Chapter 1. RACF's web interface to digital certificate generation

RACF APAR OW45211 and SAF APAR OW45212 introduces a Web based front-end into digital certificates generation for clients using a Web browser such as, Netscape Navigator or Microsoft's Internet Explorer (IE).

This is extending on Public Key Infrastructure (PKI) services we supplied with APAR OW31933, that provided the RACF Auto Registration (RAR) Application. The RAR application shipped sample HTML and REXX code to provide Web based front-end into linking a client's digital certificate with it's associated RACF User ID.

These new services provide again sample HTML and REXX code to generate client digital certificates, as is it aimed at replacing the functionality supplied earlier by the CA Servlet that shipped with the IBM HTTP Server for OS/390. As of OS/390 Version 2 Release 10, this function is no longer shipped.

1.1 Overview

Figure 1 shown an overview of the components involved, when generating a client's digital certificate using the Web based front-end.

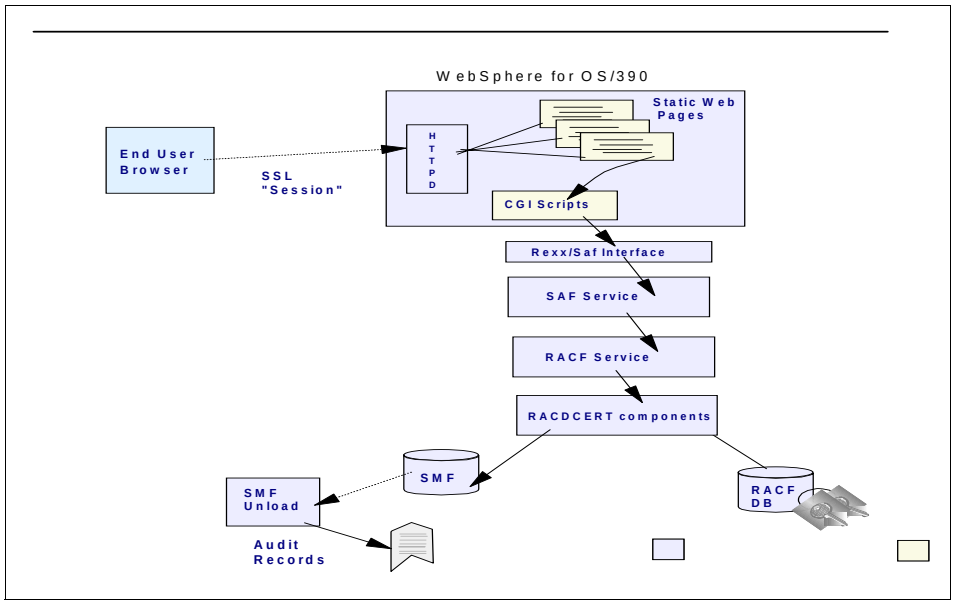

Figure 1. Overview of the Web based process of generating client certificates

The certificate request flow is as follow:

1. The client selects the web page, either linked or directly, to generate a digital certificate over an SSL established session.

   Note: SSL server side authentication only.

2. The client fills in the web page with the required information to make up the distinguished name of the client like Common Name, Organization etc.

   Note: The required fields can be tailored, according to the options defined in the configuration file, which we will discuss later.

3. The client submits the request and new RACF callable services will invoke the existing RACF RACDCERT facilities to generate a digital certificate request.

4. The client is notified of the successful or unsuccessful certificate request.

   The client is issued a transaction identifier, that is used to pick up the approved certificate.

5. The client can now pick up his certificate based on the transaction identifier and install it into it's browser. The client also has the option to defer the pick up of his certificate to a later date or time.

   Note: There is no formal approval process involved here. The setup assumes the client is authorized by RACF to generate a certificate and have it signed by the Certificate Authority (CA) selected. This is explained later in the customization section.

1.2 Directory structure provided

RACF APAR OW45211 and SAF APAR OW45212 supplies the files and separate directories according to the following directory structure, as shown in Figure 2 on page 3.

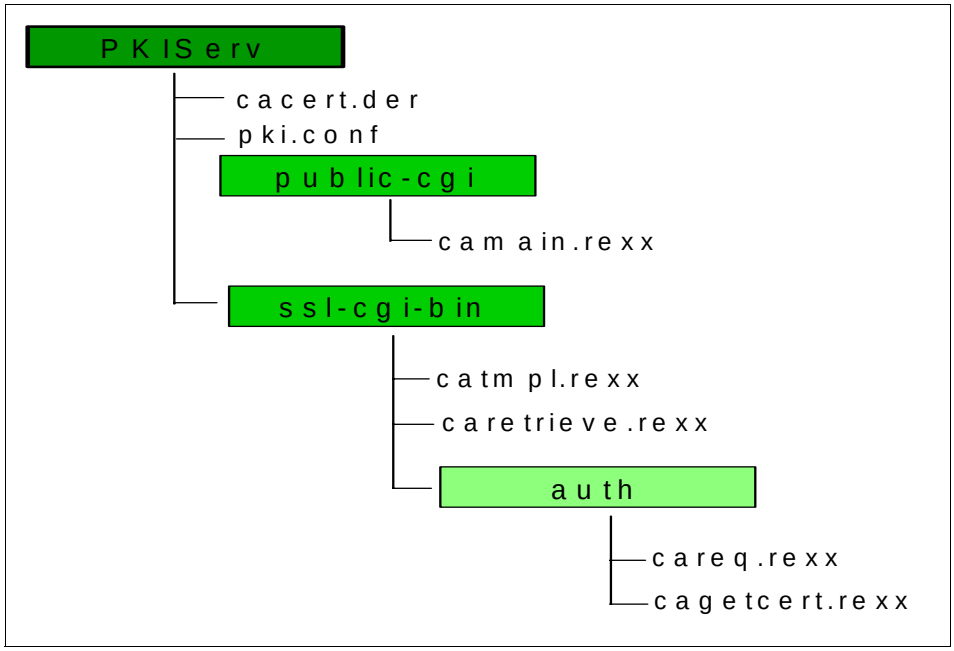

Figure 2. Directory Structure provided by APAR

The directory structure consists of the following root directory:

PKIServ this is the root directory and contains the following files:

- **cacert.der** this is the local certificate authority (CA) signing certificate
- **pki.conf** this is the main configuration file, and basically controls the setup of the new functions provided.

