GUIDE SHARE EUROPE
UK REGION

GSE UK Conference 2020
Virtually Unstoppable

What keyring? What certificates? All I know is TLS doesn't work!

Wai Choi
IBM RACF/PKI Development & Design
Poughkeepsie, NY

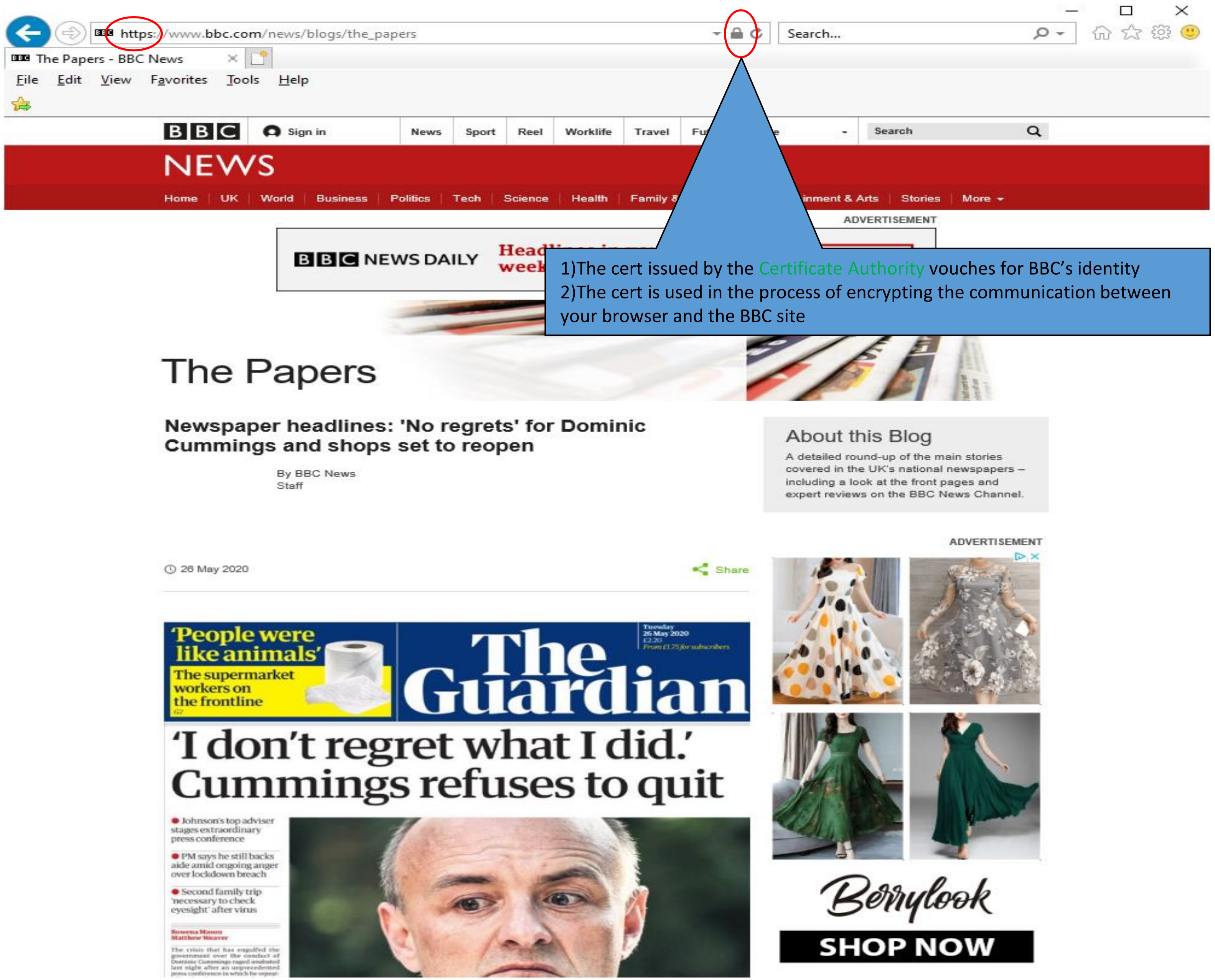
November 2020
Session 5BE

Agenda

- What is a digital certificate
- How to set up server and client keyrings for TLS
- Some tips on RACDCERT, keyring set up
- Steps to tackle a certificate related handshake problem in TLS

First encounter with digital certificate

- Do you know you come across it every day?
- Do you ever look at it?



