual Private Network (VPN) builds a secure communication tunnel on the insecure network environment, and it works like a private tunnel. Windows2000 supports VPN applications. Windows users may connect to the website through VPN in the similar way that they dial to login their ISP server.

To set up the VPN secure tunnel the server and client need to exchange trust and create secure session keys, and encrypt the future information with this session key. Windows2000 station allows the users to login with the smart card at a client site. We will introduce how to configure VPN server taking Win2000 VPN router software as the example.

1. From **Control Panel > Administrative tools**, double click **Routing and Remote Access**, and the console will pop up.
2. Right click the sever name in the left, and select **Configure routing server**.



Figure 5.57

3. Click **Next** to continue and select **Virtual private network (VPN) server** in below figure.



Figure 5.58

4. Click **Next** to select the protocol type.
5. Click **Next** to select the network interface.
6. Click **Next** to select VPN logon client IP address allocation policy. You may ask the DHCP server to allocate automatically or specify fixed IP address range as the address pool.
7. Click **Next** to configure RADIUS for VPN. You may not configure RADIUS from this screen, either.

VPN server has been successfully installed. Now we need to configure logon authentication methods for VPN server and we need to log on with smart card.

8. Right click the server name in the left of the “Routing and Remote Access” console and select **Properties**.
9. Click “Security” table, then click **Authentication** button.

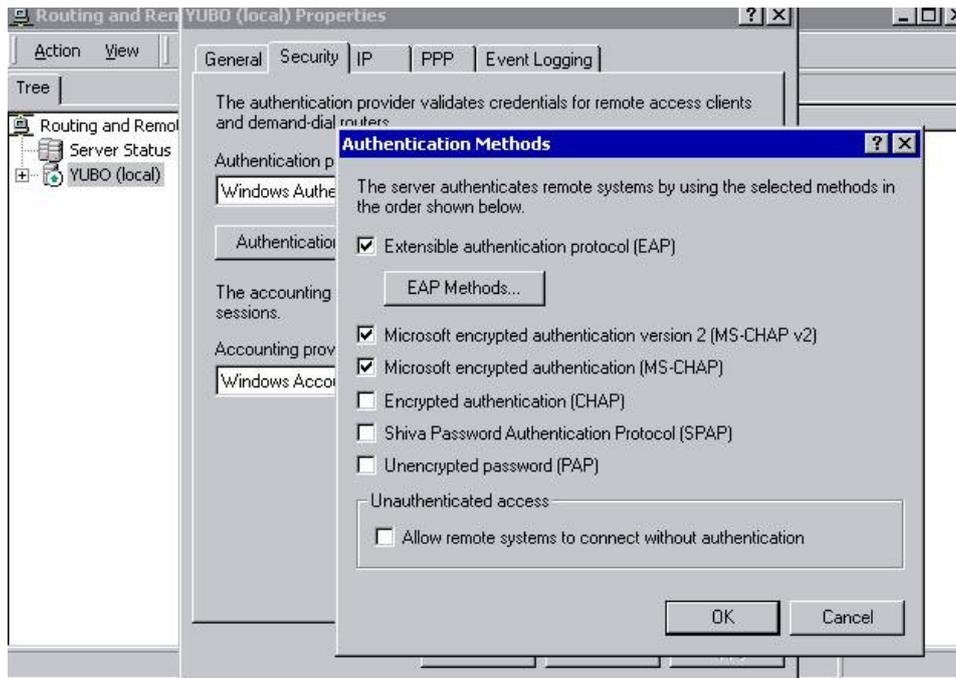


Figure 5.59

10. Select **Extensible authentication protocol (EAP)**. EAP is an improvement to traditional user name and password authentication. Smart card user authentication is one type of EAP.
11. Close “Authentication Methods” dialogue box.
12. Close “Server Properties” dialogue box.
13. Select “Remote Access Policy” in the left tree structure of the “Routing and Remote Access” console.
14. Double click to display the settings.
15. Click **Edit configuration file** button.
16. Click “Authentication” table in “Edit Dial-in Profile” dialogue box.

