IBM ITSO Poughkeepsie
OS/390 in an e-business environment

IBM HTTP Server 5.1 Secure Server Setup

Roland Trauner
trauner@us.ibm.com

Generate a Server Certificate

- **Server Certificate required to enable SSL**
  - A certificate is a public key (asymmetric / public-private key processing) signed by a Certificate Authority.
  - A Certificate Authority is usually a trusted third party that guarantees the authentication of a certificate holder. Many countries have legal processes established regarding CA's.
  - The OS/390 Web server is able to self-sign server certificates.
  - IBM HTTP Server 5.1 can also act as a CA.
    - IKEYMAN is used to handle server certificates for the HTTP server
    - HTTP Server CA function used to sign 3rd party (user and server) certificates

Roland Trauner trauner@us.ibm.com

Generate a Server Certificate

- **IBM HTTP Server 5.1 IKEYMAN Utility**
  - is a line mode utility issued from the UNIX shell
- **IKEYMAN can be used to:**
  - Migrate DGW 4.6.1 or older key ring into a DGW 5.0 or HTTP Server key database
  - Create a key database
  - Create server keys
  - Create a CA certificate
  - Create a certificate request
  - Self sign certificates
  - Receive CA signed certificated into the key database

Roland Trauner trauner@us.ibm.com

IKEYMAN Utility

- **IKEYMAN utility**
  - is located in /usr/lpp/internet/bin
  - Default: Permission bits 751; Owner WEBADM; Group IMWEB
    - Can be used without superuser authority
  - To execute IKEYMAN you need to prepare the environment
    - Set up PATH, LIBPATH and NLSPATH environment variables
  - If you plan to use IKEYMAN frequently, modify your user profile to enable it there. To do so add the following definitions to your *.profile* dataset.

```
export PATH=$PATH:/usr/lpp/internet/bin
export LIBPATH=$LIBPATH:/usr/lpp/internet/bin
export NLSPATH=$NLSPATH:/usr/lpp/internet/%L/%N
```


Roland Trauner trauner@us.ibm.com

- Superuser or WEBADM authority no longer needed to execute IKEYMAN
- HTTP Server 5.1 runs out of the linklib - no steplib needed

Migrate a Key Ring file

- **Migrate a key ring to a key database**
  - The easiest way to generate HTTP Server keys and certificates is to migrate an existing (DGW 4.6.1 or older) key ring.
  - Just one command:
    - ikeyman -m -r *keyringname*

```
TRAUNER:/web/apple/sec: >ikeyman -m -r apple.kyr
Enter password for the keyring file.......> secret

Please wait while the keyring is converted to a key
database......

Your request has completed sucessfully
```

    - you now created apple.kdb
    - If you copied the keyring to a new directory, be sure to copy the *"stash"* file as well

Roland Trauner trauner@us.ibm.com

Building Certificates

- **Building self signed certificates**
  - IKEYMAN provides a simple way to create a self signed certificate.
  - This certificate, which is in fact a CA (or root) certificate can be used as an operational server certificate.
  - From an organization and hierarchy point of view it is more clear however to create a CA certificate and an operational server certificate signed by the CA.
  - Both methods will be explained:

Roland Trauner trauner@us.ibm.com

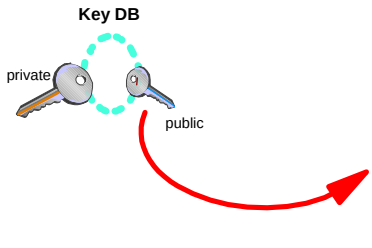
Building Certificates

- **Method 1:**
  - Create just one self signed certificate
- **Method 2:**
  - Create a CA and a server certificate
  - this method is also used when receiving certificates from an external CA

Roland Trauner trauner@us.ibm.com

Building Certificates

Building a self signed certificate to be used as CA and operational server certificate



1. Use ikeyman to create a key database.
2. Use ikeyman to create a self signed certificate.

Roland Trauner trauner@us.ibm.com

Create a new key database

- **Use IKEYMAN**
  - Option 1 Create a new key database
  - Name Use a name to identify this file as the server key database
  - Password Select a password --- and don't forget it
  - Next selection select option 5 to create a self signed certificate

Roland Trauner trauner@us.ibm.com

Create a new key database

```
            IBM Key Management Utility

Choose one of the following options to proceed.

   1  - Create new key database
   2  - Open key database
   3  - Change database password

   0  - Exit program

Enter your option number: 1
Enter key database name or press ENTER for "key.kdb": apple1.kdb
Enter password for the key database.......> secret
Enter password again for verification.....> secret
Should the password expire? (1 = yes, 0 = no) [1]: 0

The database has been successfully created, do you want to continue to work with
 the database now? (1 = yes, 0 = no) [1]: 1
```