The `PKIServ` directory contain two sub directories, called:

public-cgi this directory contains the main program, called `CAMAIN.REXX`

ssl-cgi-bin this directory contains the executables, that need to be executed only when an SSL connection is established between the client and the Web server.

The `SSL-CGI-BIN` directory contains one more sub directory, called:

auth this directory contains the code required to request and retrieve a digital certificate and requires both an SSL established session and a RACF User ID and password.

Note: The supplied directory structure can be changed if required, but would require changes to the files itself. Of course the main directory PKIServ can be installed as a subdirectory structure within your file system if required.

1.3 Explanation of the configuration file

The configuration of the new PKI services is done through a file, called PKI.CONF located in the root directory. It introduces the concept of certificate templates which define the fields that comprise a specific certificate request. These templates are examined by the REXX connector logic within the web server to establish defaults and identify the fields which can be changed on a certificate request. Certificate templates are shipped in pseudo HTML, which enables you to easily modify to suit your installation's needs. In essence, much of the intelligence with respect to certificate generation is contained within the certificate template, with the new SAF service providing the basic storage and management primitives.

1.3.1 Structure of the configuration file

The file contains a mixture of true HTML and HTML like tags. The main tags divide the file into sections, APPLICATION, TEMPLATE, and INSERT, where APPLICATION and TEMPLATE may contain various subsections, named fields, and substitution variables as explained next.

<APPLICATION NAME=appl-name> ... </APPLICATION>

This section identifies the applications that will make use of PKI Services For OS/390. The product ships with one application defined, “PKISERV”. This sections contains one subsection, CONTENT.

1. <CONTENT> ... </CONTENT>

   This subsection contains the HTML to be presented to the end user requesting and retrieving certificates.The subsection should contain one or more named fields identifying certificate templates to be used for requesting or managing certificates through this application. (See next for a description of named fields.) These template names should match the HTML selection value associated with them.

<TEMPLATE NAME=tmpl-name> ... </TEMPLATE>

This section defines the certificate templates referenced in the APPLICATION sections. Applicable subsections are:

1. <CONTENT> ... </CONTENT>

   This subsection contains the HTML to be presented to the end user requesting certificates of this type. Any named fields in this subsection are

interpreted as certificate field names defined by INSERT sections. (See next for a description of INSERT sections.) For PKISERV, the INSERT sections are included as part of the HTML presented to the end user. (i.e., the end user provides values for these fields.) Named fields in this subsection are considered optional if the named field contains more that one word within the %% delimiters, e.g., %%AltName (Optional)%%. The user need not supply a value for AltName

2. <APPL> ... </APPL>

   This subsection identifies certificate fields that the application itself should provide values for. This subsection should contain named fields only, one per line. Currently, the only supported named field allowed in this section is “UserId”

3. <CONSTANT> ... </CONSTANT>

   This subsection identifies certificate fields that have a constant (hardcoded) value for everyone. This subsection should contain named fields only, one per line. The syntax for specifying the values is %%field-name=field-value%%, e.g., %%KeyUsage=handshake%%

4. <SUCCESSCONTENT> ... </SUCCESSCONTENT>

   This subsection contains the HTML to be presented to the end user when the certificate request was submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

5. <FAILURECONTENT> ... </FAILURECONTENT>

   This subsection contains the HTML to be presented to the end user when the certificate request submit failed. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

6. <RETRIEVECONTENT> ... </RETRIEVECONTENT>

   This subsection contains the HTML to be presented to the end user to enable certificate retrieval. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

7. <RETURNCERT> ... </RETURNCERT>

   This subsection contains the HTML to be presented to the end user upon successful certificate retrieval. For PKISERV, if the certificate being retrieved is a browser certificate, then this section must contain a single

line containing a browser qualified INSERT name, e.g., %%returnbrowsercert[browsertype]%%. Additionally, INSERTs for Netscape (returnbrowsercertNS) and Internet Explorer (returnbrowsercertIE) containing browser specific HTML for returning certificates must be defined elsewhere in the configuration file. If the certificate being retrieved is a server certificate, this section should contain the HTML necessary to present the certificate to the user as text

8. <INSERT NAME=insert-name> ... </INSERT>

   This section contains HTML that either describes a certificate field or defines other common HTML that may be referenced in the TEMPLATE sections. INSERTs are referenced elsewhere by using a *named field* of the form %%insert-name%%

Named Fields are delineated with %%, e.g., %%Label%%. Their meaning is specific to the section they are contained in. Named fields are case sensitive. Named fields are also using to reference common includeable HTML. Note, PKISERV treats named fields that begin with a dash as just includeable code. Any special meaning a named field may have, given the section it's contained in, is ignored if it begins with a dash. For example, if %%-*pagefooter*%% was specified in a TEMPLATE CONTENT section, -pagefooter would not be considered a certificate field name. However, the INSERT with the name *-pagefooter* would be included in the HTML page presented to the end user.

Substitution variables are delineated with square brackets, e.g., [base64cert]. They represent variables that get replaced with an actual value at run time. Substitution variables are case sensitive. The valid substitution variables are:

transactionid Unique value returned from a certificate request

tmplname Certificate template name. Primed from the HTML tag <SELECT NAME="Template"> in the <APPLICATION NAME=PKISERV> section. This is selected by the end user on the first web page.

base64cert The requested certificate base64 encoded

iecert The requested certificate in a form the Microsoft Internet Explorer accepts

browsertype Special substitution variable to be used to qualify named fields only. It use enables the different browsers, Netscape and Internet Explorer, to perform browser specific operations, i.e., Netscape uses a KEYGEN HTML tag to generate a public/private key pair while Internet Explorer uses ACTIVEX controls. For example, if %%PublicKey[browsertype]%% was specified in a TEMPLATE

CONTENT section referenced by a user with the Netscape Navigator browser then INSERT PublicKeyNS would be included. Likewise, if the user's browser was the Microsoft Internet Explorer, INSERT PublicKeyIE would be included

optfield Special substitution variable that should be placed in any certificate field name INSERT where the value may be supplied by the end user. It enables the field to be displayed as optional if desired

Note: Depending on where a substitution variable is used, it may not have a valid meaning, e.g., base64cert would be meaningless prior to the certificate being retrieved. The value of [base64cert] would be the empty string (aka NULL) in this case.