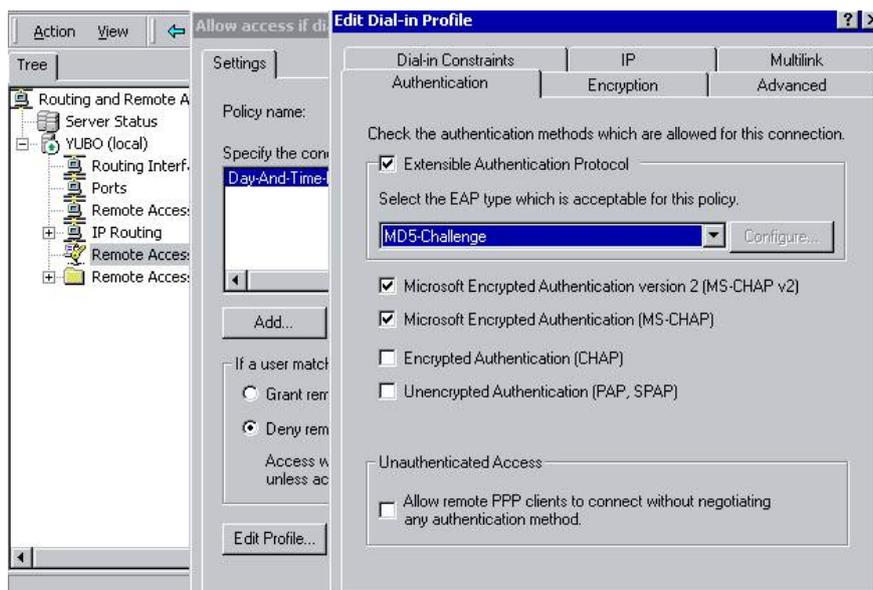


Figure 5.60

17. Select **Extensive Authentication Protocol**. Click **OK**.

The VPN server configuration is finished.

Then let us configure the client software:

- 1) Go to **Control Panel > Dial Tool**.
- 2) Double click **Set up new link** icon. The Network Connection Wizard will be activated.

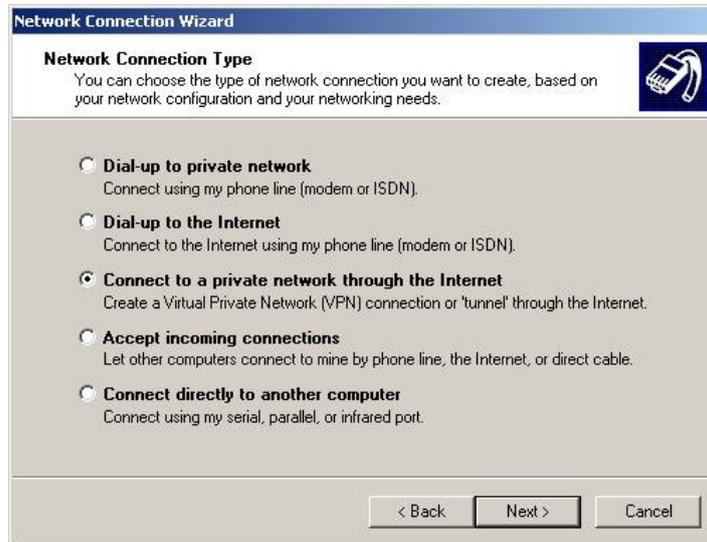


Figure 5.61

- 3) Select **Connect to a private network through the Internet** and click **Next**.
- 4) Fill the IP address and domain for the VPN server.
- 5) Select **Use my smart card**.



Figure 5.62

- 6) Input the new VPN connection name.

The VPN client software configuration is finished.

You may request a smart card certificate with ePass1000 and then attach this ePass1000 to your computer. Double click the newly created VPN dial icon and input ePass1000 User PIN code as prompted.

5.6 Other Applications

ePass1000 also supports other application besides PKI.

Feitian offers an ePass1000 proprietary API - **ePass1000 C/C++ API**. ePass1000 proprietary API supports other applications and it can raise data security to a new level.

Please refer to ePass1000 Developer's Reference in ePass1000 SDK to know more details about **ePass1000 C/C++ API**.