Roland Trauner trauner@us.ibm.com

Create a self signed certificate

```
            Key database menu

Current key database is /web/apple/sec/apple1.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 5
```


Roland Trauner trauner@us.ibm.com

Certificate Data

- **Define the certificate content**
  - Version number determines the level of the certificate. We suggest to use 3 (newest and latest)
  - Enter a label for the key. Select a name that will associate the certificate being the server certificate
  - Select the desired keysize
    - 512 bit is less secure but faster than 1024 bit
    - 1024 bit is more secure but requires more cryptographic resources

Roland Trauner trauner@us.ibm.com

Certificate Data

- **Define the certificate content --- cont.**
  - Define the certificate fields to define the Distinguished Name (DN)
    - Common Name

      The Common Name usually is the servers Domain Name. This field will be checked against the actual URL and a warning will be issued if they don't match.
    - Organization
    - Organization Unit
    - City/Locality
    - State Province and Country

Roland Trauner trauner@us.ibm.com

Certificate Data

- **Define the certificate content --- cont.**
  - Define when the certificate will expire.
  - Select that this key will become the default key in your key database
  - Store the certificate in a file.
  - I does not matter which format you choose - we took ASCII.
  - Give it a name
- **Now your self signed certificate is created**

Roland Trauner trauner@us.ibm.com

CA Certificate Data

```
Enter version number of the certificate to be created (1, 2, or 3) [3] 3
Enter a label for this key.............> IBM ITSO Pok The Apple Server 1
Select desired key size from the following options (512):
    1:    512
    2:    1024
Enter the number corresponding to the key size you want: 1
Enter certificate subject name fields in the following.
    Common Name (required)..............> wtsc59oe.itso.ibm.com
    Organization (required).............> IBM
    Organization Unit (optional)........> ITSO Pougkeepsie
    City/Locality (optional)............> Poughkeepsie
    State/Province (optional)...........> New York
    Country Name (required 2 characters)..> US
Enter number of valid days for the certificate [365]: 7000
Do you want to set the key as the default in your key database? (1 = yes, 0 = no)
[1]: 1
Do you want to save the certificate to a file? (1 = yes, 0 = no) [1]: 1
Should the certificate binary data or Base64 encoded ASCII data be saved? (1 =
ASCII, 2 = binary) [1]: 1
Enter certificate file name or press ENTER for "cert.arm": apple1.arm

Please wait while self-signed certificate is created...

Your request has completed successfully, exit ikeyman? (1 = yes, 0 = no) [0]: 0
```


Roland Trauner trauner@us.ibm.com

Store encrypted database password

- **For certain functions it is required to have the encrypted database password in a separate file**
  - Create this so called "stash file".
  - File needs to be in the same directory as the key database itself.
  - The name needs to be associated with the key database.

Roland Trauner trauner@us.ibm.com

Store encrypted database password

```
            Key database menu

Current key database is /web/apple/sec/apple1.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 11

The encrypted password has been stored in file /web/apple/sec/apple1.sth

Your request has completed sucessfully, exit ikeyman? (1 = yes, 0 = no) [0]: 1
```


Roland Trauner trauner@us.ibm.com

Create a certificate

- **Certificate creation finished**

```
TRAUNER:/web/apple/sec: >ls -al
total 512
drwxrwx---   2 TRAUNER  SYS1        8192 Jun  7 11:11 .
drwxr-xr-x   9 TRAUNER  SYS1        8192 Jun  7 09:50 ..
-rw-r-----   1 TRAUNER  SYS1         729 Jun  7 10:40 apple1.arm
-rw-r-----   1 TRAUNER  SYS1       70080 Jun  7 10:40 apple1.kdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 10:40 apple1.sth
```

- **Key database can be used to SSL-enable the web server**
  - httpd.conf configuration directives are described later

Roland Trauner trauner@us.ibm.com

Building Certificates

- **Method 1:**
  - Create just one self signed certificate
- **Method 2:**
  - Create a CA and a server certificate
  - this method is also used when receiving certificates from an external CA

Roland Trauner trauner@us.ibm.com

Building Certificates

- **Two certificates - CA and server**
  - This method describes the process if we like to have two separated certificates. A CA and an operational server certificate.
  - Create a Certification Authority (CA) certificate
  - Create a Web server key pair and certificate request (operational key)
  - Sign the server certificate request using the CA
  - Receive the server certificate into the key database