1)The cert issued by the Certificate Authority vouches for BBC’s identity
2)The cert is used in the process of encrypting the communication between your browser and the BBC site

The Papers

Newspaper headlines: 'No regrets' for Dominic Cummings and shops set to reopen

By BBC News Staff

About this Blog

A detailed round-up of the main stories covered in the UK's national newspapers – including a look at the front pages and expert reviews on the BBC News Channel.

26 May 2020

Share

‘People were like animals’
The supermarket workers on the frontline

The Guardian

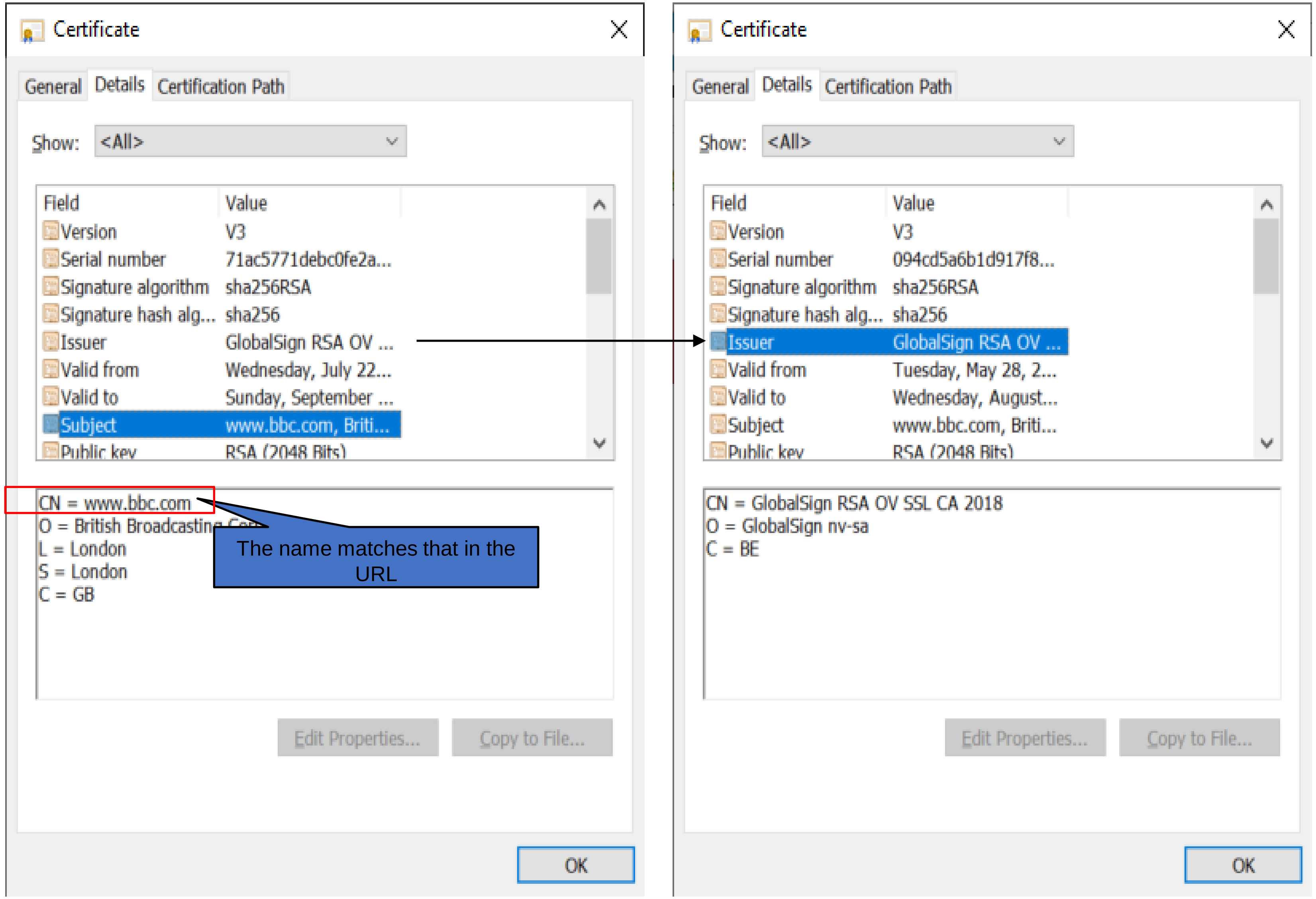
Tuesday
26 May 2020
£2.20
From £1.75 for subscribers

