z/OS® CS - SHARE in Anaheim, CA - February/March 2005

Session 3925 - Networking - TCP/IP

Secure Communications with FTP and TN3270 on z/OS CS: Implementation Scenarios

Thursday 3-Mar-2005 - 11:00 AM

Enterprise Networking Solutions, Raleigh
Alfred B Christensen - alfredch@us.ibm.com

Secure Business-to-Business Communication with FTP and TN3270 on z/OS CS - Implementation Scenarios

Session Number:	3925
Date:	Thursday, 3-Mar-2005
Time:	11:00 AM
Location:	Hilton Anaheim - Fourth floor - San Clemente
Speaker:	Alfred B Christensen, IBM
Chair:	Alfred B Christensen, IBM
Abstract:	In this part session, the speaker will provide practical examples on how to configure and use the security technologies discussed in session 3924 to implement secure communication for terminal access and file transfer functions. The scenarios in this session will show how to implement secure FTP and TN3270 using keyrings to hold keys and digital certificates in both a gskkyman key database and a RACF keyring. The scenarios will include server and client authentication for both TN3270 and FTP, and Express Logon for TN3270.

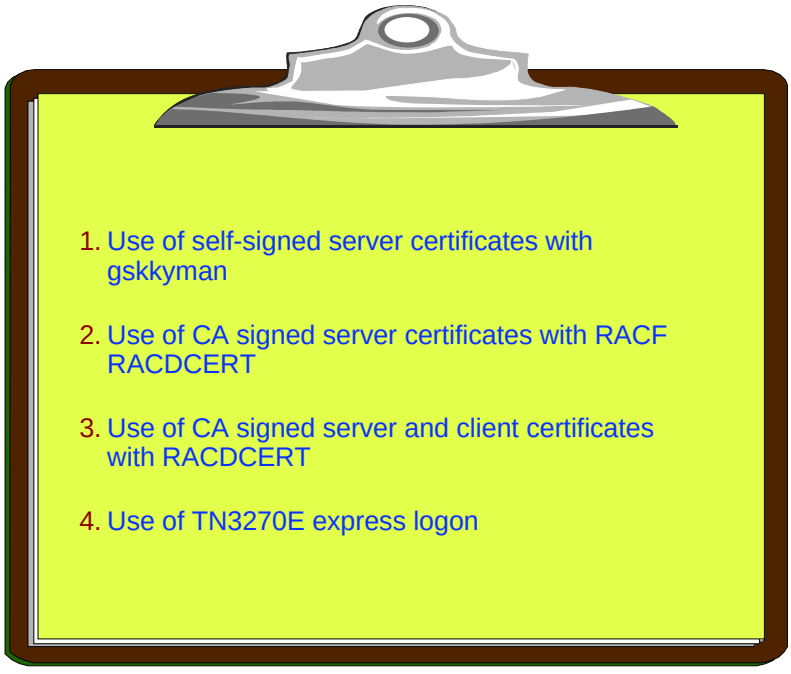
Trademarks and Notices

- The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both:

- AIX®
- AnyNet®
- AS/400®
- Candle®
- CICS®
- CICSPlex®
- CICS/ESA®
- DB2®
- DB2 Connect™
- DPI®
- DRDA®
- e business(logo)®
- ESCON®
- eServer™
- ECKD™
- FFST™
- GDDM®
- GDPS®
- HiperSockets™
- IBM®
- Infoprint®
- IMS™
- IP PrintWay™
- iSeries™
- Language Environment®
- MQSeries®
- MVS™
- MVS/ESA™
- NetView®
- OS/2®
- OS/390®
- Parallel Sysplex®
- PrintWay™
- PR/SM™
- pSeries®
- RACF®
- Redbooks™
- Redbooks (logo)™
- S/390®
- System/390®
- ThinkPad®
- Tivoli®
- Tivoli (logo)®
- VM/ESA®
- VSE/ESA™
- VTAM®
- WebSphere®
- xSeries®
- z/Architecture™
- z/OS®
- z/VM®
- zSeries®

- Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Intel, Intel Inside (logos), MMX and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Linux is a trademark of Linus Torvalds in the United States, other countries, or both.
- Other company, product, or service names may be trademarks or service marks of others.
- This information is for planning purposes only. The information herein is subject to change before the products described become generally available.
- All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represents goals and objectives only.

Topics



1. Use of self-signed server certificates with gskkyman
2. Use of CA signed server certificates with RACF RACDCERT
3. Use of CA signed server and client certificates with RACDCERT
4. Use of TN3270E express logon

This session will not discuss Kerberos-based security.

Self-signed Server Certificate using gskkyman

Objective and task outline

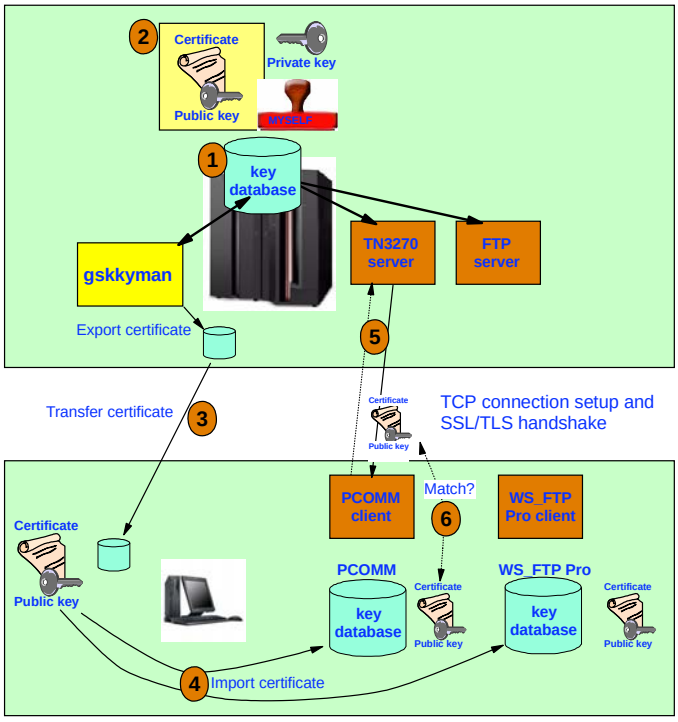
- Scenario objective:
  - Allow intranet clients to authenticate servers and to establish secure connections where data flows are encrypted.
  - Selfsigned certificates should in general only be used for testing purposes or in controlled environments
    - There is no 3rd party trust relationship present for selfsigned certificates

- Task outline:

1. Create server key database using gskkyman
2. Create self-signed server certificate
3. Update TN3270 and FTP server for server authentication only
4. Download self-signed server certificate to client workstation
5. Install self-signed server certificate as trusted root in PCOMM's keyring
6. Install self-signed server certificate as trusted root in WS_FTP Pro's keyring

Start simple !!

gskkyman selfsigned server certificate Scenario overview



1. Create key database
2. Create keypair and selfsigned server certificate
   - Please note that the private key never leaves z/OS - it is used by system SSL when cryptographic functions are done on behalf of the servers
3. Transfer certificate to workstation
4. Import certificate into workstation product key databases
   - PCOMM
   - WS_FTP Pro
   - Others ?? - such as IE
5. Establish TCP connection to server
6. During SSL/TLS handshake, the certificate is sent to the client and the client searches in its key database to see if it recognizes the certificate
   - Some products require that the certificate is in the key database before a new connection can use it
   - If the certificate does not match any in the key database, other products will prompt asking if you want to import the new certificate on-the-fly at this point in time

Create gskkyman key database

```
USER1:/u/user1: >mkdir SSLcase1
USER1:/u/user1: >cd SSLcase1
USER1:/u/user1/SSLcase1: >gskkyman

       Database Menu

   1 - Create new database
   2 - Open database
   3 - Change database password
   4 - Change database record length
   5 - Delete database

   0 - Exit program

Enter option number: 1
Enter key database name (press ENTER to return to menu): abckey.kdb
Enter database password (press ENTER to return to menu): xxxxxx
Re-enter database password: xxxxxx
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 2500):

Key database /u/user1/SSLcase1/abckey.kdb created.

Press ENTER to continue.
```

I usually create a separate directory in the HFS for each test case.

Assign a file name to your key database.

The key database is password protected since it will contain private keys. Do not loose the password - there is no way to access the key database if you loose the password.

gskkyman is documented in "System SSL Programming, Chapter 9", SC24-5901

gskkyman key database primary menu

```
       Key Management Menu

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 6
```

These are the functions that can be performed against a key database.

We will start by creating a selfsigned certificate.

Create selfsigned 2048-bit RSA key server certificate

```
       Certificate Type

   1 - CA certificate with 1024-bit RSA key
   2 - CA certificate with 2048-bit RSA key
   3 - CA certificate with 1024-bit DSA key
   4 - End user certificate with 1024-bit RSA key
   5 - End user certificate with 2048-bit RSA key
   6 - End user certificate with 1024-bit DSA key

Select certificate type (press ENTER to return to menu): 5
Enter label (press ENTER to return to menu): mvs098_self_signed
Enter subject name for certificate
  Common name (required): MVS098 self signed server certificate
  Organizational unit (optional): z/OS CS development
  Organization (required): IBM
  City/Locality (optional): Raleigh
  State/Province (optional): NC
  Country/Region (2 characters - required): US
Enter number of days certificate will be valid (default 365): 1000

Please wait .....

Certificate created.

Press ENTER to continue.
```

End user certificates are used for server and personal certificates. We are creating a server certificate in this case and choose a 2048 bit RSA key certificate.

The label name is the name in the key database by which we will refer to this certificate.

Subject name is entered as required/needed.

The validity period of this certificate. The certificate will no longer be valid after this many days from today.

Manage keys and certificates in the gskkkyman database

```
       Key Management Menu

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 1

       Key and Certificate List

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - mvs098_self_signed

   0 - Return to selection menu

Enter label number (ENTER to return to selection menu, p for previous list): 1
```

Back to the key database main menu.

We want to do some more with the new certificate, so we choose to manage keys and certificate.

They key and certificates are listed and we choose the one we just created.

Set selfsigned certificate as the default key and certificate in the gskkyman database

```
       Key and Certificate Menu

       Label: mvs098_self_signed

   1 - Show certificate information
   2 - Show key information
   3 - Set key as default
   4 - Set certificate trust status
   5 - Copy certificate and key to another database
   6 - Export certificate to a file
   7 - Export certificate and key to a file
   8 - Delete certificate and key
   9 - Change label
  11 - Create a certificate renewal request

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 3

Default key set.

Press ENTER to continue.
```

In order for our server configurations to be able to only refer to the key database name to choose a certificate to work with, we need to indicate which key and certificate is the default in the key database.

Export selfsigned certificate to flat file ready for download to workstation

```
       Key and Certificate Menu

       Label: mvs098_self_signed

   1 - Show certificate information
   2 - Show key information
   3 - Set key as default
   4 - Set certificate trust status
   5 - Copy certificate and key to another database
   6 - Export certificate to a file
   7 - Export certificate and key to a file
   8 - Delete certificate and key
   9 - Change label
  11 - Create a certificate renewal request

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 6

       Export File Format

   1 - Binary ASN.1 DER
   2 - Base64 ASN.1 DER
   3 - Binary PKCS #7
   4 - Base64 PKCS #7

Select export format (press ENTER to return to menu): 2
Enter export file name (press ENTER to return to menu): mvs098_self_signed.crt

Certificate exported.
```

The server certificate (not the private key) has to be exported to a flat file for later download to our workstation.

I have had most luck with DER encoded certificates. Binary and Base64 seem to work both. In this example, we choose DER base64.

We assign a file name to the exported certificate (this is the file we wil later download).

Selfsigned certificate and key are ready

```
       Key and Certificate Menu

       Label: mvs098_self_signed

   1 - Show certificate information
   2 - Show key information
   3 - Set key as default
   4 - Set certificate trust status
   5 - Copy certificate and key to another database
   6 - Export certificate to a file
   7 - Export certificate and key to a file
   8 - Delete certificate and key
   9 - Change label
  11 - Create a certificate renewal request

   0 - Exit program

Enter option number (press ENTER to return to previous menu):

       Key and Certificate List

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - mvs098_self_signed

   0 - Return to selection menu

Enter label number (ENTER to return to selection menu, p for previous list): 0
```

We're done working with our key.

Here we just press ENTER to get back to the list of keys and certificates.

And here we choose 0 to get back to the key database main menu.