Roland Trauner trauner@us.ibm.com

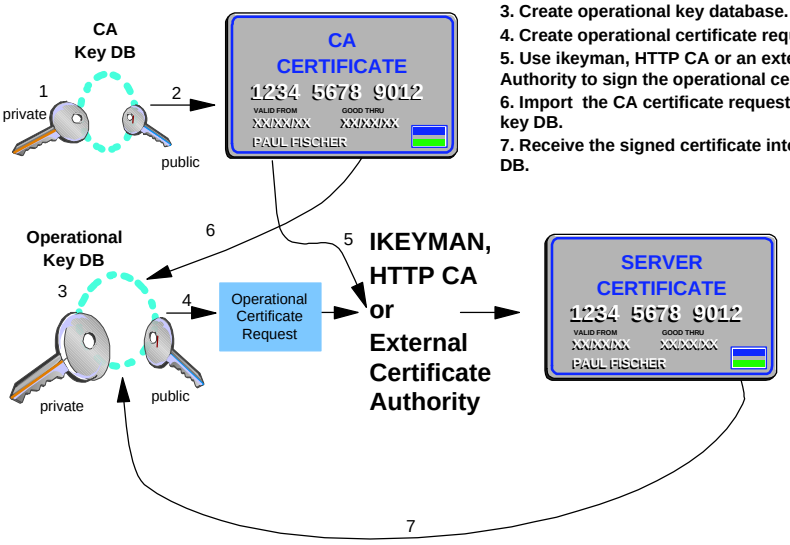
Building Certificates

- **Two certificates - CA and server**
  - This method describes the process if we like to have two separated certificates. A CA and an operational server certificate.
  - Create a Certification Authority (CA) certificate
  - Create a Web server key pair and certificate request (operational key)
  - Sign the server certificate request using the CA
  - Receive the server certificate into the key database

Roland Trauner trauner@us.ibm.com

Building Certificates

HTTP Server Self-signed Certificate Complete Flow



1. Create CA key database.
2. Create self signed CA certificate.
3. Create operational key database.
4. Create operational certificate request.
5. Use ikeyman, HTTP CA or an external Certificate Authority to sign the operational certificate request.
6. Import the CA certificate request into the operational key DB.
7. Receive the signed certificate into the operational key DB.

Roland Trauner trauner@us.ibm.com

Create a new key database

- **Use IKEYMAN**
  - Option 1 Create a new key database
  - Name Use a name to identify this file as the CA key database
  - Password Select a password --- and don't forget it
  - Next selection select option 5 to create a self signed certificate

Roland Trauner trauner@us.ibm.com

Create a new key database

```
            IBM Key Management Utility

Choose one of the following options to proceed.

   1  - Create new key database
   2  - Open key database
   3  - Change database password

   0  - Exit program

Enter your option number: 1
Enter key database name or press ENTER for "key.kdb": itsoca.kdb
Enter password for the key database.......> secret
Enter password again for verification.....> secret
Should the password expire? (1 = yes, 0 = no) [1]: 0

The database has been successfully created, do you want to continue to work with
 the database now? (1 = yes, 0 = no) [1]: 1
```


Roland Trauner trauner@us.ibm.com

Create a self signed certificate

```
            Key database menu

Current key database is /web/apple/sec/itsoca.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 5
```


Roland Trauner trauner@us.ibm.com

CA Certificate Data

- **Define the CA certificate content**
  - Version number determines the level of the certificate. We suggest to use 3 (newest and latest)
  - Enter a label for the key. Select a name that will associate the certificate being a CA certificate
  - Select the desired keysize
    - 512 bit is less secure but faster than 1024 bit
    - 1024 bit is more secure but requires more cryptographic resources

Roland Trauner trauner@us.ibm.com

CA Certificate Data

- **Define the CA certificate content cont.**
  - Define the certificate fields to define the Distinguished Name (DN)
    - Common Name

      The Common Name usually is the servers Domain Name. This field will be checked against the actual URL and a warning will be issued if they don't match.

      Since we define a CA certificate, the rules are different because CA certificates are usually not bound to URLs
    - Organization
    - Organization Unit
    - City/Locality
    - State Province and Country

Sample CA Certificate:

```
This Certificate belongs to:
  IBM World Registry Certification Authority
  IBM World Registry
  US

This Certificate was issued by:
  IBM World Registry Certification Authority
  IBM World Registry
  US

Serial Number: 33:82:0A:D2
This Certificate is valid from Tue May 20, 1997
to Sat May 20, 2017
Certificate Fingerprint:

7C:73:0A:91:E2:FF:94:34:93:36:FE:B0:35:30:82:4F
```


Roland Trauner trauner@us.ibm.com

CA Certificate Data

- **Define the CA certificate content cont.**
  - Define when the CA certificate will expire.
    - Bear in mind that when the CA certificate becomes invalid, all certificates signed by this CA also become invalid.
  - Select that this key will become the default key in your key database
  - Store the certificate in a file.
  - I does not matter which format you choose - we took ASCII.
  - Give it a name
- **Now your self signed CA certificate is created**