1.4 User Interface

The end user Interface is browser based static HTML pages. These are produced by the REXX connector CGIs by reading the installation customized configuration file. The user is required to have a RACF user ID and password for authentication. This is similar in concept to the RACF Auto-Registration (RAR) samples. The User's (or application's) access to new and existing RACF FACILITY class profiles determine if the user is authorized to generate and retrieve certificates through this interface. Installation customizable certificate templates contained within the configuration file, control which fields are user customizable via the HTML dialogues.

Figure 3 on page 8 shows the first screen in a flow to request a digital certificate.

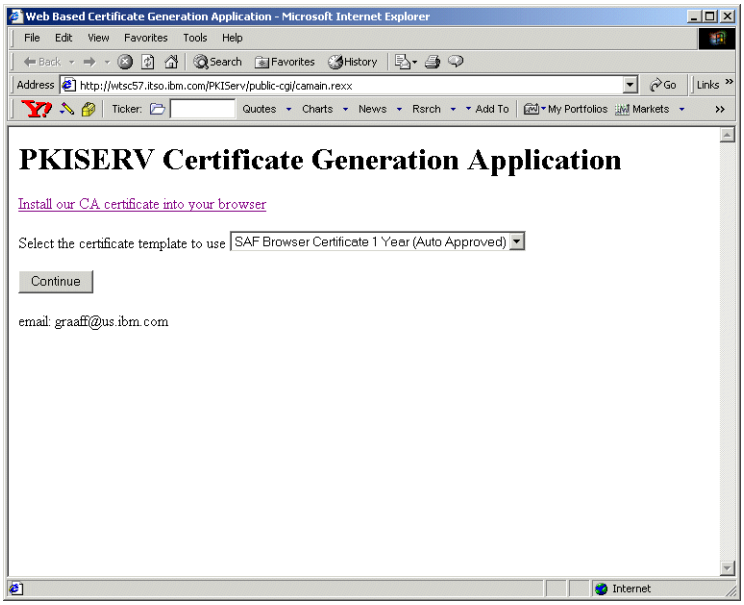

Figure 3. PKISERV certificate generation application page

This is the main page where the user can select a certificate template (model). The REXX CGI (CAMAIN.REXX) produces this page by reading the configuration file and answering everything in the <CONTENT> section under <APPLICATION NAME=PKISERV>. This page is not SSL protected but subsequent pages in this flow are.The Hypertext link “downloads” and installs OS/390 Root CA certificate into the user's web browser.

There are two selection choices to make on this page, to request a digital certificate:

1. SAF Browser Certificate 1 Year (Auto Approved)

   This option is intended to request a digital certificate for a end user/client. The certificate has a validity period of one year (365 days) and is automatically approved.

2. SAF Server Certificate 1 Year (Auto Approved)

   This option is intended to request a digital certificate for a server, like the HTTP, LDAP, TN3270 or other server. Again the certificate has a validity period of one year (365 days) and is automatically approved.

Note: These are the options that come as samples, but can be customized to suit the installations needs, as we discuss later.

Pressing the **Continue** button moves use to the next dialogue in the sequence, as shown Figure 4.

Figure 4. PKISERV certificate generation application - browser certificate page (1)

Once a template is selected, the user then enters the information permitted for that template (in this case the certificate label and key size). The REXX CGI (`CATMPL.REXX`) produces this page (Figure 4 on page 9) by reading the configuration file and answering everything in the `<CONTENT>` section under the `<TEMPLATE>` selected by the user. The page is SSL protected.

There are only two fields generated on this page that require a selection:

1. %%Label%%

   This field indicates to specify of the label associated with the RACF certificate being generated.

2. %%PublicKey browsertype%%

   This field generates a so called INSERT based on the browser used. In this case we use Internet Explorer and select the INSERT PublicKeyIE. This provides us with a selection of the Cryptographic Service Providers (CSPs) Internet Explorer supports to generate a public/private keypair.

   Note: Depending on the operating system you are using and the version of the browser, various CSPs may or may not show.

   In our example we are using WIndows 2000 Professional Edition and Microsoft's Internet Explorer 5.5 with a cipher strength of 128 bit.

Again the page can be customized to suit your needs and allow for additional fields used in the certificate to be specified, like:

- Common Name (CN)
- Organization (O)
- Organizational Unit (OU)

This page can also be used to retrieve a certificate from a request that was submitted earlier.

If any required fields are not filled in, the CGI would produce an error.

When the **Continue** button is pressed, the user would get prompted to enter a valid user ID/password, as shown in Figure 5 on page 10.

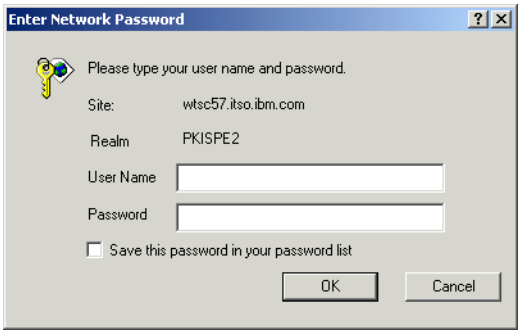

Figure 5. RACF User ID and password prompt window

To be able to request a digital certificate a User ID requires authorization to various RACF FACILITY class profiles. These profiles are discussed in section 1.5.2, "Determine RACF access to the RACDCERT services" on page 17.

Next you enter your User ID and password and if you are authorized to personal certificate services, the next page is displayed, as shown in Figure 6.