Store gskkyman key database password in a stash file

```
       Key Management Menu

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 10

Database password stored in /u/user1/SSLcase1/abckey.sth.

Press ENTER to continue.
```

Last thing we have to do is to store the database password in a stash file allowing servers to access the database for key and certificate use (not modification) without coding the password in the clear in their configuration files.

And we are done with gskkyman

```
       Key Management Menu

       Database: /u/user1/SSLcase1/abckey.kdb

   1 - Manage keys and certificates
   2 - Manage certificates
   3 - Manage certificate requests
   4 - Create new certificate request
   5 - Receive certificate issued for your request
   6 - Create a self-signed certificate
   7 - Import a certificate
   8 - Import a certificate and a private key
   9 - Show the default key
  10 - Store database password
  11 - Show database record length

   0 - Exit program

Enter option number (press ENTER to return to previous menu): 0
USER1:/u/user1/SSLcase1: >ls -al
total 184
drwxr-xr-x   2 ADSERVE  SYS1        8192 Feb  3 12:01 .
drwx------  40 ADSERVE  SYS1       32768 Feb  3 11:51 ..
-rw-------   1 ADSERVE  SYS1       37580 Feb  3 11:56 abckey.kdb
-rw-------   1 ADSERVE  SYS1          80 Feb  3 11:52 abckey.rdb
-rw-------   1 ADSERVE  SYS1         129 Feb  3 12:01 abckey.sth
-rw-r--r--   1 ADSERVE  SYS1        1411 Feb  3 11:57 mvs098_self_signed.crt
USER1:/u/user1/SSLcase1: >
```

The .kdb and .rdb files comprise the gskkyman key and certificate database itself.

The .sth file is the database password stash file.

The .crt file is the exported version of our selfsigned certificate - we need for download to our workstation.

Download certificate to work station

```
-----BEGIN CERTIFICATE-----
MIID5jCCAs6gAwIBAgIIQB/SWwAKSTkwDQYJKoZIhvcNAQEFBQAwgYgxCzAJBgNV
BAKTAlVTMQswCQYDVQQIEwJOQzEQMA4GA1UEBxMHUmFsZWlnaDEMMAoGA1UEChMD
SUJNMRwwGgYDVQQLExN6L09TIENTIGRldmVsb3BtZW50MS4wLAYDVQQDEyVNVlMw
OTggc2VsZiBzaWduZWQgc2VydmVyIGNlcnRpZmljYXRlMB4XDTA0MDIwMzE2NTQ1
MVoXDTA2MTAzMDE2NTQ1MVowgYgxCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJOQzEQ
MA4GA1UEBxMHUmFsZWlnaDEMMAoGA1UEChxDSUJNMRwwGgYDVQQLExN6L09TIENT
IGRldmVsb3BtZW50MS4wLAYDVQQDEyVNVlMwOTggc2VsZiBzaWduZWQgc2VydmVy
IGNlcnRpZmljYXRlMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAr7Lt
K2CJ5/MBvcAjXNV1lfxgg24yMor/Id1BNgT/XhM090ZKH71sH4fTUtwZHfWLrtPK
doAM+kYy7Hp7aAxOJZUWaDdRXti75PmTtFhxb7/d6kGuJ1mYe7hCTwz1VPkV+QdF
NwHSVtwoCN/xl21Aw9rtijM2HCQ71oPvto7j//sTX328w/ldgdvXdGa8zVbYNjN+
zY2m/ifgd9EHkS6J9wwTNiPt12hh2wWXKtx183ARasnrYEwqMLPuQCuido3aM4zs
sssjLmBAEmRQJzOmMc1D29IA2D2B0qZy0MZ/rTp34YCwj3UWvEqaPvDXLXNU76RL
E1CfOqa66SJNvR1pjwIDAQABo1IwUDAdBgNVHQ4EFgQUIDasu14nwbWF046foevd
E5FLGQAwHwYDVR0jBBgwFoAUIDaqu14nwbWF046foevdE5FLGQAwDgYDVR0PAQH/
BAQDAcTwMA0GCSqGSIb3DQEBBQUAA4IBAQAoPRuB33wknWFK53IBV5+l4mt0eZDx
1jEpGBkAcTvOhwSo9BElwqSlBW1OKCFbkjAXTN5gtesc6LAhrooF+k/qcDslgzi/
FATaKTJWmIOv6aKO6OG9Im0D16Q3ryExLzZTslVfKmFfdf+dMzn79eVHyROtTMLa
pIRac0UvcMBzvgpN6ZOSMTu0ISLDOvbPRjCL6dnU0fINNaC63n3ho2e3YqsEJypC
y/pGjlKqgiKk2vfX1JU5Tdemwseu/1kBlSqKt2VVBPw+0V5nzYhaqL3Vywalt+CL
BbLgzCXJE6MRwu9xCNI7iZC8grw3Bwdgy0M/NG1JK8HjmrI5O1fwud8G
-----END CERTIFICATE-----
```

The certificate was stored in base64 encoding (a text file), so we must download using an ASCII transfer.

You can use FTP or whatever file transfer method you prefer as long as you translate from EBCDIC to ASCII during the transfer.

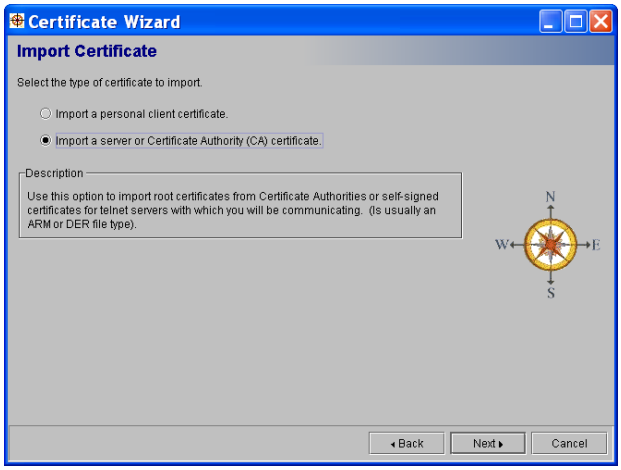
Import certificate into PCOMM - Step 1

1. Choose IBM Personal Communications in Start Programs
2. Choose Utilities
3. Choose Certificate Wizard

Select "Import a Certificate" - and press Next.

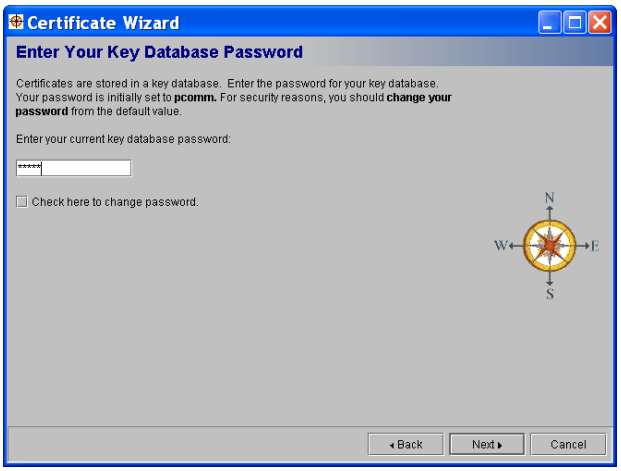
I am testing with IBM Personal Communications Workstation Program Version 5.6 for Windows

Import certificate into PCOMM - Step 2



Select "Import a server or Certificate Authority (CA) Certificate" - and press Next.

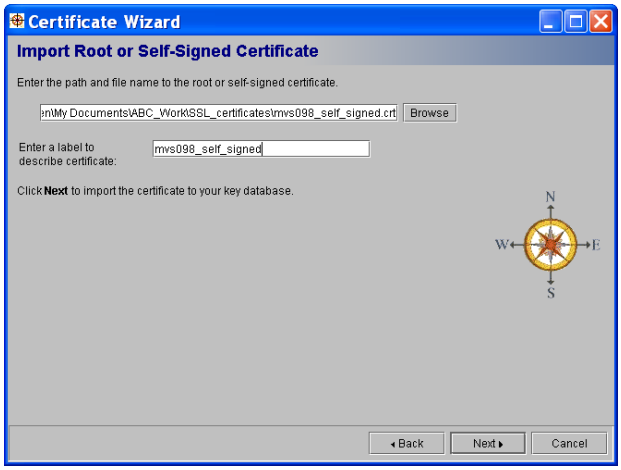
Import certificate into PCOMM - Step 3



Enter your PCOMM key database password - and press Next.

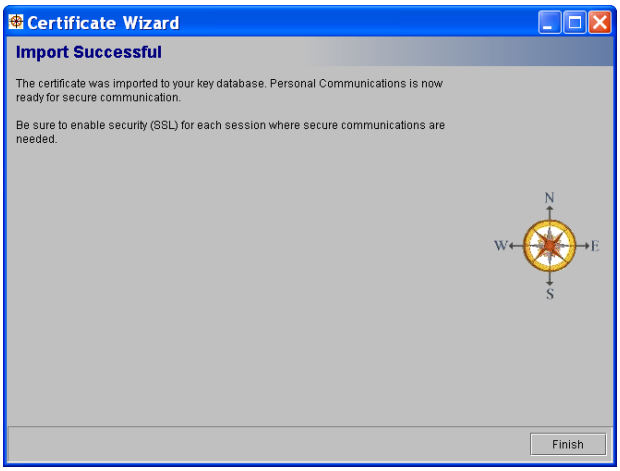
If you didn't change it, it is pcomm (you really should change it to something else!)

Import certificate into PCOMM - Step 4



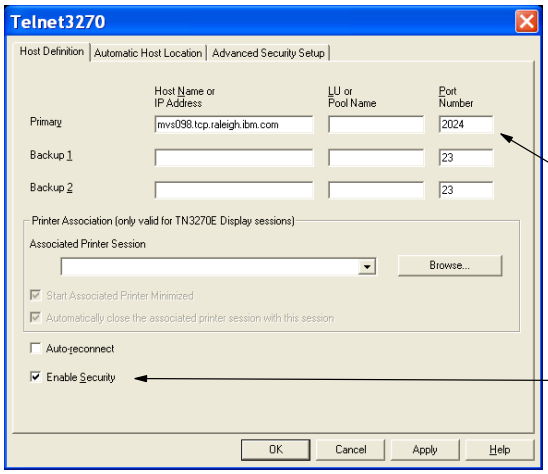
1. Point to the downloaded certificate file
2. Enter a local label to be used in PCOMM's key database so you know what this certificate is
3. and press Next

Import certificate into PCOMM - Step 5



This is just a confirmation that the certificate was imported and that PCOMM now is ready to accept a connection with a TN3270 server that presents this certificate.

Configure PCOMM session to be secure



1. Select the Communication pulldown
2. Choose Configure
3. Choose "link Parameters"
4. Point to your secure TN3270 server port number (in this example 2024)
5. Check the "enable Security" box

Updating TN3270 server definitions

We use a separate port number for this telnet server instance - port 2024.

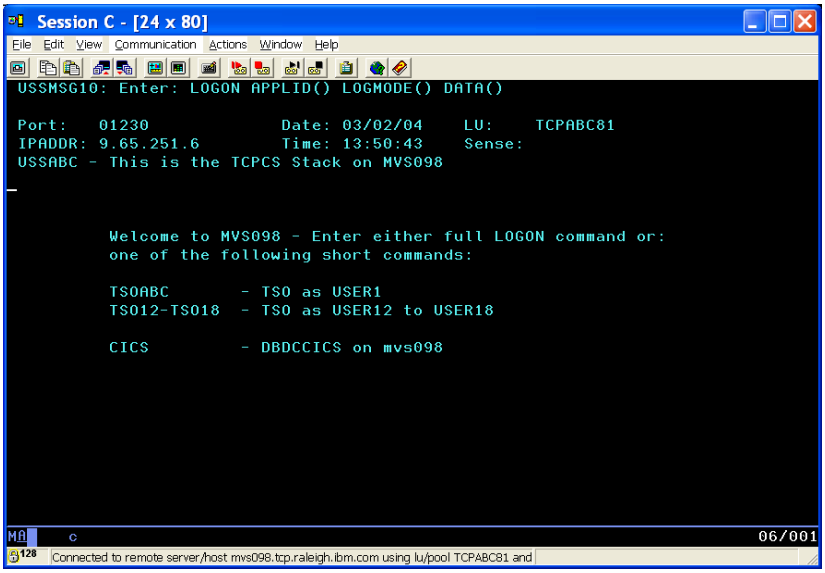
Conntype secure means that the connections must be secured and that we first try a traditional SSL handshake, if it times out we try a negotiable SSL handshake. If handshake fails, the connection is closed.