Roland Trauner trauner@us.ibm.com

CA Certificate Data

```
Enter version number of the certificate to be created (1, 2, or 3) [3] 3
Enter a label for this key.............> IBM ITSO Pok Apple CA on wtsc59
Select desired key size from the following options (512):
    1:    512
    2:    1024
Enter the number corresponding to the key size you want: 1
Enter certificate subject name fields in the following.
    Common Name (required).............> IBM ITSO Poughkeepsie - The Apple CA
    Organization (required)............> IBM
    Organization Unit (optional).......> IBM ITSO Poughkeepsie
    City/Locality (optional)...........> Poughkeepsie
    State/Province (optional)..........> NY
    Country Name (required 2 characters)..> US
Enter number of valid days for the certificate [365]: 7000
Do you want to set the key as the default in your key database? (1 = yes, 0 = no)
[1]: 1
Do you want to save the certificate to a file? (1 = yes, 0 = no) [1]: 1
Should the certificate binary data or Base64 encoded ASCII data be saved? (1 =
ASCII, 2 = binary) [1]: 1
Enter certificate file name or press ENTER for "cert.arm": itsoca.arm

Please wait while self-signed certificate is created...

Your request has completed successfully, exit ikeyman? (1 = yes, 0 = no) [0]: 0
```


Roland Trauner trauner@us.ibm.com

Store encrypted database password

- **For certain functions it is required to have the encrypted CA database password in a separate file**
  - Create this so called "stash file".
  - File needs to be in the same directory as the key database itself.
  - The name needs to be associated with the key database.

Roland Trauner trauner@us.ibm.com

Store encrypted database password

```
            Key database menu

Current key database is /web/apple/sec/itsoca.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 11

The encrypted password has been stored in file /web/apple/sec/itsoca.sth

Your request has completed sucessfully, exit ikeyman? (1 = yes, 0 = no) [0]: 1
```


Roland Trauner trauner@us.ibm.com

Create a certificate

- CA Certificate creation finished
- Key database can be used to sign certificate requests

Roland Trauner trauner@us.ibm.com

Building Certificates

- **Two certificates - CA and server**
  - This method describes the process if we like to have two separated certificates. A CA and an operational server certificate.
  - Create a Certification Authority (CA) certificate
  - Create a Web server key pair and certificate request (operational key)
  - Sign the server certificate request using the CA
  - Receive the server certificate into the key database

Roland Trauner trauner@us.ibm.com

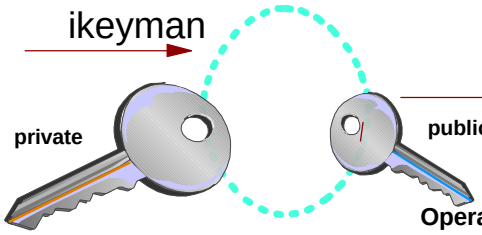
Building Certificates

- This step is the one that is required if you are not using your own, but an "official" CA like Verisign.
- In that case, you create a certificate request and use a procedure of the signer to sign it.

Roland Trauner trauner@us.ibm.com

Building Certificates

Creating an Operational Key DB and Certificate Request



ikeyman

Operational Certificate Request

Must be signed (in next step) before use.

Operational Key Database

The operational key database is created using ikeyman and is a completely separate key database from the CA self-certified key database.

Roland Trauner trauner@us.ibm.com

Create a new key database

- **Use IKEYMAN**
  - Option 1 — Create a new key database
  - Name — Use a name to identify this file as the server key database
  - Password — Select a password --- and don't forget it
  - Next selection — select option 3 to create a new key pair and a certificate request

Roland Trauner trauner@us.ibm.com

Create a new key database

```
            IBM Key Management Utility

Choose one of the following options to proceed.

   1  - Create new key database
   2  - Open key database
   3  - Change database password

   0  - Exit program

Enter your option number: 1
Enter key database name or press ENTER for "key.kdb": apple2.kdb
Enter password for the key database.......> secret
Enter password again for verification.....> secret
Should the password expire? (1 = yes, 0 = no) [1]: 0

The database has been successfully created, do you want to continue to work with
 the database now? (1 = yes, 0 = no) [1]: 1
```


Roland Trauner trauner@us.ibm.com

Create a self signed certificate

```
            Key database menu

Current key database is /web/apple/sec/apple2.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 3
```


Roland Trauner trauner@us.ibm.com

Server Certificate Data

- **Define the Web server certificate content**
  - Define a name for the certificate request.
  - Enter a label for the key.
    Select a name that will associate the certificate being a Web server certificate.
  - Select the desired keysize
    - 512 bit is less secure but faster than 1024 bit
    - 1024 bit is more secure but requires more cryptographic resources