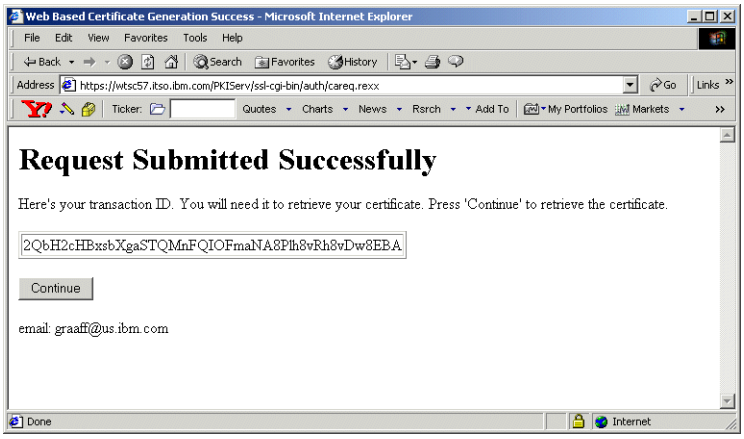

Figure 6. Certificate request submitted successful page

The user provided fields are gathered along with fields specified in the <APPL> and <CONSTANT> sections under the <TEMPLATE> selected by the user. All field values become parameters and the request is submitted to RACF (IRRSPXGL is called). If successful, a *transaction ID* (aka Certificate ID) is returned. The *Certificate ID* uniquely identifies the newly created certificate in RACF**.** The REXX CGI (CAREQ.REXX) produces this page by reading the configuration file and answering everything in the <SUCCESSCONTENT> section under the <TEMPLATE> selected by the user. The digital certificate has now been generated and can be picked up using the so called “transaction ID”. If we take a look at RACF, we can see the certificate as well off course using a RACDCERT LIST command, as shown in Figure 7.

```
racdcert list(label('Paul IE cert 09/24/2000')

Digital certificate information for user GRAAFF:

Label: Paul IE cert 09/24/2000
Certificate ID: 2QbH2cHBxsbXgaSTQMnFQIOFmaNA8Plh8vRh8vDw8EBA
Status: TRUST
Start Date: 2000/09/24 00:00:00
End Date:   2001/09/24 23:59:59
Serial Number:
     >02<
Issuer's Name:
     >CN=IBM-ITSO PDG CA.OU=ITSO.O=IBM.L=Poughkeepsie.SP=New York.C=US<
Subject's Name:
     >OU=ITSO.O=IBM.C=US<
Key Usage: HANDSHAKE
Private Key Type: None
Ring Associations:
*** No rings associated ***
```

Figure 7. RACDCERT LIST output from the generated certificate using the PKISERV application

As you can see from the RACDCERT LIST output, it shows the LABEL name we requested, the *transaction ID* (Certificate ID) and it indicates that there is no private key.

Note: The private key is at the workstation, not on the host (RACF) side.

To complete the digital certificate request process, we need to retrieve the certificate from RACF. To do this, we press the **Submit** button, to progress to the next page.

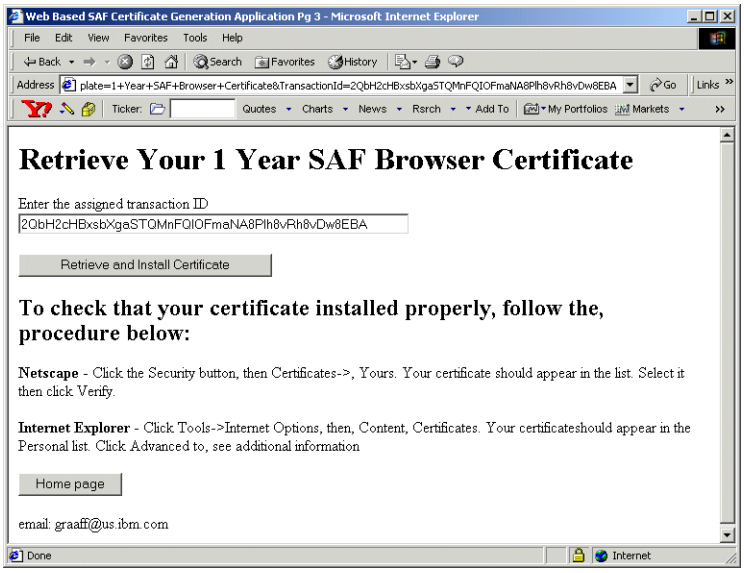

Figure 8. Retrieve browser certificate page

Pressing the **Submit** button presents the user with the page, as shown in Figure 8, to retrieve the certificate. The REXX CGI (CARETRIEVE.REXX) produces this page by reading the configuration file and answering everything in the <RETRIEVECERT> section under the <TEMPLATE> selected by the user.

Figure 8 also explains how you can check that the certificate is indeed properly installed.

To retrieve the certificate, we press the **Retrieve and Install Certificate** button. Figure 9 on page 14 shows the install page for Internet Explorer.

If the certificate being retrieved is a browser certificate, pressing the submit button will have the certificate downloaded directly into the browser certificate store. If the certificate being retrieved is a server certificate, the user will be presented with a page that enables the user to cut an paste the certificate to a file. The REXX CGI (CAGETCERT.REXX) produces this page by reading the configuration file and answering everything in the <RETURNCERT> section under the <TEMPLATE> selected by the user.

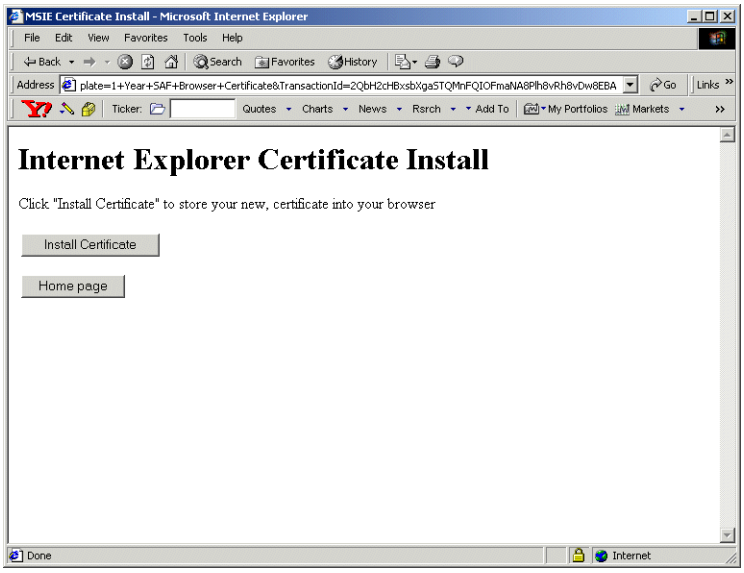

Figure 9. Internet Explorer certificate install page

Internet Explorer requires one more step then Netscape's Navigator. So again we press the **Continue** button, to install the certificate into the browser (IE). We receive a conformation message that the install was successful as shown in Figure 10.