We do not require client authentication (client certificate is not needed).

We use the key database we just created with gskkyman. The self signed certificate is set as the default for that key database.

```
;
; Set TN3270(E) server secure port 2024 options
;
 TelnetParms
     SecurePort 2024                ; Port 2024
     Conntype secure                ; All connections must be secure
     Clientauth none                ; No client authentication
     Keyring HFS /u/user1/SSLcase1/abckey.kdb
     CodePage ISO8859-1 IBM-1047    ; Linemode ASCII, EBCDIC code pages
     Inactive 0                     ; Let connections stay around
     PrtInactive 0                  ; Let connections stay around
     TimeMark 600
     Debug detail console           ; Just for testing purposes
     ScanInterval 120
     SMFinit Type119                ; SMF119 records
     SMFterm Type119                ; SMF119 records
 EndTelnetParms
```

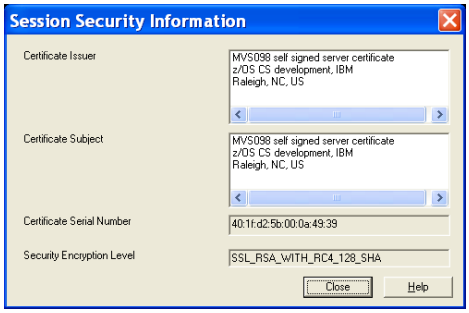
Start secure PCOMM session



The lock indicates that this is a secure connection.

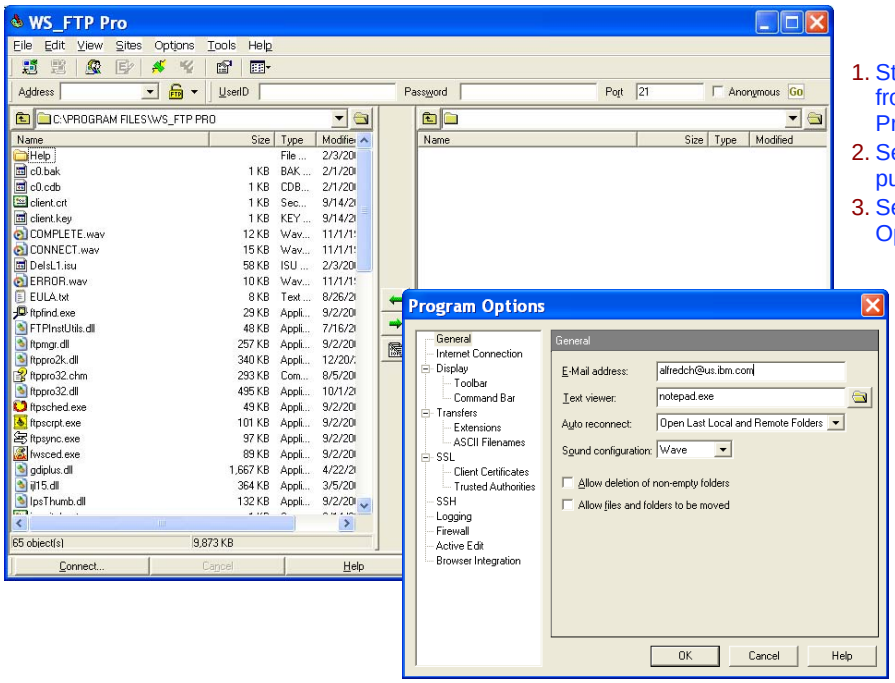
Verify which server certificate is in use by PCOMM

1. Select the Communication pulldown
2. Select Security
3. Select Server



This is the self signed server certificate we created in gskkyman many pages earlier...

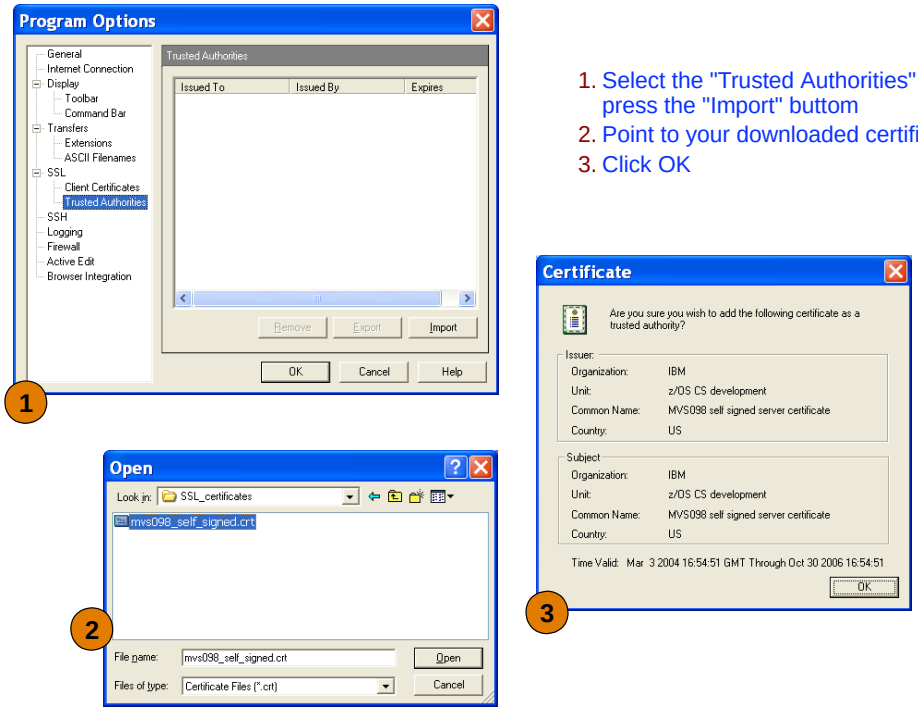
Import certificate into WS_FTP Pro client - Step 1



1. Start WS_FTP Pro from Start Programs
2. Select Options pulldown
3. Select Program Options

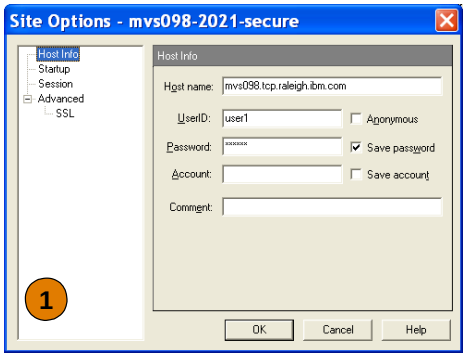
I am using the WS_FTP Pro client Version 8.02 2003.09.02 by Ipswitch Inc.

Import certificate into WS_FTP Pro client - Step 2

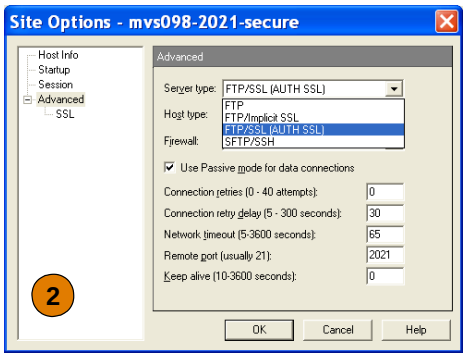


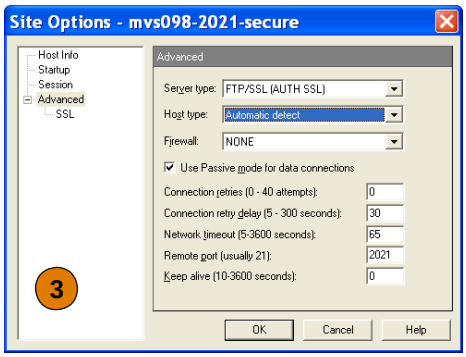
1. Select the "Trusted Authorities" tab and press the "Import" buttom
2. Point to your downloaded certificate file
3. Click OK

Configure WS_FTP Pro client for secure connection



1. In the normal connection definition panel, you select the Advanced tab
2. You then select server type as FTP/SSL (AUTH SSL)
3. In our case the secure FTP server listens on port 2021





Configure the FTP server for secure sessions

```
EXTENSIONS          AUTH_TLS            ; Enable TLS authentication
                                        ; Default is disabled.
SECURE_FTP          REQUIRED            ; Authentication indicator
                                        ; ALLOWED        (D)
                                        ; REQUIRED
SECURE_LOGIN        NO_CLIENT_AUTH      ; Authorization level indicator
                                        ; NO_CLIENT_AUTH (D)
                                        ; REQUIRED
                                        ; VERIFY_USER
SECURE_CTRLCONN     PRIVATE             ; Minimum level of security for
                                        ; the control connection
                                        ; CLEAR          (D)
                                        ; SAFE
                                        ; PRIVATE
SECURE_DATACONN     PRIVATE             ; Minimum level of security for
                                        ; the data connection
                                        ; NEVER
                                        ; CLEAR          (D)
                                        ; SAFE
                                        ; PRIVATE
;SECURE_PBSZ        16384               ; Kerberos maximum size of the
                                        ; encoded data blocks
                                        ; Default value is 16384
                                        ; Valid range is 512 through 32768
```

AUTH_TLS enables use of SSL/TLS.

This server instance requires use of SSL/TLS.

Do not require client certificates in this test scenario.

SECURE_CTRLCONN is really not used by SSL/TLS - the control connection will always be secured when SSL/TLS is in use. (Has meaning for Kerberos support)

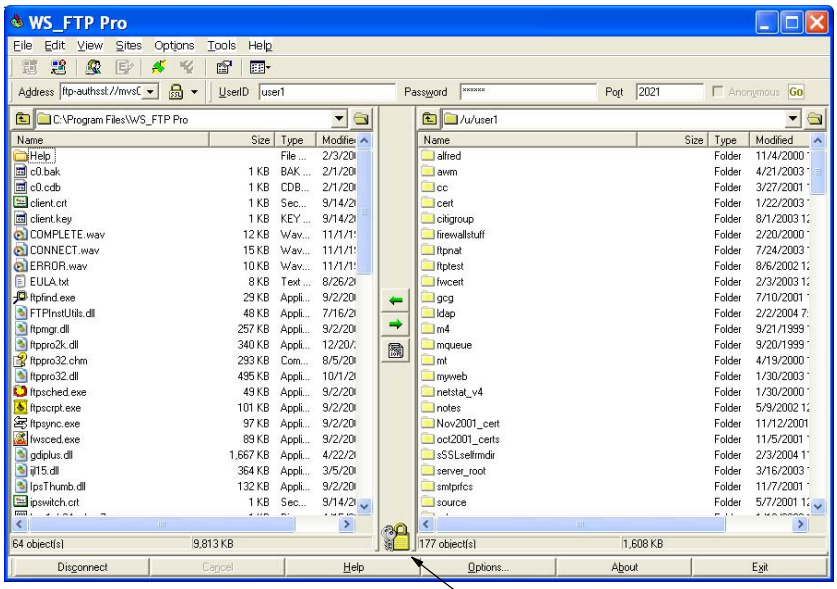
This server instance requires that the data connection be secured too.

```
; Name of a ciphersuite that can be passed to the partner during
; the TLS handshake. None, some, or all of the following may be
; specified. The number to the far right is the cipherspec id
; that corresponds to the ciphersuite's name.
;CIPHERSUITE        SSL_NULL_MD5        ; 01
;CIPHERSUITE        SSL_NULL_SHA        ; 02
 CIPHERSUITE        SSL_RC4_MD5_EX      ; 03
 CIPHERSUITE        SSL_RC4_MD5         ; 04
 CIPHERSUITE        SSL_RC4_SHA         ; 05
 CIPHERSUITE        SSL_RC2_MD5_EX      ; 06
 CIPHERSUITE        SSL_DES_SHA         ; 09
 CIPHERSUITE        SSL_3DES_SHA        ; 0A

KEYRING /u/user1/SSLcase1/abckey.kdb
                                       ; It can be the name of an hfs
                                       ; file (name starts with /) or
                                       ; a resource name in the security
                                       ; product (e.g., RACF)
TLSTIMEOUT          100                ; Maximum time limit between full
                                       ; TLS handshakes to protect data
                                       ; connections
                                       ; Default value is 100 seconds.
                                       ; Valid range is 0 through 86400
FTPLOGGING TRUE
```

Use our self signed certificate as server certificate.