Roland Trauner trauner@us.ibm.com

Server Certificate Data

- **Define the web server certificate content --- cont.**
  - Define the certificate fields to define the Distinguished Name (DN)
    - Common Name

      The Common Name usually is the servers Domain Name. This field will be checked against the actual URL and a warning will be issued if they don't match.
    - Organization
    - Organization Unit
    - City/Locality
    - State Province and Country
  - Be sure that the Distinguished Name of the server certificate is not exactly the same as the CA certificate, If this happens, you are unable to sign the server certificate.
- **Finish --- Server certificate request created**

Roland Trauner trauner@us.ibm.com

Server Certificate Data

```
Enter certificate request file nameor press ENTER for "certreq.arm": apple2.arm
Enter a label for this key.............> Apple Server Cert 2
Select desired key size from the following options (512):
    1:    512
    2:    1024
Enter the number corresponding to the key size you want: 1
Enter certificate subject name fields in the following.
    Common Name (required).............> wtsc59oe.itso.ibm.com
    Organization (required)............> IBM ITSO Poughkeepsie
    Organization Unit (optional).......> HTTP Server /390 - The Apple
    City/Locality (optional)...........> Poughkeepsie
    State/Province (optional)..........> New York
    Country Name (required 2 characters)..> US

Please wait while key pair is created...

Your request has completed successfully, exit ikeyman? (1 = yes, 0 = no) [0]: 0
```


Roland Trauner trauner@us.ibm.com

Store encrypted database password

- **For certain functions it is required to have the encrypted key database password in a separate file**
  - Create this so called "stash file".
  - File needs to be in the same directory as the key database itself.
  - The name needs to be associated with the key database.

Roland Trauner trauner@us.ibm.com

Store encrypted database password

```
            Key database menu

Current key database is /web/apple/sec/apple2.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 11

The encrypted password has been stored in file /web/apple/sec/apple2.sth

Your request has completed sucessfully, exit ikeyman? (1 = yes, 0 = no) [0]: 1
```


Roland Trauner trauner@us.ibm.com

Create a Web server certificate

- **Certificate request creation finished**

```
TRAUNER:/web/apple/sec: >ls -al
total 512
drwxrwx---   2 TRAUNER  SYS1        8192 Jun  7 11:11 .
drwxr-xr-x   9 TRAUNER  SYS1        8192 Jun  7 09:50 ..
-rw-r--r--   1 TRAUNER  SYS1         554 Jun  7 11:09 apple2.arm
-rw-r--r--   1 TRAUNER  SYS1       65080 Jun  7 11:02 apple2.kdb
-rw-r--r--   1 TRAUNER  SYS1        5080 Jun  7 11:09 apple2.rdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:11 apple2.sth
-rw-r--r--   1 TRAUNER  SYS1         765 Jun  7 11:00 itsoca.arm
-rw-r--r--   1 TRAUNER  SYS1       70080 Jun  7 11:00 itsoca.kdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:00 itsoca.sth
```

- **Certificate request now needs to be signed**
- **Signed certificate needs to be received into the key database**

Roland Trauner trauner@us.ibm.com

Building Certificates

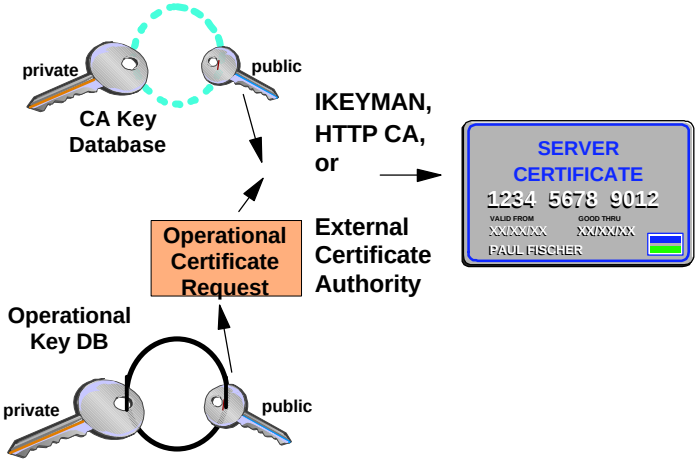
- **Two certificates - CA and server**
  - This method describes the process if we like to have two separated certificates. A CA and an operational server certificate.
  - Create a Certification Authority (CA) certificate
  - Create a Web server key pair and certificate request (operational key)
  - Sign the server certificate request using the CA
  - Receive the server certificate into the key database

Roland Trauner trauner@us.ibm.com

Building Certificates

Signing the Operational Certificate

You use the CA key and the operational certificate request to generate a signed operational certificate