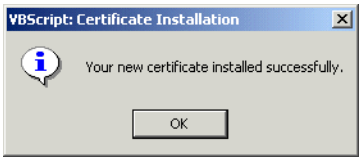

Figure 10. Certificate Installation successful message window

To check whether you have indeed successfully installed your certificate, for Internet Explorer you have to take the following steps:

1. select Tools from the main menu
2. select Internet options from the pull down list
3. select the content tab and then select certificates, as shown in Figure 11 on page 15

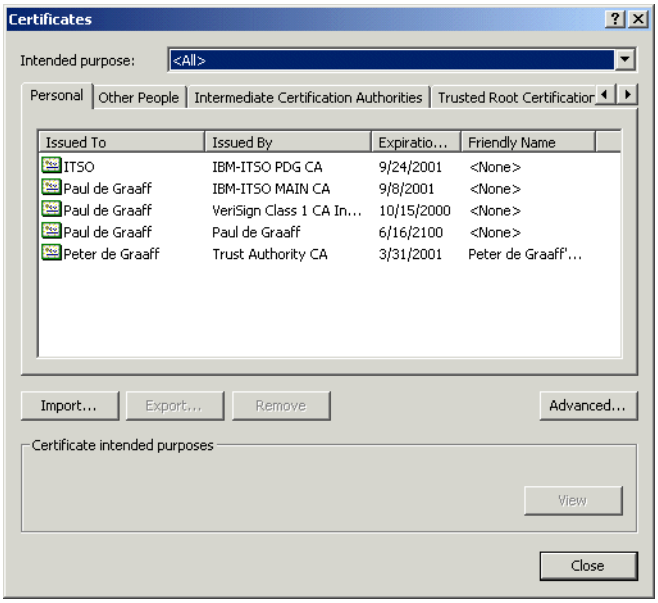

Figure 11. Internet Explorer Certificates Window

When you select you certificate, by double clicking on it, you can see the details about your certificate as shown in Figure 12 on page 16.

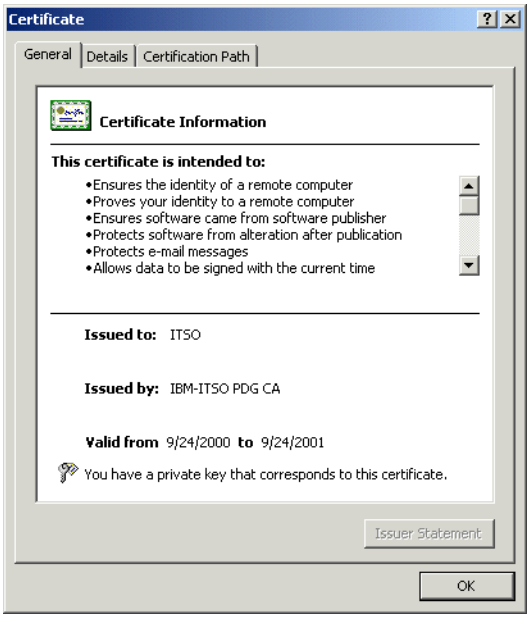

Figure 12. Certificate details window

You can see that the certificate was signed by the Certificate Authority (CA) `IBM-ITSO PDG CA` and issued to `ITSO`. Now where did he get that information from? We did not specify any of that information when we submitted our request for a certificate. The configuration file `PKI.CONF` holds the key to that information. Next we discuss the actual installation of the application and the configuration of `PKI.CONF` to make it better suit your needs.

1.5 Installation and configuration

This section details the steps needed to install the Web based digital certificate application. In logical order the following steps are needed:

1. SMP/E Install the PTFs related to RACF APAR 45211 and SAF APAR 45212

   See 1.5.1, “SMP/E install the PTF” on page 17.

2. Determine and setup the RACF access control needed to protect the `R_PKIServ` callable service, through profiles in the RACF FACILITY class.

   See 1.5.3, “Determine RACF access to the `R_PKIServ` callable service” on page 19.

3. Determine and setup the RACF access control needed to protect the RACDCERT services, through profiles in the RACF FACILITY class.

   See 1.5.2, “Determine RACF access to the RACDCERT services” on page 17.

4. Create your Certificate Authority certificate and make it available to your end users.

   See 1.5.4, “Create your Certificate Authority (CA) Certificate” on page 20.

5. Determine the server root directory for the web application and install the sample REXX scripts and PKI.CONF configuration file

   See 1.5.5, “Installing the sample code” on page 21

6. Configure the IBM HTTP Server (IHS) for OS/390 to use SSL

   See 1.5.6, “Configure the IBM HTTP Server for SSL mode” on page 21

7. Add the directives necessary to the HTTPD.CONF file to enable the PKISERV application

   See 1.5.7, “Other changes to make to the HTTPD.CONF file” on page 25

8. Customize the sample PKI.CONF configuration file and/or REXX scripts as needed for your organization’s use

   See 1.5.8, “Customizing the PKI.CONF file” on page 26

1.5.1 SMP/E install the PTF

Depending on your maintenance level of OS/390 Version 2 Release 10, you may or may not need to install the PTF’s related to APAR 45211 and 45212.

1.5.2 Determine RACF access to the RACDCERT services

To be able to generate a digital certificate with the RACF RACDCERT command, you need RACF access to the following profiles in the RACF FACILITY class:

1. IRR.DIGTCERT.ADD
2. IRR.DIGTCERT.GENCERT

The PKISERV application uses a Certificate Authority Certificate to sign all digital certificates indicated by the %%SignWith%% field in the configuration file PKI.CONF. The Certificate Authority (CA) certificate is likely to be owned by a User ID that is not used for signon purposes. So to allow somebody to sign

his/her certificate with that CA certificate the following authorities are needed for the PKISERV application to work:

1. READ access to the IRR.DIGTCERT.ADD profile in the RACF FACILITY class;
2. CONTROL access to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class;

As an alternative you can choose to use a surrogat User ID, like PKISERV to be authorized to this service, because you do not want everybody to have CONTROL access to IRR.DIGTCERT.GENCERT FACILITY profile. User ID PKISERV requires the following authorities:

1. UPDATE access to the IRR.DIGTCERT.ADD profile in the RACF FACILITY class;
2. CONTROL access to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class;