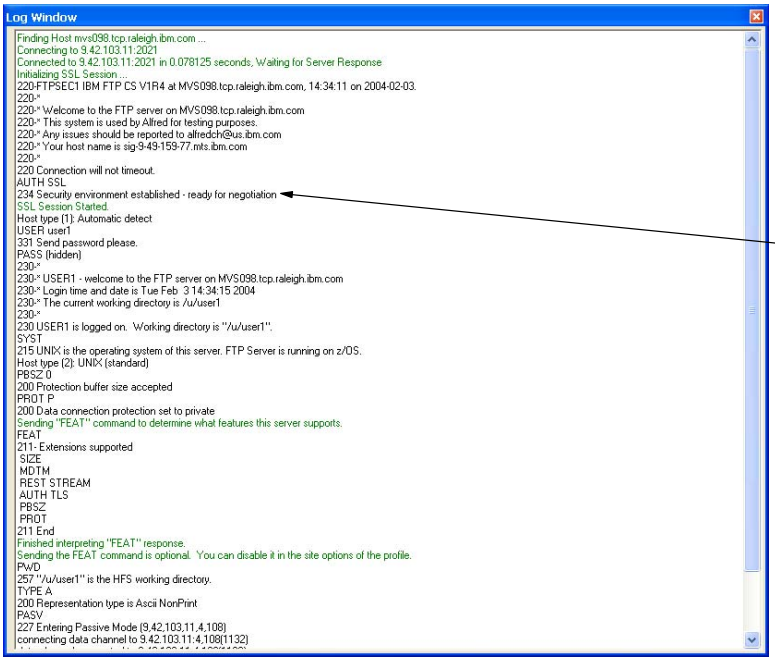
Btw: If it is a long time since you created your FTP.DATA from the sample FTPSDATA member in tcpip.SEZAINST - you should re-create your FTP.DATA based on that sample - it is now filled with comments.

Testing secure FTP connection



The lock indicates a secure connection.

Testing secure FTP connection



In the lower pane, you can see the AUTH SSL command being sent to the server and the positive 234 reply message.

FTP server activity log example

```
EZYFS50I ID=FTPSEC100001 CONN   starts Client IPaddr=::ffff:9.49.159.77 hostname=sig-9-49-159-77.mts.ibm.com
EZYFS54I ID=FTPSEC100001 SECURE OK     Mechanism=TLS-P
EZYFS56I ID=FTPSEC100001 ACCESS OK     USERID=USER1
EZYFS67I ID=FTPSEC100001 ALLOC  OK     Use HFS filename=/u/user1/testftp/filed2.txt
EZYFS77I ID=FTPSEC100001 DEALL  OK     Release HFS filename=/u/user1/testftp/filed2.txt
EZYFS82I ID=FTPSEC100001 TRANS  HFS filename=/u/user1/testftp/filed2.txt
EZYFS84I ID=FTPSEC100001 TRANS  Stru=F Mode=S Type=A Output=15 bytes
EZYFS80I ID=FTPSEC100001 TRANS  Reply=250 Transfer completed successfully.
EZYFS52I ID=FTPSEC100001 CONN   ends   Input=0 bytes Output=1911 bytes
```

The activity log will include a SECURE entry that indicates the session is secured using TLS.

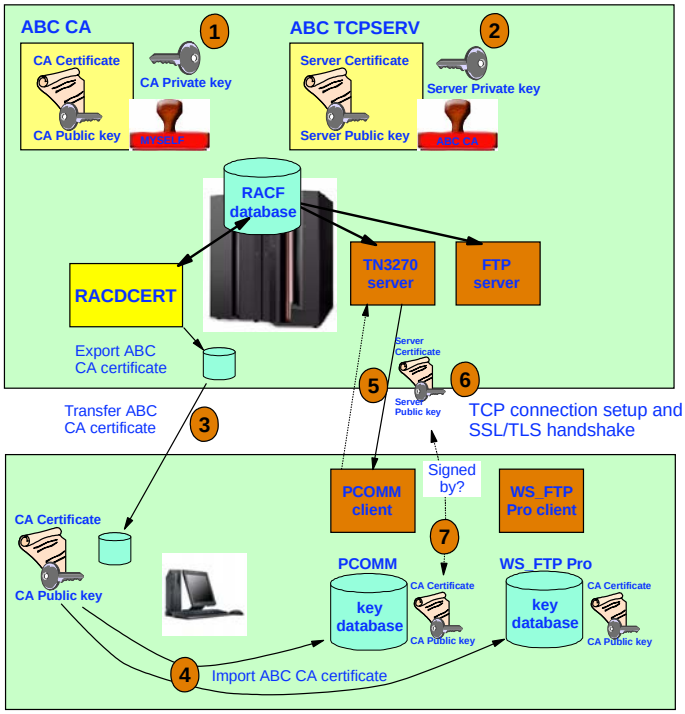
CA and RACF Key database scenario

CA Sample Scenario Based on use of RACF keyrings managed by RACDCERT - server certificate only

Task outline

- Create selfsigned CA certificate and keys in RACF ('ABC CA') as a CERTAUTH certificate
- Create a server certificate ('ABC TCPSERV') and keys signed with 'ABC CA'
- Create a keyring ('TCPRING') for the server started task user ID (TCPCS)
- Connect the CA certificate to that keyring as a CERTAUTH certificate
- Connect the server certificate to keyring as the default certificate in that ring
- Export the CA certificate to a base64-encoded text file
- Download the CA certificate to the client workstation
- Import the CA certificate into the PCOM key database as a certificate authority and do the same into the WS FTP pro key database
- Change the TN3270 and FTP server configuration to point to the SAF keyring
- Test secure connections from PCOMM and WS FTP Pro

RACDCERT CA and server certificates Scenario overview



1. Create the CA certificate (ABC CA) as a selfsigned certificate
2. Create the server certificate (ABC TCPSERV) signed by the CA certificate
3. Export and transfer the CA certificate to the workstation
4. Install the CA certificate as trusted root (aka signer) certificate into the workstation product key databases
5. Establish a connection to the server
6. During the SSL/TLS handshake the server certificate is sent to the client.
7. The client looks into its key database to see if it knows the certificate, or if it knows the signing certificate
   - In this case it will find the certificate that signed the server certificate and it can verify the authenticity of the server certificate

RACF hints and tips

- The server started task user ID (TCPCS) must have READ access to
  - IRR.DIGTCERT.LIST
  - IRR.DIGTCERT.LISTRING
- Individual users who execute the z/OS FTP client and transmit user certificates must also have READ access to the above two profiles.
- Diagnosing system SSL problems can be done by passing an environment variable to the server:
  - GSK_TRACE_FILE=/dir/rawtracefile
- The system SSL trace file can be formatted with
  - gsktrace /dir/rawtracefile > /dir/formattedtracefile
- All certificate-related tasks can be performed using the RACF command interface (RACDCERT) or using the RACF ISPF interface.
- With FTP APAR PQ80574, users who log on to the FTP server when the server uses a private key managed by ICSF, no longer need access to SAF resource CSFSERV

I refer to RACF throughout this section. Where similar functions are supported by other security products, such products can be used instead of RACF.

Create certificates and key rings in RACF

```
//ALFREDCI JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*
//IEFPROC  EXEC PGM=IKJEFT01,REGION=4M,DYNAMNBR=10
//SYSTSPRT DD SYSOUT=*                BATCH TSO SESSION LOG
//SYSTSIN  DD *
RACDCERT CERTAUTH GENCERT +
         SUBJECTSDN(CN('ABC CA') +
         OU('CS Z/OS CA') +
         O('IBM') +
         C('US')) +
         NOTBEFORE(DATE(2004-01-01)) +
         NOTAFTER(DATE(2010-12-31)) +
         WITHLABEL('ABC CA')
RACDCERT ID(TCPCS) GENCERT +
         SUBJECTSDN(CN('ABC TCPSERV') +
         OU('CS Z/OS TCPSERV') +
         O('IBM') +
         C('US')) +
         NOTBEFORE(DATE(2004-01-01)) +
         NOTAFTER(DATE(2010-12-31)) +
         WITHLABEL('ABC TCPSERV') +
         SIGNWITH(CERTAUTH LABEL('ABC CA'))
RACDCERT CERTAUTH EXPORT(LABEL('ABC CA')) DSN('USER1.ABCCA.B64')
RACDCERT ID(TCPCS) ADDRING(TCPRING)
RACDCERT ID(TCPCS) CONNECT(CERTAUTH LABEL('ABC CA') +
         RING(TCPRING)
RACDCERT ID(TCPCS) CONNECT(LABEL('ABC TCPSERV') +
         RING(TCPRING) DEFAULT)
RACDCERT ID(TCPCS) LISTRING(TCPRING)
/*
```

Create our selfsigned CA certificate by which all our other certificates will be signed.

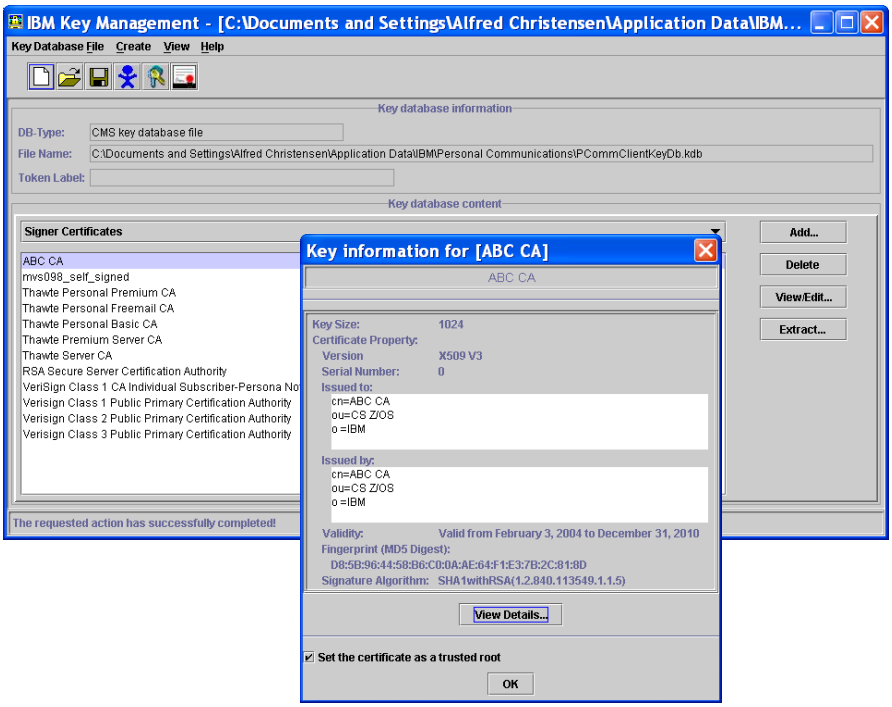
Create our server certificate and sign it with our CA certificate.

Export our CA certificate so we can download and install as trusted root in PCOMM's and WS_FTP Pro's key databases

Create our server keyring

Connect both our CA and our server certificate to that keyring and set the server certificate as the default key and certificate in that ring.

Import RACF CA (signer) certificate into PCOMM's key database



You can use the same certificate wizard we used for the selfsigned certificate, or you can choose to use the PCOMM key management component (both are under the PCOMM Utilities option in start programs).