‘I don’t regret what I did.’ Cummings refuses to quit

- Johnson’s top adviser stages extraordinary press conference
- PM says he still backs aide amid ongoing anger over lockdown breach
- Second family trip ‘necessary to check eyesight’ after virus

Rowena Mason
Matthew Weaver

BBC’s certificate and its issuer



Show: <All>

Field	Value
Version	V3
Serial number	71ac5771debc0fe2a...
Signature algorithm	sha256RSA
Signature hash alg...	sha256
Issuer	GlobalSign RSA OV ...
Valid from	Wednesday, July 22...
Valid to	Sunday, September ...
Subject	www.bbc.com, Briti...
Public key	RSA (2048 Bits)

CN = www.bbc.com
O = British Broadcasting Corp
L = London
S = London
C = GB

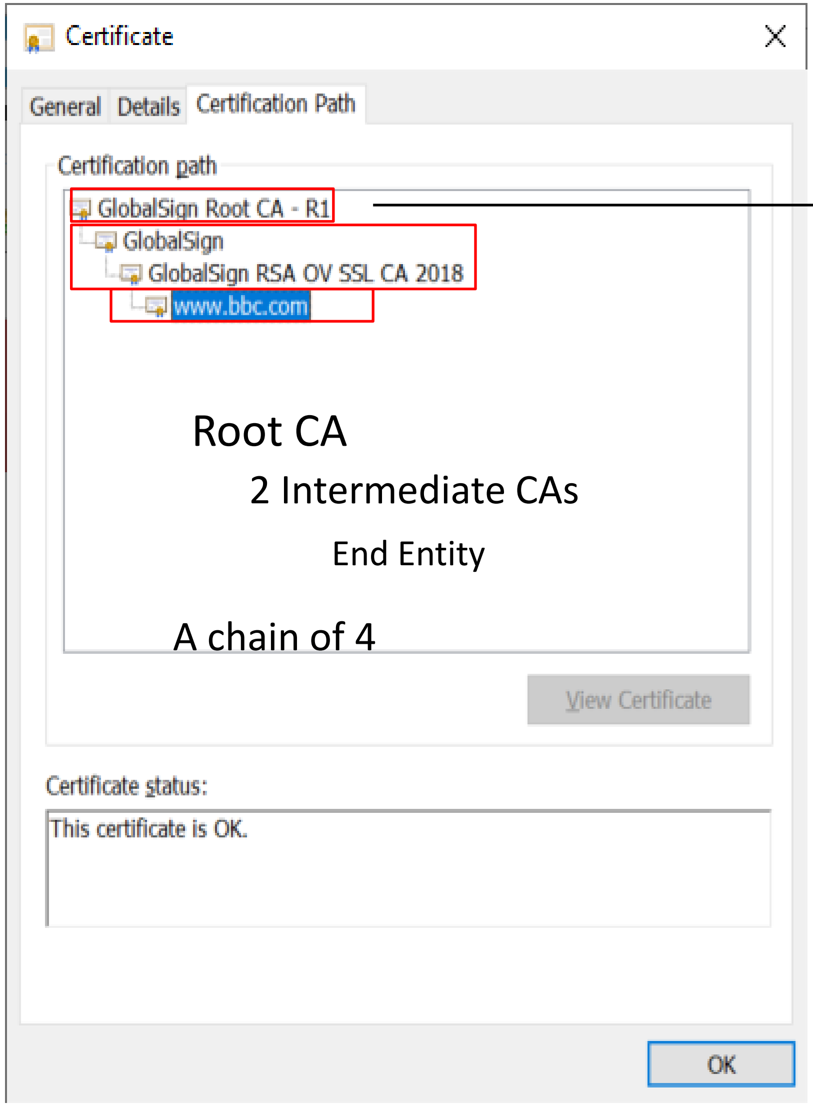
The name matches that in the URL

Show: <All>

Field	Value
Version	V3
Serial number	094cd5a6b1d917f8...
Signature algorithm	sha256RSA
Signature hash alg...	sha256
Issuer	GlobalSign RSA OV ...
Valid from	Tuesday, May 28, 2...
Valid to	Wednesday, August...
Subject	www.bbc.com, Briti...
Public key	RSA (2048 Bits)