We defined User ID PKISERV as follows:

```
addgroup pki supgroup(sys1) owner(sys1)

adduser pkiserv dfltgrp(pki) owner(pki) name('PKISERV RACF APPL')

lu pkiserv

USER=PKISERV NAME=PKISERV RACF APPL     OWNER=PKI       CREATED=00.270
 DEFAULT-GROUP=PKI       PASSDATE=00.000  PASS-INTERVAL=180
 ATTRIBUTES=NONE
 REVOKE DATE=NONE   RESUME DATE=NONE
  LAST-ACCESS=00.273/12:05:55
  CLASS AUTHORIZATIONS=NONE
  NO-INSTALLATION-DATA
  NO-MODEL-NAME
  LOGON ALLOWED   (DAYS)          (TIME)
  ----------------------------------------
  ANYDAY                          ANYTIME
   GROUP=PKI       AUTH=USE      CONNECT-OWNER=PKI       CONNECT-DATE=00.273
     CONNECTS=    01  UACC=NONE     LAST-CONNECT=00.273/12:05:55
     CONNECT ATTRIBUTES=NONE
     REVOKE DATE=NONE   RESUME DATE=NONE
 SECURITY-LEVEL=NONE SPECIFIED
 CATEGORY-AUTHORIZATION
  NONE SPECIFIED
 SECURITY-LABEL=NONE SPECIFIED
```

Depending on your security requirements and policy, access to these profiles can be allowed to all User IDs defined on the system.

Note: A RACF User ID with the SPECIAL attribute is exempt from any access control checks to these profiles.

1.5.3 Determine RACF access to the R_PKIServ callable service

The R_PKIServ callable service is protected by new profiles in the RACF FACILITY class called IRR.RPKISERV.<function>.

The function we are securing is called GENCERT and EXPORT like a RACDCERT GENCERT. The authority given to a User ID controls subsequent RACDCERT access checks, as follows:

NONE Access to the callable service is denied.

This is also true if no profile is defined (RC4).

READ Access is permitted based on subsequent RACDCERT access checks against the client's (end user's) User ID.

UPDATE Access is permitted based on subsequent RACDCERT access checks against the daemon's User ID.

Note: This does not apply to the Web Server daemon. This is intended for future use.

CONTROL Access is permitted with no subsequent RACDCERT access checks made.

If you User ID is RACF SPECIAL then no subsequent checks are made as well.

The subsequent access control check are made against the RACF FACILITY class profile IRR.DIGTCERT.<function>, as discussed in 1.5.2, "Determine RACF access to the RACDCERT services" on page 17.

If you have choosen to use a surrogat User ID as suggested, then you need the following RACF authorizations:

```
PERMIT IRR.RPKISERV.GENCERT CLASS(FACILITY) ID(pkiserv) ACC(READ)

PERMIT IRR.RPKISERV.EXPORT CLASS(FACILITY) ID(userid) ACC(READ)
```

Note: PKISERV is our surrogat User ID.

Note: userid is any RACF User ID that is allowed to generate a certificate. The export function allows the user to retrieve the certificate.

1.5.4 Create your Certificate Authority (CA) Certificate

In order to create and sign digital certificates for others you need to define a Certificate Authority certificate and associated private key. This is done using the RACF RACDCERT GENCERT command. Before issuing the command, decide what the Certificate Authority's distinguished name will be. Typically, Certificate Authorities have distinguished names in the following form:

```
OU=<your-CA's-friendly-name>.O=<your-organization>.C=<your-2-letter-country-abbreviation>
```

The RACDCERT GENCERT command to create our CERTAUTH certificate for use in our examples is :

```
RACDCERT GENCERT CERTAUTH SUBJECTSDN(DN('IBM-ITSO PDG CA') OU('ITSO')
O('IBM') C('US')) NOTBEFORE(DATE(2000-09-25)) NOTAFTER(DATE(2005-09-24))
WITHLABEL('ITSO PDG CA'))
```

Note: Set the validity date of the CA certificate to more years then what you will use for your personal certificates, to avoid date inconsistencies.

You now need to export the CA certificate and publish it to your root directory, so it is available for download. Issue the following RACF command to export the CA certificate:

```
RACDCERT CERTAUTH EXPORT(LABEL('ITSO PDG CA')) FORMAT(CERTDER)
DSN('GRAAFF.EXPORT.CADER')
```

Next you need to copy it from the dataset created with the RACDCERT command to your HFS root directory you choose for the PKISERV application, like:

```
OPUT 'graaff.export.cader' '/usr/lpp/internet/etc/cacert.der' BINARY
```

For more information on the RACDCERT command, defining profiles, and granting access see the *SecureWay Security Server for OS/390 (RACF) Security Administrator's Guide,* SC28-1915 and *SecureWay Security Server for OS/390 (RACF) Command Language Language Reference,* SC28-1919.

1.5.5 Installing the sample code

We assume the IBM HTTP Server is installed and functional for at least normal non-SSL mode before going on to this step. The default location for the server root is:

```
/usr/lpp/internet/server_root
```

Depending where you want to install the PKISERV application, the root might be:

```
/usr/lpp/internet/server_root/PKIServ
```

Create the directory structure for PKIServ as defined in 1.2, “Directory structure provided” on page 2. Obtain the sample TAR file from the RACF website:
`http://www.s390.ibm.com/products/racf/webca.html`

Next do a file transfer in binary mode using FTP to OS/390. Then untar the file using the following command:

```
cd /usr/lpp/internet/server_root
```

```
tar -xvf PKISERV.tar
```

Remember to set the permission bits on all files. The permission bits for the executables can be the octal value of “644”. Permissions bits can be changed using the `CHMOD` command, as shown:

```
CHMOD 644 CAMAIN.REXX
```

The permissions bits for the configuration file `PKI.CONF` and the Certificate Authority (CA) certificate `CACERT.DER` can be set according to your installation’s security requirement.

1.5.6 Configure the IBM HTTP Server for SSL mode

The PKISERV application requires that the IBM HTTP Server operate in both normal and SSL mode. To be able to use SSL, your server will need to obtain a digital certificate. You can choose to purchase one from an external Certificate Authority (e.g. Verisign) or create one using RACF.

Note: If your server is already operating in SSL mode you can skip the next “Obtain a Certificate Using RACF” step.

Depending on your level of the IBM HTTP Server for OS/390, the utility you create a server certificate with is different. Prior to IHS Release 5.3, you use

IKEYMAN, with IHS Release 5.3 you may use GSKKYMAN or a RACF keyring using the RACDCERT ADDRING command.