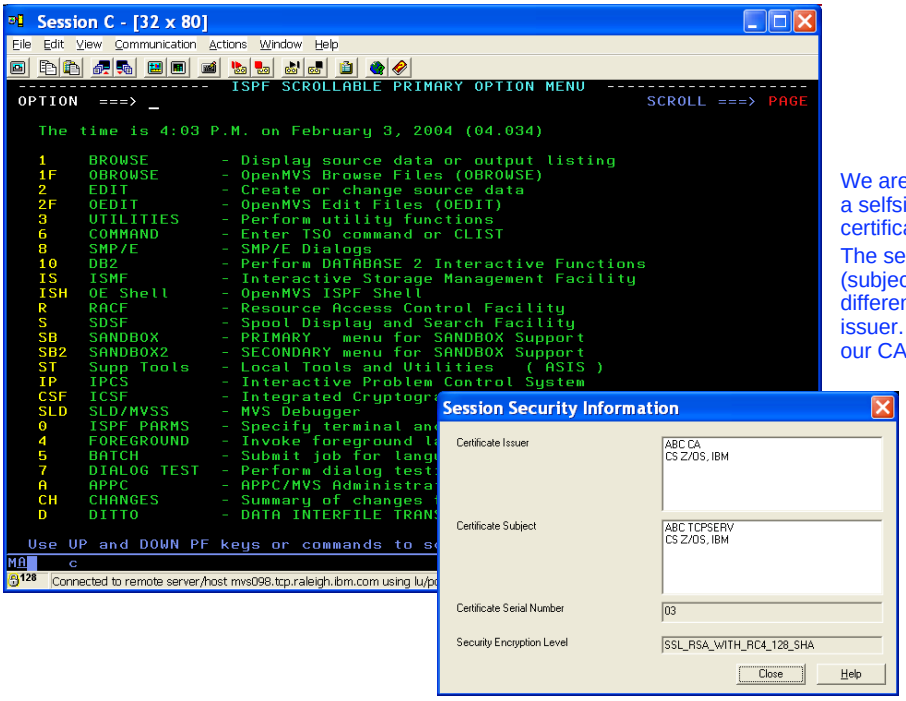
You open your PCOMM keydatabase (remember the password!), choose to work with signer certificates, push Add, select the downloaded CA certificate from RACF, and install that as a trusted root.

Change the TN3270 server to point to the RACF keyring

```
;
; Set TN3270(E) server secure port 2024 options
;
 TelnetParms
     SecurePort 2024               ; Port 2024
     Conntype secure               ; All connections must be secure
     Clientauth none               ; No client authentication
     Keyring SAF TCPRING           ; RACF key ring
     CodePage ISO8859-1 IBM-1047   ; Linemode ASCII, EBCDIC code pages
     Inactive 0                    ; Let connections stay around
     PrtInactive 0                 ; Let connections stay around
     TimeMark 600
     Debug detail console          ; Just for testing purposes
     ScanInterval 120
     SMFinit Type119               ; SMF119 records
     SMFterm Type119               ; SMF119 records
 EndTelnetParms
;
```

The only change to our TN3270 server definition is to point to a RACF (SAF) keyring instead of a gskkyman key database file name.

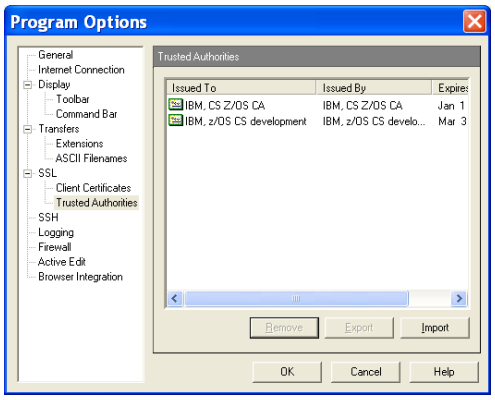
Test TN3270 server



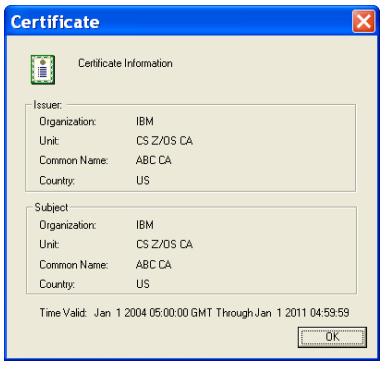
We are no longer using a selfsigned server certificate now.

The server certificate (subject) is now different from the issuer. The issuer is our CA certificate.

Add CA certificate to WS_FTP Pro's key database

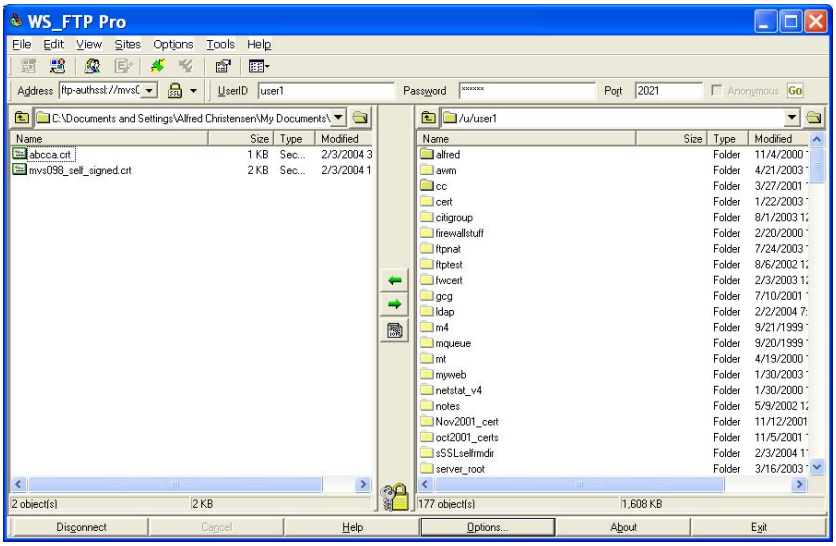


Same process as before: add the CA certificate to WS_FTP Pro's trusted authorities.



Modify the FTP server's configuration and test secure FTP transfers

```
KEYRING tcpring
                                ; It can be the name of an hfs
                                ; file (name starts with /) or
                                ; a resource name in the security
                                ; product (e.g., RACF)
```



The only change that is needed to our FTP server's FTP.DATA is to now point to a RACF keyring.

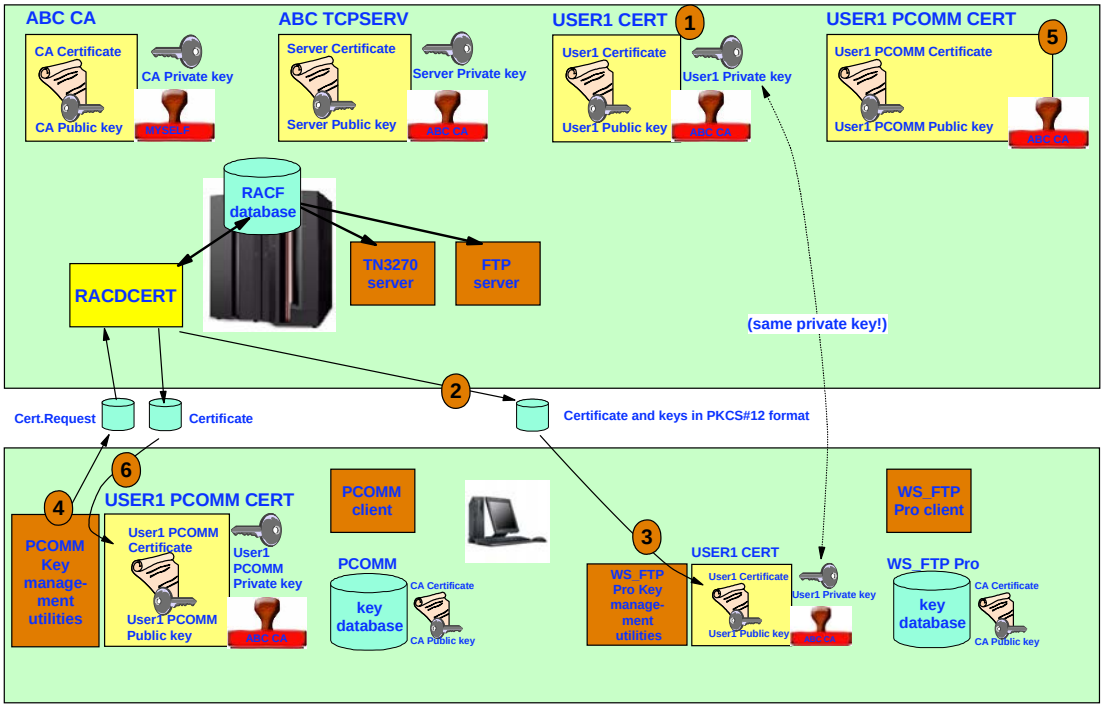
Please note you don't specify the keyword SAF (as you did for the TN3270 server). You just enter the RACF keyring name.

Extend CA scenario with client certificate

Task outline

- Create a personal certificate and keys for USER1 in RACF
- Create a keyring owned by USER1 - the 'USER1RING' keyring
- Connect the CA certificate ('ABC CA') to USER1RING
- Connect USER1's personal certificate to USER1RING as the default certificate
- I use two different methods for providing PCOMM and WS_FTP Pro with client certificates - both methods can be used with any of the two products (just illustrating both):
- PCOMM:
  - Use PCOMM's key management utilities to generate a new key pair (private and public) and a certificate request file
  - Upload the certificate request file z/OS
  - Use RACDCERT to generate a certificate based on the request, sign it with our CA certificate, and add it to USER1's keyring
  - Export the certificate, download to the workstation, and receive it into PCOMM's key database associated with the keypair that was generated earlier
- WS_FTP Pro
  - Export USER1's personal certificate from and private key from RACF into a password protected PKCS#12 data set in DER format
  - Download PKCS#12 data set in binary to the workstation
  - Import USER1's personal certificate and private key as a client certificate into WS FTP Pro's key databases
- Configure the FTP server and the TN3270 server to require client authentication
- Configure PCOMM and WS FTP Pro to send client certificates

Adding client certificates
Scenario overview



Create personal certificate and keys for USER1

```
//ALFREDCI JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*
//IEFPROC  EXEC PGM=IKJEFT01,REGION=4M,DYNAMNBR=10
//SYSTSPRT DD SYSOUT=*                BATCH TSO SESSION LOG
//SYSTSIN  DD *
RACDCERT ID(USER1) GENCERT +
         SUBJECTSDN(CN('USER1 CERT') +
         OU('CS Z/OS') +
         O('IBM') +
         C('US')) +
         NOTBEFORE(DATE(2004-01-01)) +
         NOTAFTER(DATE(2010-12-31)) +
         WITHLABEL('USER1 CERT') +
         SIGNWITH(CERTAUTH LABEL('ABC CA'))
RACDCERT ID(USER1) ADDRING(USER1RING)
RACDCERT ID(USER1) CONNECT(CERTAUTH LABEL('ABC CA') +
         RING(USER1RING)
RACDCERT ID(USER1) CONNECT(LABEL('USER1 CERT') +
         RING(USER1RING) DEFAULT)
RACDCERT ID(USER1) LISTRING(USER1RING)
/*
```

Create personal certificate and keys for user USER1

Create keyring for USER1

Connect both our CA certificate and USER1's personal certificate to the keyring and set USER1's personal certificate as the default.

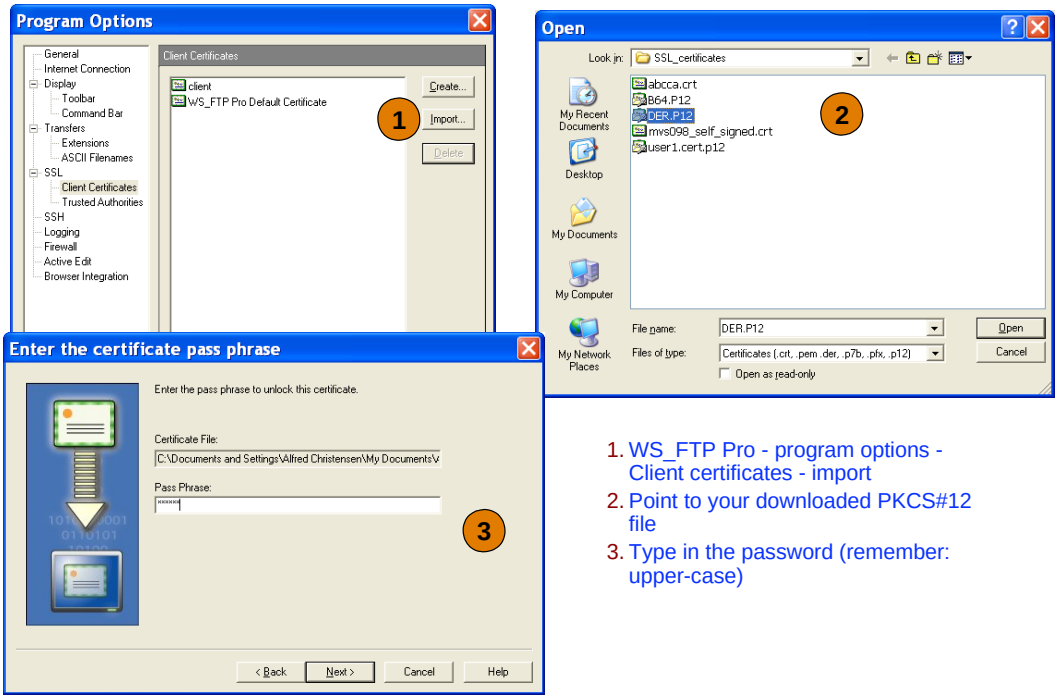
```
Digital ring information for user USER1:

  Ring:
       >USER1RING<
  Certificate Label Name              Cert Owner     USAGE      DEFAULT
  --------------------------------    ------------   --------   -------
  ABC CA                              CERTAUTH       CERTAUTH     NO
  USER1 CERT                          ID(USER1)      PERSONAL     YES
```