CN = GlobalSign RSA OV SSL CA 2018
O = GlobalSign nv-sa
C = BE

Certificate chain and the root CA certificate



Certification path

- GlobalSign Root CA - R1
  - GlobalSign
    - GlobalSign RSA OV SSL CA 2018
      - www.bbc.com

Root CA

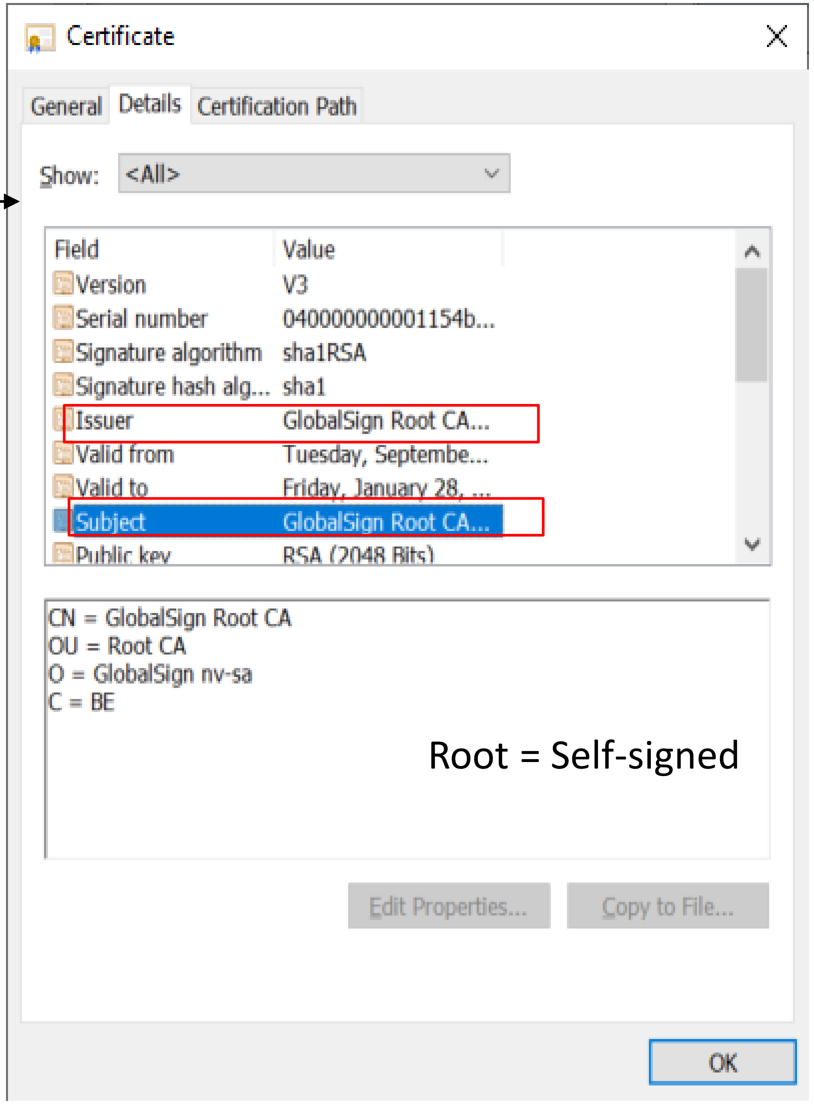
2 Intermediate CAs

End Entity

A chain of 4

Certificate status:

This certificate is OK.