1.5.6.1 Obtaining a Certificate for your Web Server

The procedure we describe next applies to the IBM HTTP Server (IHS) Release 5.3 that ships with OS/390 Version 2 Release 10. To be able to use SSL, your server needs a digital certificate to identify the server, like we did earlier when we created a digital certificate for you (an individual). IHS alows you to choose between two options:

1. create a digital certificate using a utility called GSKKYMAN, or
2. create a digital certificate using RACF, using the RACDCERT GENCERT command

Next the digital certificate of your server needs to be installed in a so called key ring, that is used by your server. The GSKKYMAN utility creates a key database or key ring for you. The RACF RACDCERT ADDRING and RACDCERT CONNECT commands can do simaliar functions. The difference between the two keyrings is the location of the key ring and the security of the key ring. The key ring created by GSKKYMAN is a UNIX file and the security relies on permission bits and a password to access the file. The RACF key ring as the name implies is stored in the RACF database, and requires RACF authorizations to be read and updated.

You need to decide who (what Certificate Authority Certificate) is signing the server's certificate, as we see later in this section. You may use the same CA certificate that signs the browser's (user) digital certificates.

1.5.6.2 Using GSKKYMAN to obtain a Server Certificate

1. Determine the location for your key database (key ring) and change to that directory, for example CD to /usr/lpp/internet/etc and invoke /usr/lpp/gskssl/bin/gskkyman
2. Choose option **1** to create a key database. Type in a name or let it default to key.kdb and enter the desired password. When asked to "work with the database now?" Enter **1** for yes.
3. Choose **3** - Create new key pair and certificate request. Answer the prompts for file name, label, key size (1024 recommended), and subject name fields. (Note, Common Name should be your server's symbolic IP address (e.g., www.ibm.com)). Exit GSKKYMAN when done.
4. From TSO, OGET the certificate request file to an MVS dataset, e.g.,

```
OGET certreq.arm '/usr/lpp/internet/etc/certreq.arm'
```

5. Use the RACF RACDCERT command to read the request and generate the server certificate, e.g.,

```
RACDCERT GENCERT(certreq.arm) ID(websrv) SIGNWITH(CERTAUTH LABEL('ITSO
PDG CA')) WITHLABEL('SSL Cert')
```

6. Export the new server certificate to a dataset then do an OPUT to the HFS file, e.g.,

```
RACDCERT EXPORT(LABEL('SSL Cert')) ID(websrv) DSN(cert.arm)
FORMAT(CERTB64)

OPUT cert.arm '/usr/lpp/internet/etc/cert.arm'
```

7. CD to /usr/lpp/internet/etc and invoke /usr/lpp/gskssl/bin/gskkyman again
8. Choose option **2** to open the key database you created before. Reply to the name and password prompts.
9. Choose option **6** to Store a CA certificate and specify the '/usr/lpp/internet/etc/cacert.der' file. When asked to "exit ikeyman?" Enter **0** for No.
10. Choose option **4** to Receive a certificate issued for your request and specify the '/usr/lpp/internet/etc/cert.arm' file. Again enter **0** when asked to "exit ikeyman?"
11. Choose option 11 to Store encrypted database password

1.5.6.3 Using RACF to obtain a Server Certificate

The following steps detail the setup of a RACF key ring and a server certificate for your Web server:

1. The User ID associated with the IHS started task needs to be authorized to the following RACF FACILITY class profiles:

```
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(websrv) ACCESS(CONTROL)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ID(websrv) ACCESS(CONTROL)
```

2. Next we need to set up the RACF key ring for use by the Web server:

```
RACDCERT ID(websrv) ADDRING(WEB57)
```

Note: You need RACF SPECIAL to be able to define the RACF key ring.

3. You generate a server certificate for your Web server, using the RACF RACDCERT command, as shown:

```
RACDCERT ID(websrv) GENCERT SUBJECTSDN(CN('WTSC57.ITSO.IBM.COM) O('IBM')
OU('ITSO') L('Poughkeepsie') C('US') SP('New York')) SIZE(1024)
WITHLABEL('WEB57CERT') SIGNWITH(CERTAUTH LABEL('ITSO PDG CA'))
```

4. You then need to connect the server certificate and the Certificate Authority certificate to the keyring we created in step 2, usin the RACF RACDCERT CONNECT command, as shown:

```
RACDCERT ID(websrv) CONNECT(ID(websrv) LABEL('WEB57CERT') RING(WEB57)
DEFAULT USAGE(PERSONAL)
RACDCERT ID(websrv) CONNECT(CERTAUTH LABEL('ITSO PDG CA') RING(WEB57)
USAGE(CERTAUTH)
```

5. Configure your Webserver to use the RACF keyring. (APAR PQ39311 provides the function to do this)

1.5.6.4 Turning on SSL mode

To turn on SSL in your Web server, you need to make the following changes in your configuration file HTTPD.CONF:

1. set SSLMODE ON
2. designate a port for SSL usage, like SSLPORT 443
3. set the parameter KEYFILE to indicate either the UNIX file name for the key ring to be used like. KEYFILE /usr/lpp/internet/etc/key.kdb or indicate the label name of the RACF keyring to be used.

1.5.7 Other changes to make to the HTTPD.CONF file

The following directives are to be added to the HTTPD.CONF file to be able to use the PKISERV application:

```
Protection PublicUser {                                    1
        ServerId        PublicUser
        UserID          PUBLIC
        Mask            Anyone
}

Protect /PKIServ/public-cgi/*   PublicUser
Protect /PKIServ/ssl-cgi-bin/*  PublicUser
Protect /PKIServ/*              PublicUser

Protection AuthenticatedUser {                          2
        ServerId        AuthenticatedUser
        AuthType      Basic
        PasswdFile    %%SAF%%
        UserID        %%CLIENT%%
        Mask             All
}
Protection SurrogateUser {                                 3
        ServerId       SurrogateUser
        AuthType       Basic
        PasswdFile     %%SAF%%
        UserID         PKISERV
        Mask           All
}

Protect /PKIServ/ssl-cgi-bin/auth/*          AuthenticatedUser
Protect /PKIServ/ssl-cgi-bin/auth/careq.rexx SurrogateUser

Redirect  /PKIServ/ssl-cgi/*      https://<server-domain-name>/PKIServ/ssl-cgi-bin/*
Exec      /PKIServ/public-cgi/*  <server-root>/PKIServ/public-cgi/*
Exec      /PKIServ/ssl-cgi/*       <server-root>/PKIServ/ssl-cgi-bin/*
Exec      /PKIServ/ssl-cgi-bin/* <server-root>/PKIServ/ssl-cgi-bin/*
Pass      /PKIServ/*                  <server-root>/PKIServ/*

AddType  .cer      application/x-x509-user-cert      ebcdic  0.5 # Browser Certificate
AddType  .der      application/x-x509-ca-cert         binary  1.0 # CA Certificate
```

Figure 13. HTTPD.CONF file required changes

The setup requires three protection setups:

PublicUser (1) used for the “public” resources

AuthenticatedUser (2) used to determine the client requesting the certificate.