Chapter6

Distributing the ePass1000 Application

Drivers and other software components must be properly installed before ePass1000 may be used with an application. There are two ways to install the ePass1000 software components:

- ✓ Use the Redistribution package provided with ePass1000 SDK
- ✓ Use the API provided with the ePass1000 SDK to program your own installation utility.

To program with the ePass1000 API, the following files are required:

Files	SDK Paths
ePassAPI.h	/Private/Include
ePsInst_drv.lib	/Private/Lib
ePsInst_drv.dll	/Private /Lib (Must be available at run time)
inst_drv.dll	/Private /Lib(Must be available at run time)

Invoke the `esa_DoInstall` function to install the ePass1000 drivers. Uninstall the drivers with the `esa_DoUninstall` function.

The ePass1000 Setup API library will not reboot the machine automatically. The `esa_IsNeedReboot` function may be used to determine if the system needs a reboot. If an installation progress bar is desired the `esa_SetShowMsgNotifyCallback` may be used to set up a call back function to generate a current installation message. This information may be used to program a UI update.

Sometimes the uninstall program may need to delete a file which is used by another program. The ePass1000 Setup API provides the `esa_SetDeleteFileCallback` function to handle this requirement.

For more information regarding the ePass1000 Setup API, please refer to the ePass1000 Developer's Reference and sample program under the SDK\Samples directory.

Appendix I

Frequently Asked Questions

1. ***When ePass1000 is inserted into the USB port, the system does not prompt that new hardware is found and ePass1000 is unusable.***

First verify that the operating system is one that supports ePass1000 (WIN 98/ME/2000/XP; MAC OS 8/9; Linux). Then check the system BIOS to determine if options related to the USB port are enabled. If the USB options are disabled, enable them and retry ePass1000.

If the system is still not working properly, there is likely a problem with the USB port or connection cables. Connect another USB device to the USB port (for example, a USB mouse or keyboard). If the other USB device works properly, it is likely that ePass1000 has malfunctioned. If the other USB device does not work, the USB port has probably malfunctioned.

2. ***ePass1000 has been normally installed, but it fails to log into the ePass1000 protected Web site?***

Please check the security setup of the browser. Verify that the ActiveX widget and plug-in unit are permitted to run.

3. ***ePass1000 can be opened fine by one program, but cannot be found by the second program that tries to open it?***

For purposes of security, ePass1000 can only be opened by one program at a time. Only after the first program has properly closed ePass1000 can the second program open it.

4. ***Why does ePass1000 seem to work inconsistently when attached to a USB Hub?***

We have found that some USB Hubs with internal power supplies seem to have electrical interference problems that can affect ePass1000. If it is necessary to use a Hub, try to select one with an external power supply.

5. ***Does my computer have a USB interface?***

Generally, all of computers above PII have a USB interface and some older computers may have one also. If there is no USB port on your computer, insert a USB interface extension card into PCI slot. Please note that if motherboard has a USB function, you must disable it in BIOS.

Appendix II

Technological Specifications for ePass1000

Supported Operating Systems	Windows 98/ME/2000/XP/2003/Vista; Mac OS 8 or 9; Linux
Certifications and Standards	PKCS#11, MS CAPI, PC/SC, X.509 v3 Certificate Storage, SSL v3, IPSec/IKE
Memory Size (by Model)	8k & 32k
On-Board Security Algorithms	MD5
Chip Security Level	Secured and Encrypted Data Storage
Dimensions (by Model)	58 x 14 x 7 mm (2.28 x 0.55 x 0.28 inches) & 50 x 17 x 7 mm (1.97 x 0.67 x 0.28 inches)
Weight (by Model)	8g & 6g
Power Dissipation	< 250 mW
Operating Temperature	0°C to 70°C
Storage Temperature	-40°C to 85°C
Humidity Rating	0 to 100% without condensation
Connector Type	USB type A (Universal Serial Bus)
Casing	Hard Molded Plastic, Tamper Evident
Memory Data Retention	At least 100 years
Memory Cell Rewrites	At least 100,000