Export personal certificate and keys to password-protected data set (PKCS#12 format)

```
//ALFREDCI JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*
//IEFPROC  EXEC PGM=IKJEFT01,REGION=4M,DYNAMNBR=10
//SYSTSPRT DD SYSOUT=*                BATCH TSO SESSION LOG
//SYSTSIN  DD *
RACDCERT EXPORT (LABEL('USER1 CERT')) +
         DSN('USER1.CERT.DER.P12') +
         FORMAT(PKCS12DER) +
         PASSWORD('xxxxxx')
/*
```

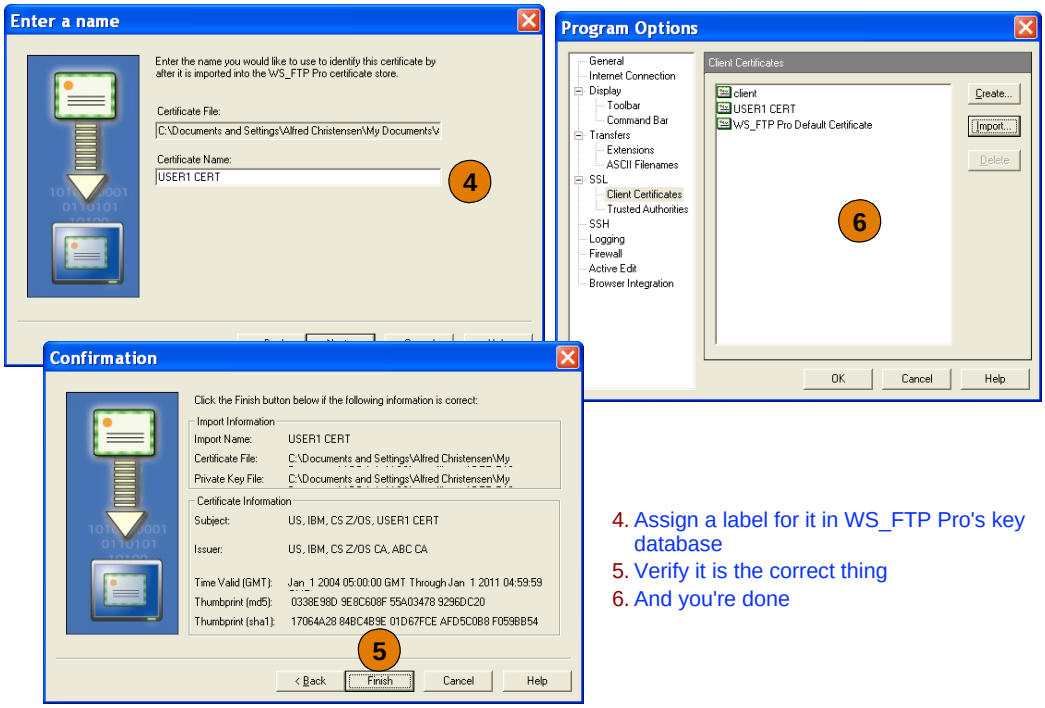
- A PKCS#12 file does not just hold the certificate - it also holds the matching private key. That's why the file itself needs password protection.
- A DER encoded file is a binary file and must be downloaded to the workstation in binary form.
- The certificate and keys could also be exported into a Base64 encoded format (that must be downloaded as a text file), but I had problems getting WS_FTP Pro to import such a file. The DER encoded file worked fine with both PCOM and WS_FTP Pro.
- The password is case sensitive - make sure you enter it in upper-case when you are prompted for the password at the workstation.

Import personal certificate and key into WS_FTP Pro



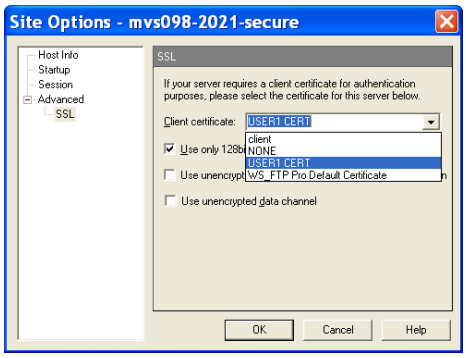
1. WS_FTP Pro - program options - Client certificates - import
2. Point to your downloaded PKCS#12 file
3. Type in the password (remember: upper-case)

Import personal certificate and key into WS_FTP Pro



4. Assign a label for it in WS_FTP Pro's key database
5. Verify it is the correct thing
6. And you're done

Configure WS FTP Pro for client authentication



1. For the FTP server connection in question, select the SSL tab and select the client certificate you want to use (the one we installed earlier)

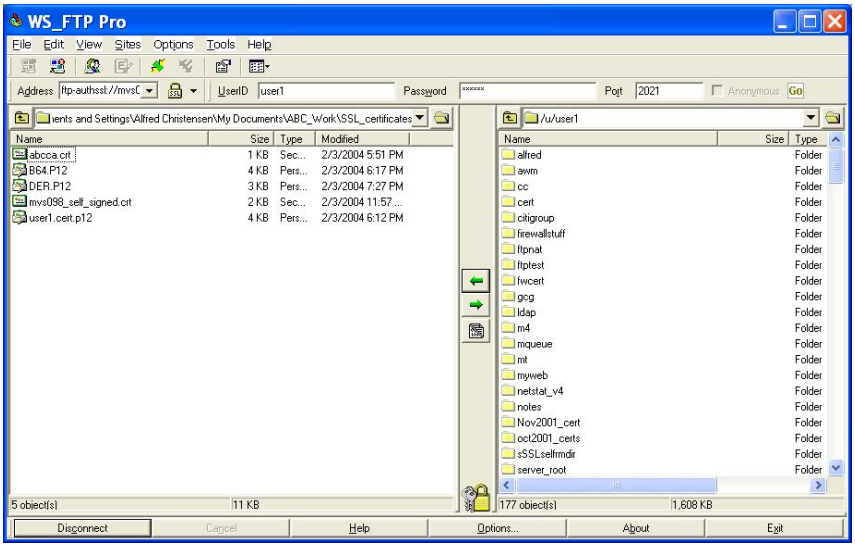
FTP server configuration for client authentication

```
EXTENSIONS          AUTH_TLS          ; Enable TLS authentication
                                      ; Default is disabled.
SECURE_FTP          REQUIRED          ; Authentication indicator
                                      ; ALLOWED        (D)
                                      ; REQUIRED
SECURE_LOGIN        VERIFY_USER       ; Authorization level indicator
                                      ; NO_CLIENT_AUTH (D)
                                      ; REQUIRED
                                      ; VERIFY_USER
SECURE_CTRLCONN     PRIVATE           ; Minimum level of security for
                                      ; the control connection
                                      ; CLEAR          (D)
                                      ; SAFE
                                      ; PRIVATE
SECURE_DATACONN     PRIVATE           ; Minimum level of security for
                                      ; the data connection
                                      ; NEVER
                                      ; CLEAR          (D)
                                      ; SAFE
                                      ; PRIVATE
SECURE_PBSZ         16384             ; Kerberos maximum size of the
                                      ; encoded data blocks
                                      ; Default value is 16384
                                      ; Valid range is 512 through 32768
```

```
; Name of a ciphersuite that can be passed to the partner during
; the TLS handshake. None, some, or all of the following may be
; specified. The number to the far right is the cipherspec id
; that corresponds to the ciphersuite's name.
;CIPHERSUITE        SSL_NULL_MD5       ; 01
;CIPHERSUITE        SSL_NULL_SHA       ; 02
 CIPHERSUITE        SSL_RC4_MD5_EX     ; 03
 CIPHERSUITE        SSL_RC4_MD5        ; 04
 CIPHERSUITE        SSL_RC4_SHA        ; 05
 CIPHERSUITE        SSL_RC2_MD5_EX     ; 06
 CIPHERSUITE        SSL_DES_SHA        ; 09
 CIPHERSUITE        SSL_3DES_SHA       ; 0A
 KEYRING TCPRING
                                       ; It can be the name of an hfs
                                       ; file (name starts with /) or
                                       ; a resource name in the security
                                       ; product (e.g., RACF)
 TLSTIMEOUT         100                ; Maximum time limit between full
                                       ; TLS handshakes to protect data
                                       ; connections
```

1. The only thing we need to change in the FTP server's FTP.DATA is the SECURE_LOGIN option to REQUIRED or VERIFY_USER

Testing WS FTP Pro with client authentication



WS FTP Pro doesn't have any displays that show which certificates are in use. The only way I've found to verify that client certiificate was requested by the server is by enabling FTP server debug mode and searching for the following debug lines in SyslogD:

```
FR1799 ftpAuth: entered
FR1842 ftpAuth: environment_open()
FR1932 ftpAuth: connect as a server requesting client certificate
FR1975 ftpAuth: environment_init()
FR1984 ftpAuth: environment initialization complete
```


Configure z/OS FTP client FTP.DATA for client authentication

```
SECURE_MECHANISM  TLS             ; Name of the security mechanism
                                  ; that the client uses when it
                                  ; sends an AUTH command to the
                                  ; server.
                                  ; GSSAPI = Kerberos support
                                  ; TLS    = TLS
SECURE_FTP        REQUIRED        ; Authentication indicator
                                  ; ALLOWED       (D)
                                  ; REQUIRED
SECURE_CTRLCONN   PRIVATE         ; Minimum level of security for
                                  ; the control connection
                                  ; CLEAR          (D)
                                  ; SAFE
                                  ; PRIVATE
SECURE_DATACONN   PRIVATE         ; Minimum level of security for
                                  ; the data connection
                                  ; NEVER
                                  ; CLEAR          (D)
                                  ; SAFE
                                  ; PRIVATE
;SECURE_PBSZ      16384           ; Kerberos maximum size of the
                                  ; encoded data blocks
                                  ; Default value is 16384
                                  ; Valid range is 512 through 32768
```

```
;CIPHERSUITE      SSL_NULL_MD5     ; 01
;CIPHERSUITE      SSL_NULL_SHA     ; 02
 CIPHERSUITE      SSL_RC4_MD5_EX   ; 03
 CIPHERSUITE      SSL_RC4_MD5      ; 04
 CIPHERSUITE      SSL_RC4_SHA      ; 05
 CIPHERSUITE      SSL_RC2_MD5_EX   ; 06
 CIPHERSUITE      SSL_DES_SHA      ; 09
 CIPHERSUITE      SSL_3DES_SHA     ; 0A
 KEYRING USER1RING                 ; Name of the keyring for TLS
                                   ; It can be the name of an hfs
                                   ; file (name starts with /) or
                                   ; a resource name in the security
                                   ; product (e.g., RACF)
 TLSTIMEOUT       100              ; Maximum time limit between full
                                   ; TLS handshakes to protect data
                                   ; connections
                                   ; Default value is 100 seconds.
                                   ; Valid range is 0 through 86400
```

- The client FTP user's FTP.DATA must be configured to point to that user's keyring - the keyring in which that user's personal certificate is defined as the default certificate.
- Also in that keyring is the certificate issuer's root certificate:

```
Ring:
     >USER1RING<
Certificate Label Name            Cert Owner     USAGE      DEFAULT
--------------------------------  ------------   --------   -------
ABC CA                            CERTAUTH       CERTAUTH     NO
USER1 CERT                        ID(USER1)      PERSONAL     YES
```

Sample batch FTP client job for a secure session

```
//ALFREDA JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*            USER=USER2,PASSWORD=TCPSUP
//*
//* test of client authentication
//*
//FTP   EXEC PGM=FTP,PARM='-a TLS'
//SYSTCPD   DD DSN=USER1.TCPCS.TCPPARMS(TCPDATA),DISP=SHR
//SYSFTPD   DD DSN=USER1.TCPCS.TCPPARMS(FTPUSER1),DISP=SHR
//SYSPRINT  DD SYSOUT=*
//INPUT     DD *
;
; Test of client authentication
;
mvs098.tcp.raleigh.ibm.com 2021 (exit
user1 xxxxxx
cd 'user1.alfred.cntl'
dir
quit
//OUTPUT    DD SYSOUT=*
```

Start option '-a TLS' instructs the FTP client to try and establish a TLS connection with the FTP server.