SurrogatUser (3) used for the actual generation of the certificate, because we do not want every user to be authorized with CONTROL authority to the IRR.DIGTCERT.GENCERT profile in the RACF FACILITY class.

Note: if you choose not to use a surrogat User ID to issue certificates, then replace *SurrogatUser* with the *AuthenticatedUser*.

The `PROTECT`, `PASS` and `EXEC` directives are their to insure the proper protection and execution of the PKISERV application.

There is one `REDIRECT` directive to insure that SSL is invoked for the PKISERV application.

1.5.8 Customizing the PKI.CONF file

Depending on your installation’s needs and security requirements, you may decide to change your configuration file `PKI.CONF`. One change that is definitely required is to indicate the label of the Certificate Authority Certificate that is going to sign the certificates requested.

In the <constant> section of the each <template>, the `%%SignWith=%%` parameter indicates the label of the CERTAUTH certificate that signs the certificate issued with the PKISERV application. In our example we use the CERTAUTH certificate generated earlier, like `%%SignWith=SAF:CERTAUTH/ITSO PDG CA%%`.

Another area where you may want to change things is the information necessary to build the browser’s certificate. The only two fields displayed today are the label and public key type (CSP) on the certificate request page, as shown in Figure 4 on page 9. The following fields can be added to the request page to provide us with more meaningfull information:

CommonName the name of the individual, like Paul de Graaff

Note: If you leave the `%%CommonName%%` field in the constant section, it uses the `NAME` field from your RACF User profile as the Common Name.

OrgUnit the organizational unit the individual resides, like ITSO

Org	the organization the indivudual works for, like IBM
Locality	the locality, like Poughkeepsie
StateProv	the State or Province you live in, like New York
Country	the country you live in, abrreviated to two charcaters, like US
AltEmail	your email address, like graaff@us.ibm.com
AltDomain	your internet domain, like www.ibm.com
AltURI	your universal resource identifier, like wtsc57.itso.ibm.com
AltIPAddr	numeric IP address, like 9.12.14.247

Depending on your security requirements, you may have the user enter his/her, CommonName, Org, OrgUnit and Locality. To make these changes, you need to edit the `PKI.CONF` file.

Figure 14 shows part of the `PKI.CONF` file, before we made the changes to include the new fields on the certificate request page.

```
<TEMPLATE NAME=1 Year SAF Browser Certificate>
.........
<p> Enter values for the following field(s)
 %%Label%%
 %%PublicKeyÝbrowsertype¨%%
.......
<CONSTANT>
 %%Org=IBM%%
 %%OrgUnit=ITSO%%
 %%KeyUsage=handshake%%
 %%NotAfter=365%%
 %%Country=US%%
 %%SignWith=SAF:CERTAUTH/ITSO PDG CA%%
</CONSTANT>
```

Figure 14. PKI.CONF file before changes

To make the changes, you need to move the fields you want to include in the certificate request page from the <constant> section to the line under “Enter values for the following Field(s)” and remove the value specified, as shown in Figure 15 on page 28.

```
<TEMPLATE NAME=1 Year SAF Browser Certificate>
.........
<p> Enter values for the following field(s)
 %%Label%%
 %%CommonName%%
 %%Org%%
 %%OrgUnit%%
 %%PublicKeyÝbrowsertype¨%%
<CONSTANT>
 %%KeyUsage=handshake%%
 %%NotAfter=365%%
 %%Country=US%%
 %%SignWith=SAF:CERTAUTH/ITSO PDG CA%%
</CONSTANT>
```

Figure 15. PKI.CONF file after changes

When you have made your changes, save them and invoke the PKISERV application again. When you make the request for a browser certificate, you will see the following screen, including the new fields added, as shown in Figure 16 on page 28.

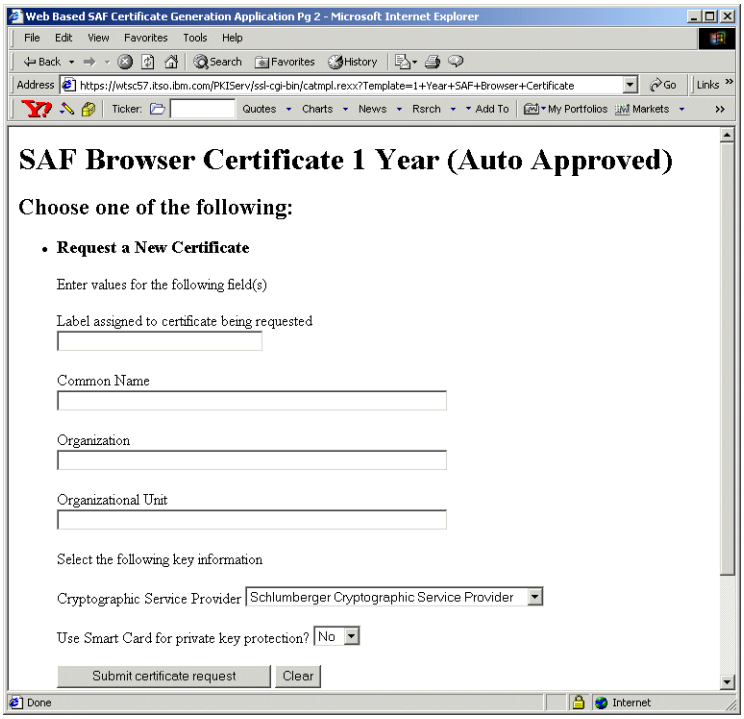

Figure 16. Certificate request page after PKI.CONF updates