Roland Trauner trauner@us.ibm.com

Sign a Certificate Request

- **Sign the server certificate request**
  - The following IKEYMAN command can be used to sign the server certificate request:
    - ikeyman -g -cr *server-certificate-request-file* -ct *server-certificate-file* -k *CA-keydatabase*

```
TRAUNER:/web/apple/sec: >ikeyman -g -cr apple2.arm -ct
apple2.cert -k itsoca.kdb
Enter password for the key database.......> secret

Please wait while a certificate is generated for the
request......

Your request has completed sucessfully!
```

    - now you created apple2.cert which needs to be received into the apple2.kdb --- the Web server keydatabase

Roland Trauner trauner@us.ibm.com

Building Certificates

- **The Certificate file**

```
TRAUNER:/web/apple/sec: >ls -al
total 520
drwxrwx---   2 TRAUNER  SYS1        8192 Jun  7 11:24 .
drwxr-xr-x   9 TRAUNER  SYS1        8192 Jun  7 09:50 ..
-rw-r--r--   1 TRAUNER  SYS1         554 Jun  7 11:09 apple2.arm
-rw-r--r--   1 TRAUNER  SYS1         806 Jun  7 11:24 apple2.cert
-rw-r--r--   1 TRAUNER  SYS1       65080 Jun  7 11:02 apple2.kdb
-rw-r--r--   1 TRAUNER  SYS1        5080 Jun  7 11:09 apple2.rdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:11 apple2.sth
-rw-r--r--   1 TRAUNER  SYS1         765 Jun  7 11:00 itsoca.arm
-rw-r--r--   1 TRAUNER  SYS1       70080 Jun  7 11:00 itsoca.kdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:00 itsoca.sth
TRAUNER:/web/apple/sec: >
```


Roland Trauner trauner@us.ibm.com

Building Certificates

The Certificate

```
TRAUNER:/web/apple/sec: >cat apple2.cert
-----BEGIN CERTIFICATE-----
MIICJzCCAdGgAwIBAgIP+Pn58vn0+fL49/b58PX4MA0GCSqGSIb3DQEBBAUAMIGK
MQswCQYDVQQGEwJVUzELMAkGA1UECBMCTlkxFTATBgNVBAcTDFBvdWdoa2VlcHNp
ZTEMMAoGA1UEChMDSUJNMRowGAYDVQQLExFJVFNPIFBvdWdoa2VlcHNpZTEtMCsG
A1UEAxMkSUJNIElUU08gUG91Z2hrZWVwc2llIC0gVGhlIEFwcGxlIENBMB4XDTk5
MDYwNjE1MjQyOFoXDTAwMDYwNjE1MjQyOFowgZ4xCzAJBgNVBAYTAlVTMREwDwYD
VQQIEwhOZXcgWW9yazEVMBMGA1UEBxMMUG91Z2hrZWVwc2llMR4wHAYDVQQKExVJ
Qk0gSVRTTyBQb3VnaGtlZXBzaWUxJTAjBgNVBAsTHEhUVFAgU2VydmVyIC8zOTAg
LSBUaGUgQXBwbGUxHjAcBgNVBAMTFXd0c2M1OW9lLml0c28uaWJtLmNvbTBcMA0G
CSqGSIb3DQEBAQUAA0sAMEgCQQCUk9tLujUtaosNm9o//zds9mMqkBOGm2SuFXfA
q6YVCU3OdWUXC5tRN3CBVuFJnxq0012fnOzYQgsQ99/V1pPTAgMBAAEwDQYJKoZI
hvcNAQEEBQADQQAhuqbquBdACNWy9UMQV+O0RKZCPxo3FDqYPjY8nJRSMlejSQBx
dySXI1tNW+SNQ2AMg4G0/byBS1RAvRZtvM+c
-----END CERTIFICATE-----
TRAUNER:/web/apple/sec: >
```


Roland Trauner trauner@us.ibm.com

Building Certificates

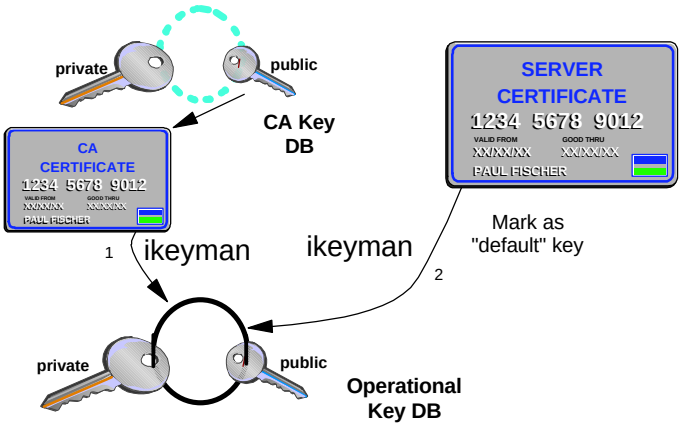
- **Two certificates - CA and server**
  - This method describes the process if we like to have two separated certificates. A CA and an operational server certificate.
  - Create a Certification Authority (CA) certificate
  - Create a Web server key pair and certificate request (operational key)
  - Sign the server certificate request using the CA
  - Receive the server certificate into the key database

Roland Trauner trauner@us.ibm.com

Building Certificates

Importing and Receiving the Signed Certificates into the Operational Key DB

IKEYMAN is used to import the CA's certificate into the operational key database and to receive the signed operational certificate into the operational key database.