By encoding the password on the same line as the userID, we prepare for a z/OS V1R5 enhancement where the server may be configured to not require a password when client certificates are used. If the password was on its own line in the INPUT file, it would be interpreted by the FTP client as a command (unknown) and generate an error.

Batch FTP client log from secure session

```
EZA1736I FTP -a TLS
EZA1450I IBM FTP CS V1R4
EZA1466I FTP: using TCPCS
EZA1456I Connect to ?
EZA1736I mvs098.tcp.raleigh.ibm.com 2021 (exit
EZA1554I Connecting to: mvs098.tcp.raleigh.ibm.com 9.42.103.11 port: 2021.
220-FTPSEC1 IBM FTP CS V1R4 at MVS098.tcp.raleigh.ibm.com, 20:17:45 on 2004-02-03.
220-*
220-* Welcome to the FTP server on MVS098.tcp.raleigh.ibm.com
220-* This system is used by Alfred for testing purposes.
220-* Any issues should be reported to alfredch@us.ibm.com
220-* Your host name is mvs098cs6.tcp.raleigh.ibm.com
220-*
220 Connection will not timeout.
EZA1701I >>> AUTH TLS
234 Security environment established - ready for negotiation
EZA2895I Authentication negotiation succeeded
EZA1701I >>> PBSZ 0
200 Protection buffer size accepted
EZA1701I >>> PROT P
200 Data connection protection set to private
EZA2906I Data connection protection is private
EZA1459I NAME (mvs098.tcp.raleigh.ibm.com:USER1):
EZA1701I >>> USER user1
331 Send password please.
EZA1701I >>> PASS
230-*
230-* USER1 - welcome to the FTP server on MVS098.tcp.raleigh.ibm.com
230-* Login time and date is Tue Feb  3 20:17:49 2004
230-* The current working directory is /u/user1
230-*
230 USER1 is logged on.  Working directory is "/u/user1".
EZA1460I Command:
```

Output log from batch FTP client shows the security setup commands and responses that flow between the client and the server

Login to z/OS V1R5 FTP server without a password

```
SECURE_LOGIN      VERIFY_USER       ; Authorization level indicator
                                    ; NO_CLIENT_AUTH (D)
                                    ; REQUIRED
                                    ; VERIFY_USER
SECURE_PASSWORD   OPTIONAL          ; W. clientuath is PW required?
                                    ; OPTIONAL
                                    ; REQUIRED (D)
```

Verify User is required.
New SECURE_PASSWORD option instructs if a password is required or not.
If user verification based on certificate doesn't succeed, user will be prompted for a password.

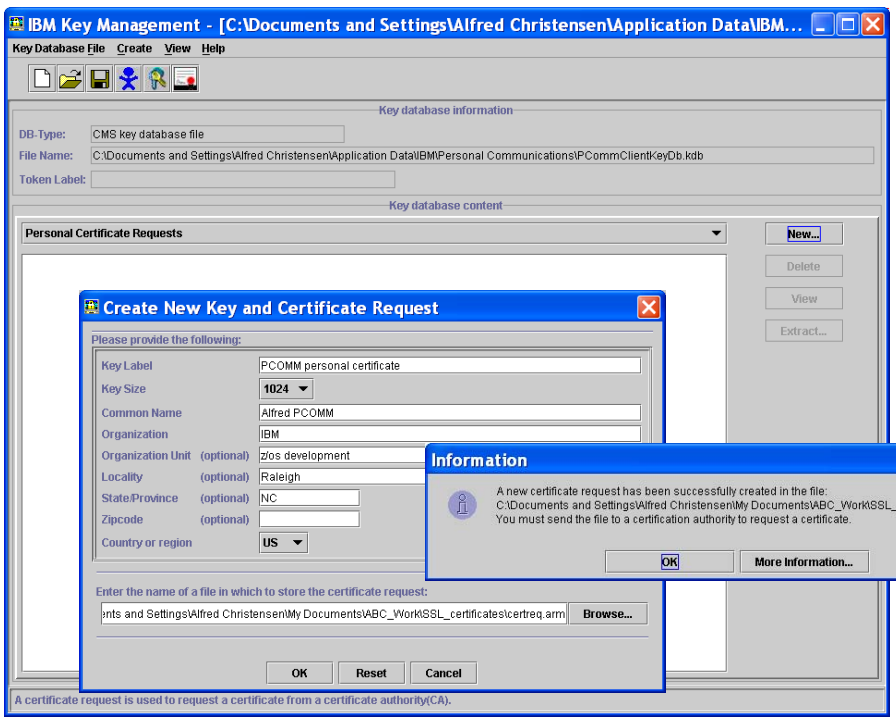
```
//ALFREDA JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*
//* test of client authentication
//*
//FTP    EXEC PGM=FTP,PARM='-a TLS'
//SYSTCPD   DD DSN=USER1.TCPCS.TCPPARMS(TCPDATA),DISP=SHR
//SYSFTPD   DD DSN=USER1.TCPCS.TCPPARMS(FTPUSER1),DISP=SHR
//SYSPRINT  DD SYSOUT=*
//INPUT     DD *
;
; Test of client authentication
;
mvs098.tcp.raleigh.ibm.com 2021 (exit
user1
cd 'user1.alfred.cntl'
dir
quit
//OUTPUT    DD SYSOUT=*
```

No password in batch FTP input stream - only the user ID.

No prompt for a password from the server.

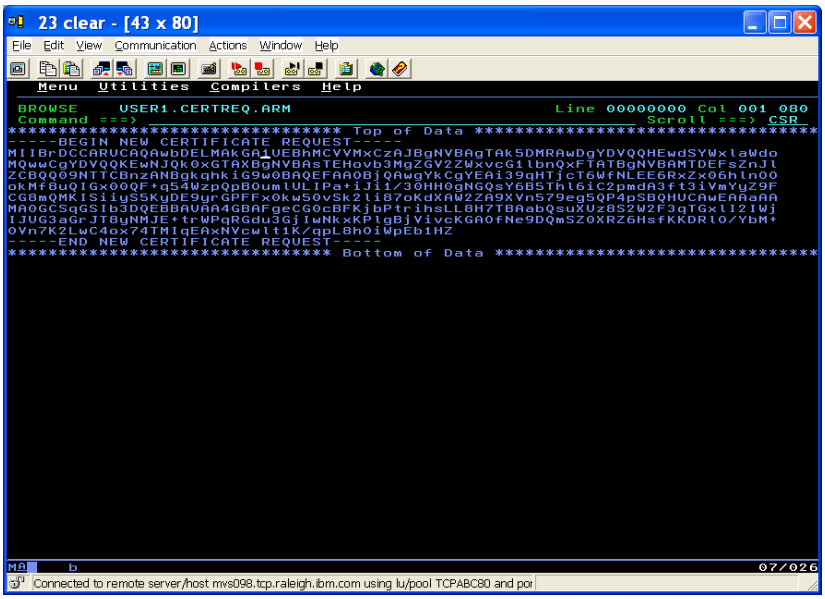
```
220-FTPSEC1 IBM FTP CS V1R5 at MVS098.tcp.raleigh.ibm.com, 11:35:39 on 2004-02-04.
220-*
220-* Welcome to the FTP server on MVS098.tcp.raleigh.ibm.com
220-* This system is used by Alfred for testing purposes.
220-* Any issues should be reported to alfredch@us.ibm.com
220-* Your host name is mvs098cs6.tcp.raleigh.ibm.com
220-*
220 Connection will not timeout.
EZA1701I >>> AUTH TLS
234 Security environment established - ready for negotiation
EZA2895I Authentication negotiation succeeded
EZA1701I >>> PBSZ 0
200 Protection buffer size accepted
EZA1701I >>> PROT P
200 Data connection protection set to private
EZA2906I Data connection protection is private
EZA1459I NAME (mvs098.tcp.raleigh.ibm.com:USER1):
EZA1701I >>> USER user1
230-*
230-* USER1 - welcome to the FTP server on MVS098.tcp.raleigh.ibm.com
230-* Login time and date is Wed Feb  4 11:35:41 2004
230-* The current working directory is /u/user1
230-*
230-User USER1 is an authorized user
230 USER1 is logged on.  Working directory is "/u/user1".
EZA1460I Command:
EZA1736I cd 'user1.alfred.cntl'
```


Generate local keys in PCOMM's key database and request for a certificate



- Private/public keys are generated and stored in PCOMM's key database (the private key never leaves this workstation).
- A certificate request is generated and stored as text file on the workstation (does not inlcude the private key)

Transfer certificate request as an ASCII transfer to an MVS data set



- Transfer the certreq.arm file to an MVS data set as an ASCII transfer.
- MVS data set attributes:
  - LRECL=84
  - RECFM=VB

Ask RACF to issue the new certificate and sign it with our CA certificate

```
//ALFREDCI JOB 1,ALFRED,CLASS=A,MSGCLASS=X,NOTIFY=USER1
//*
//IEFPROC  EXEC PGM=IKJEFT01,REGION=4M,DYNAMNBR=10
//SYSTSPRT DD SYSOUT=*                BATCH TSO SESSION LOG
//SYSTSIN  DD *
RACDCERT ID(USER1) GENCERT('USER1.CERTREQ.ARM') +
         NOTBEFORE(DATE(2004-01-01)) +
         NOTAFTER(DATE(2010-12-31)) +
         WITHLABEL('USER1 PCOMM CERT') +
         SIGNWITH(CERTAUTH LABEL('ABC CA'))
RACDCERT ID(USER1) CONNECT(LABEL('USER1 PCOMM CERT') +
         RING(USER1RING)
RACDCERT ID(USER1) LISTRING(USER1RING)
RACDCERT ID(USER1) EXPORT(LABEL('USER1 PCOMM CERT')) +
         DSN('USER1.PCOMM.B64')
/*
```

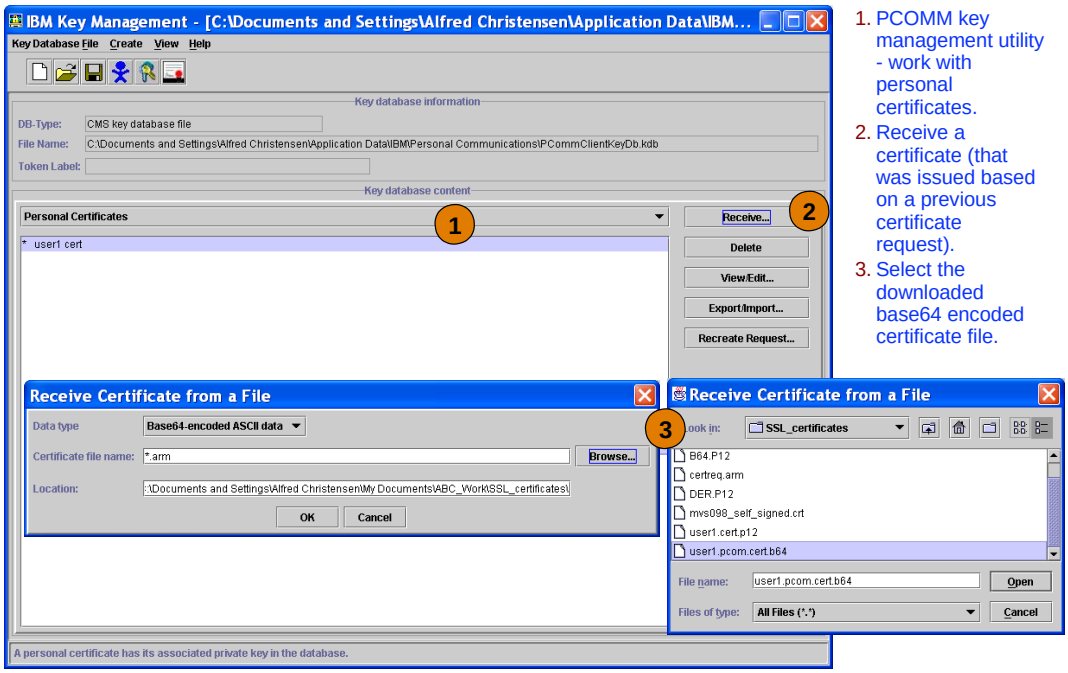
Generate a certificate from the certificate request we uploaded from PCOMM - and sign it with our CA certificate