Roland Trauner trauner@us.ibm.com

Receive a Certificate

- **Use IKEYMAN**
  - Option 2 Open the operational key database
  - Password hope you still know it
  - Option 8 Import keys --- the CA public key
    - Receive the keys from another keydatabase if you act as your own CA
    - Receive the keys from a PKCS12 file - if you received the certificate from an external CA
    - Some external CA keys are already in the key database and need not to be received.
  - Option 4 Receive a certificate issued for your request
    - Specify the name of the certificate to receive
    - Define it to be the default key in the key database

Roland Trauner trauner@us.ibm.com

Receive a Certificate

```
            IBM Key Management Utility

Choose one of the following options to proceed.

   1  - Create new key database
   2  - Open key database
   3  - Change database password

   0  - Exit program

Enter your option number: 2
Enter key database name or press ENTER for "key.kdb": apple2.kdb
Enter password for the key database.......> secret
```


Roland Trauner trauner@us.ibm.com

Receive the CA certificate

```
            Key database menu

Current key database is /web/apple/sec/apple2.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 8
```


Roland Trauner trauner@us.ibm.com

Receive the CA certificate

```
            Import Key Menu

Current key database is /web/apple/sec/apple2.kdb

     1  - Import keys from another key database
     2  - Import keys from a PKCS12 file

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 1
Enter the key database name to be imported: itsoca.kdb
Enter the password for the key database to be imported: secret
```


Roland Trauner trauner@us.ibm.com

Receive the CA certificate

```
            Key and certificate lists of the import key database

Key database name is /web/apple/sec/apple2.kdb

Please choose one of the following keys to work with.

     1  - IBM ITSO Pok APPLE CA on wtsc59
     2  - Integrion Certification Authority Root
     3  - IBM World Registry Certification Authority
     4  - Thawte Personal Premium CA
     5  - Thawte Personal Freemail CA
     6  - Thawte Personal Basic CA
     7  - Thawte Premium Server CA
     8  - Thawte Server CA
     9  - Verisign Test CA Root Certificate

Enter the numbers of the keys that you want to import (for example, 2,5) or press
ENTER for more labels: 1
Your request has been completed successfully, exit ikeyman (1 = yes, 0 = no) [0]:
0
```


Roland Trauner trauner@us.ibm.com

Receive the CA certificate

This is the page you get if you press ENTER

```
            Key and certificate lists of the import key database

Key database name is /web/apple/sec/apple2.kdb

Please choose one of the following keys to work with.

    10  - RSA Secure Server Certification Authority
    11  - Verisign Class 1 Public Primary Certification Authority
    12  - Verisign Class 2 Public Primary Certification Authority
    13  - Verisign Class 3 Public Primary Certification Authority
    14  - Verisign Class 4 Public Primary Certification Authority

Enter the numbers corresponding to the keys that you want to import (for example
, 2,5)
 or press ENTER to the previous list:
 ===>
```


Roland Trauner trauner@us.ibm.com

Receive the CA certificate

```
           Key database menu
Current key database is /web/apple/sec/apple2.kdb

     1  - List/Manage keys and certificates
     2  - List/Manage request keys
     3  - Create new key pair and certificate request
     4  - Receive a certificate issued for your request
     5  - Create a self-signed certificate
     6  - Store a CA certificate
     7  - Show the default key
     8  - Import keys
     9  - Export keys
    10  - List all trusted CAs
    11  - Store encrypted database password

     0  - Exit program

Enter option number (or press ENTER to return to the parent menu): 4
Enter certificate filename or press ENTER for "cert.arm": apple2.cert
Do you want to set the key as the default in your key database? (1 = yes, 0 = no)
[1]: 1

Please wait while certificate is received......

Your request has completed successfully, exit ikeyman? (1 = yes, 0 = no) [0]: 1
```


Roland Trauner trauner@us.ibm.com

Create a Web server certificate

- **CA keys received into key database**
- **Certificate received into key database**

```
TRAUNER:/web/apple/sec: >ls -al
total 1040
drwxrwx---   2 TRAUNER  SYS1        8192 Jun  7 11:36 .
drwxr-xr-x   9 TRAUNER  SYS1        8192 Jun  7 09:50 ..
-rw-r--r--   1 TRAUNER  SYS1         554 Jun  7 11:09 apple2.arm
-rw-r--r--   1 TRAUNER  SYS1         806 Jun  7 11:24 apple2.cert
-rw-r--r--   1 TRAUNER  SYS1       75080 Jun  7 11:36 apple2.kdb
-rw-r--r--   1 TRAUNER  SYS1          80 Jun  7 11:36 apple2.rdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:11 apple2.sth
-rw-r--r--   1 TRAUNER  SYS1         765 Jun  7 11:00 itsoca.arm
-rw-r--r--   1 TRAUNER  SYS1       70080 Jun  7 11:00 itsoca.kdb
-rw-------   1 TRAUNER  SYS1         129 Jun  7 11:00 itsoca.sth
TRAUNER:/web/apple/sec: >
```

- **Modify httpd.conf to enable the Web server for SSL mode**
  - KeyFile — Define name and place of the key database
  - SSLMode — turn SSL on or off [on]
  - SSLPort — define the SSL port number [443]
  - Normalmode — define if normalmode still allowed [on]

Roland Trauner trauner@us.ibm.com

Configure Web Server for SSL

- **Keyfile**

```
#        keyfile directive:
#
#        Specify the names of key files that are available.
#
#        Default:  <none>
#        Syntax:   keyfile  <filename.kdb>
#        (multiple instances allowed, but only last one will be used.)
# keyfile itsoca.kdb
keyfile /web/apple/sec/apple2.kdb
```


Roland Trauner trauner@us.ibm.com

Configure Web Server for SSL

SSL Mode and SSL Port

```
#        sslmode directive:
#
#        Turn on/off SSL security using the port specified by
#        the sslport directive
#
#        Default:  on
#        Syntax:  sslmode <on | off>
sslmode    on

#        sslport directive:
#
#        Specify the port that should be used for SSL transactions.
#
#        Default:  443
#        Syntax:   sslport <port number>
sslport    443
```


Roland Trauner trauner@us.ibm.com

Configure Web Server for SSL

Normal Mode

```
#        normalmode directive:
#
#        Turn on/off non-secure transactions over the port you
#        specify with the port directive.
#
#        Default:  on
#        Syntax:   normalmode:  <on | off>
normalmode  on
```


Roland Trauner trauner@us.ibm.com

Configure Web Server for SSL

- **Restart the Server**
- **Check the Server**
  - https://www.the-apple.com

Roland Trauner trauner@us.ibm.com

Troubleshooting

- **If everything is ok, the vv trace log should display the following messages:**
  - Loaded security US
  - SSL security is "on"
  - sslport..... 8101
  - normalmode is "on"
  - keyfile..... /web/apple/sec/apple2.kdb
  - secinit.c: Setting up security, state=0
  - secinit.c: retcode from skit_initialize=0
  - secinit.c: calling webdb_recoverstash
  - Daemon..... Parsed SSL as port 8101, inet 0.0.0.0
  - IP..... Opened socket number 10
  - Daemon..... Master socket(), bind() and listen() all OK
  - Daemon..... SSL socket(), bind() and listen() all OK

Roland Trauner trauner@us.ibm.com

Troubleshooting

- **Export security --- everything ok**
  - Failed to load DLL module wwwus.so: EDC5205S DLL module not found. (errnojr=053b006c)
  - Loaded security EXPORT
  - SSL security is "on"
  - sslport..... 7047
  - normalmode is "on"
  - keyfile..... /web/zorro/sec/r1.kdb
  - secinit.c: Setting up security, state=0
  - secinit.c: retcode from skit_initialize=0
  - secinit.c: calling webdb_recoverstash
  - Daemon..... Parsed SSL as port 7047, inet 0.0.0.0
  - IP..... Opened socket number 10
  - Daemon..... Master socket(), bind() and listen() all OK
  - Daemon..... SSL socket(), bind() and listen() all OK

Roland Trauner trauner@us.ibm.com

Troubleshooting

- **Problem:**
  - Loaded security US
  - SSL security is "on"
  - sslport..... 7047
  - normalmode is "on"
  - keyfile..... /web/zorro/sec/r1.kdb
  - ErrorLog.... 26/Jan/1999:13:18:32 +0500 SSL support initialization failed, server will run only in non-secure mode without listening on ssl port
  - IP..... sslmode = off;SSL Port not opened

Roland Trauner trauner@us.ibm.com

Troubleshooting

- **Possible Reasons:**
  - Key database not at expected place
  - Stash file not in the same directory as key database
  - Stash file not the same name as key database
  - Database password expired

Roland Trauner trauner@us.ibm.com