Connect the certificate to USER1's keyring

Export the certificate into a base64 encoded file for download to PCOMM

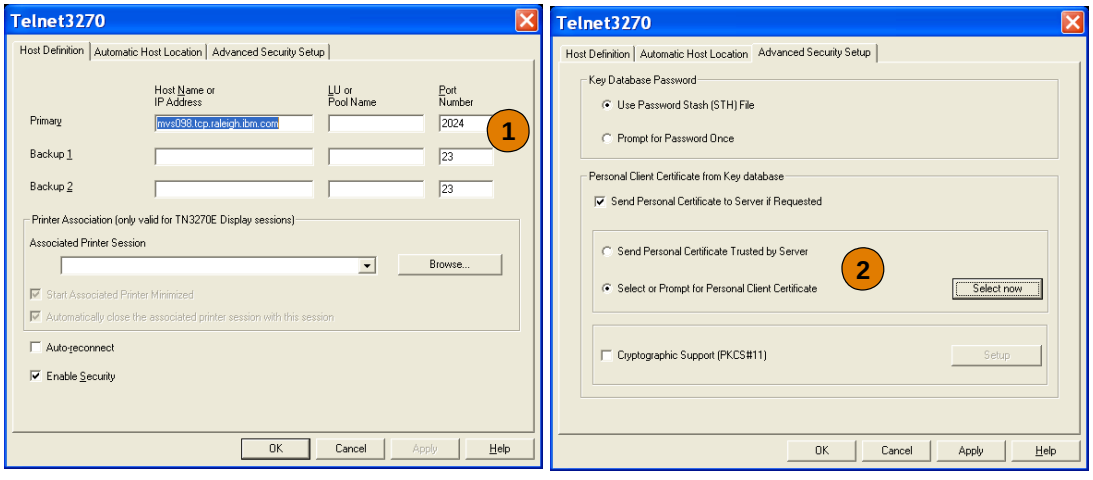
And download as an ASCII file to your workstation.

Receive newly issued certificate from RACF into PCOMM

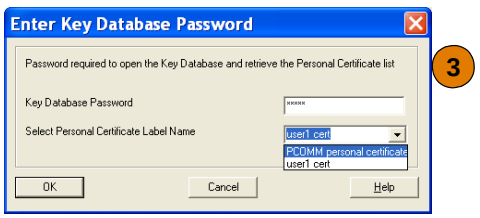


1. PCOMM key management utility - work with personal certificates.
2. Receive a certificate (that was issued based on a previous certificate request).
3. Select the downloaded base64 encoded certificate file.

Configure PCOMM session for client authentication



1. Choose TN3270 server port number, enable security
2. Select advanced security setup, enable send personal certificate, and select which one now
3. Enter your PCOMM key database password and select the personal certificate to use



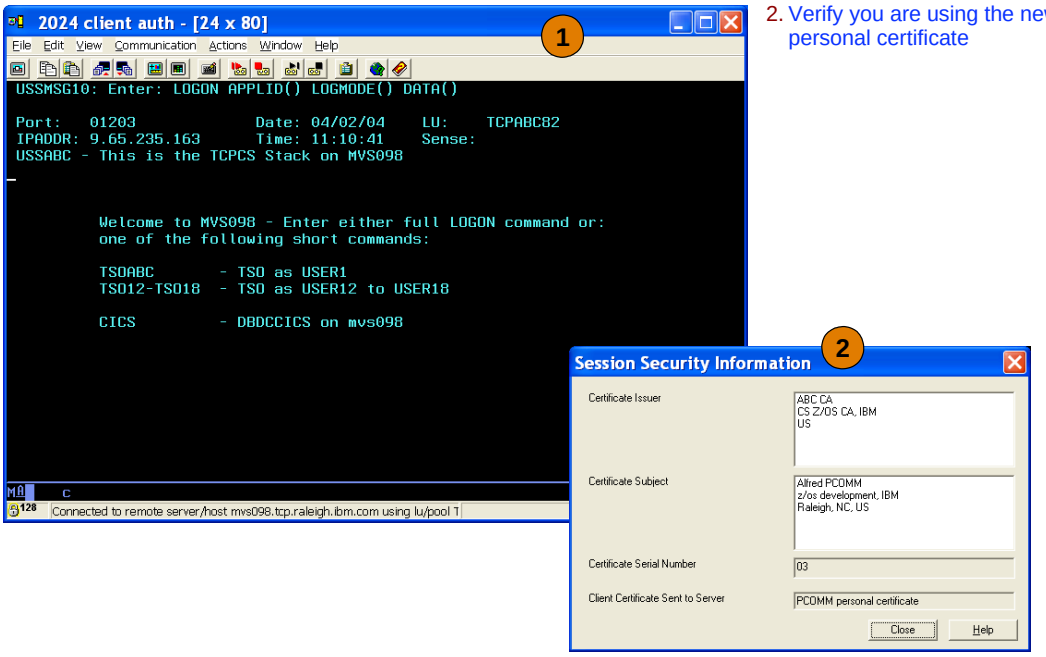
Configure the TN3270 server for client authentication

```
;
; Set TN3270(E) server secure port 2024 options
;
 TelnetParms
     SecurePort 2024               ; Port 2024
     Conntype secure               ; All connections must be secure
     Clientauth SAFCert            ; Ensure valid client cert
     Keyring SAF TCPRING           ; RACF key ring
     CodePage ISO8859-1 IBM-1047   ; Linemode ASCII, EBCDIC code pages
     Inactive 0                    ; Let connections stay around
     PrtInactive 0                 ; Let connections stay around
     TimeMark 600
     Debug detail console          ; Just for testing purposes
     ScanInterval 120
     SMFinit Type119               ; SMF119 records
     SMFterm Type119               ; SMF119 records
 EndTelnetParms
```

We need to change Clientauth to SSLCert or SAFCert

Configure PCOMM to use new personal certificate and test

1. Start your emulator
2. Verify you are using the new personal certificate



Express Logon

Express Logon Task Outline

1. Configure a PCOM session for client authentication
2. Record or use a pre-recorded logon macro
3. Define RACF passticket resource for TSO
4. Configure the telnet server for express logon

Configure for express logon

Adding express logon function to the telnet server is quite simple when client authentication already is in place:

1. Add the EXPRESSLOGON option to your TelnetParms block for your secure telnet port
2. Activate the RACF Passticket class:

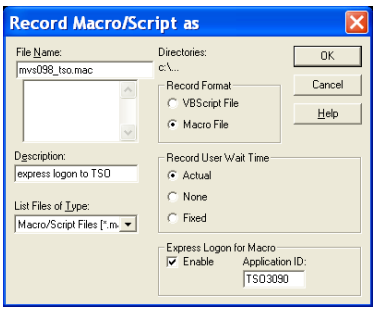
16 hexadecimal digits of your choice!

```
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA TSOxxxx SSIGNON(KEYMASKED(xxxxxxxxxxxxxxxx)) UACC(NONE)
SETROPTS RACLIST(PTKTDATA) REFRESH
```

Both PCOMM and HOD support express logon. To configure the PCOMM client you need to perform the following steps:

1. Create a new personal certificate or use your existing PCOMM personal certificate
2. Record a logon macro for the application in question - specifying an application ID that matches the PTKTDATA profile name you created in RACF earlier
3. Play the macro!

PCOMM logon macro recording

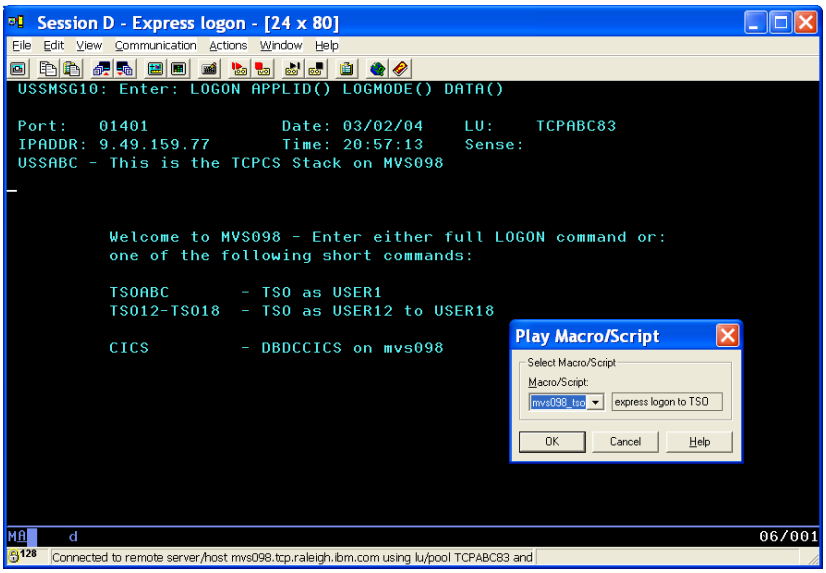


- To enable express logon, the administrator records a logon sequence to the application in question - in this example, to TSO
- We give the macro a name and select the Express Logon Feature.
- The userID and password entered during macro recording are not stored in the macro - pre-defined tokens are used in the macro. The macro can be shared/used by different users.

The application ID we enter here must match the passticket resource definition on z/OS - in this example: TSO3090

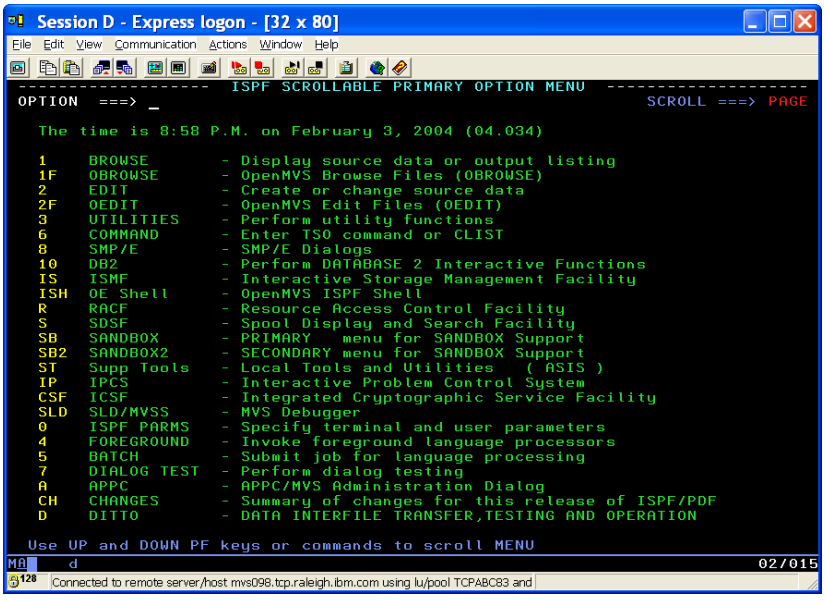
```
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA TSO3090 SSIGNON(KEYMASKED(xxxxxxxxxxxxxxxx)) UACC(NONE)
SETROPTS RACLIST(PTKTDATA) REFRESH
```

PCOMM playing the logon macro



- When the session has been established, the user selects the express logon macro to play.
- The macro does not include the user's actual user ID and password, but placeholders that are passed to the TN3270 server.

First user panel after logon macro played



For More Information....

URL	Content
http://www.ibm.com/servers/eserver/zseries	IBM eServer zSeries Mainframe Servers
http://www.ibm.com/servers/eserver/zseries/networking	Networking: IBM zSeries Servers
http://www.ibm.com/servers/eserver/zseries/networking/technology.html	IBM Enterprise Servers: Networking Technologies
http://www.ibm.com/software/network/commserver	Communications Server product overview
http://www.ibm.com/software/network/commserver/zos/	z/OS Communications Server
http://www.ibm.com/software/network/commserver/z_lin/	Communications Server for Linux on zSeries
http://www.ibm.com/software/network/ccl	Communication Controller for Linux on zSeries
http://www.ibm.com/software/network/commserver/library	Communications Server products - white papers, product documentation, etc.
http://www.redbooks.ibm.com	ITSO redbooks
http://www.ibm.com/software/network/commserver/support	Communications Server technical Support
http://www.ibm.com/support/techdocs/	Technical support documentation (techdocs, flashes, presentations, white papers, etc.)
http://www.rfc-editor.org/rfcsearch.html	Request For Comments (RFC)