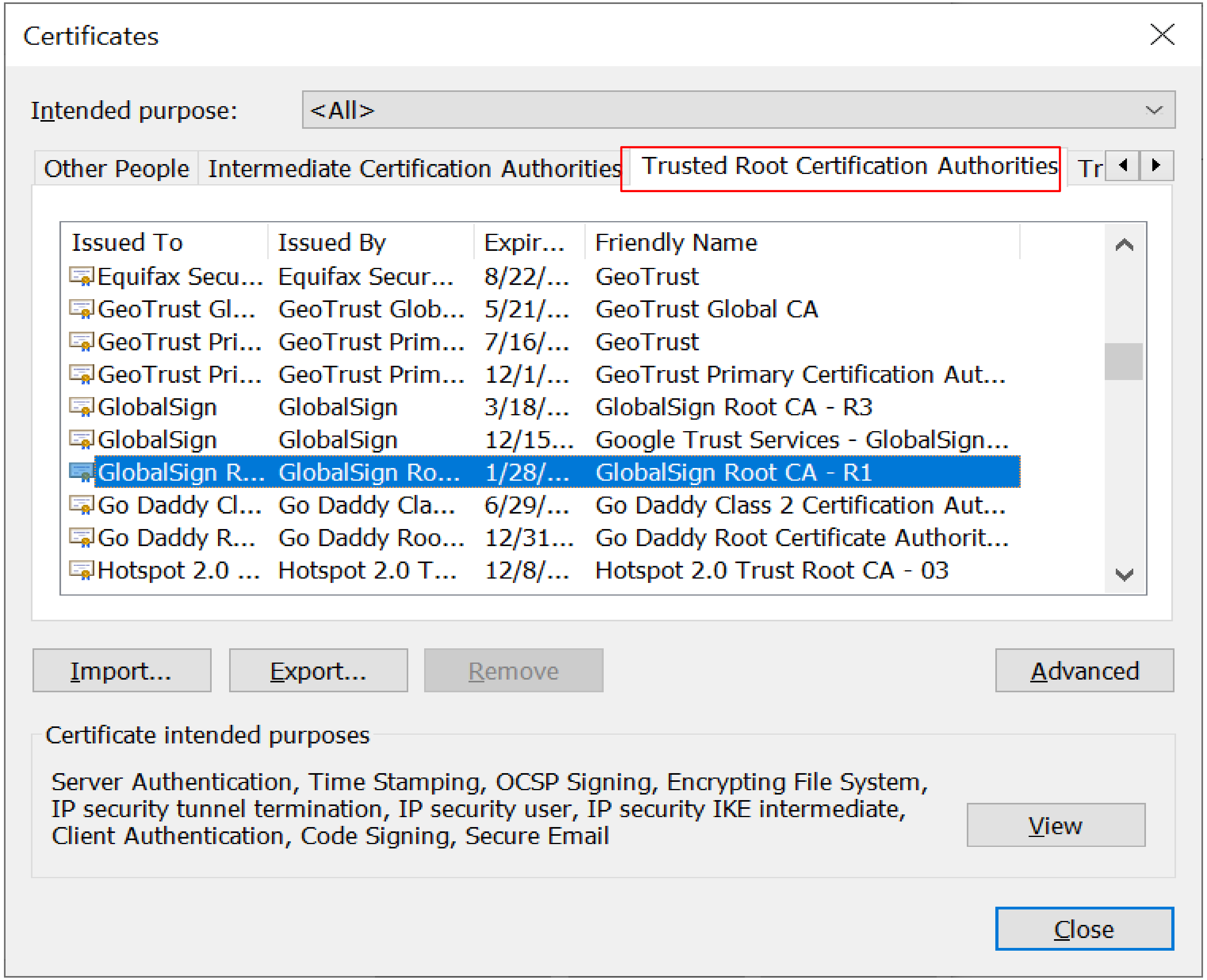
Show: <All>

Field	Value
Version	V3
Serial number	040000000001154b...
Signature algorithm	sha1RSA
Signature hash alg...	sha1
Issuer	GlobalSign Root CA...
Valid from	Tuesday, Septembe...
Valid to	Friday, January 28, ...
Subject	GlobalSign Root CA...
Public key	RSA (2048 Bits)

CN = GlobalSign Root CA
OU = Root CA
O = GlobalSign nv-sa
C = BE

Root = Self-signed

Root CAs in browser's Certificate Store



Certificates

Intended purpose: <All>

Other People
Intermediate Certification Authorities
Trusted Root Certification Authorities
Tr

Issued To	Issued By	Expir...	Friendly Name
Equifax Secu...	Equifax Secur...	8/22/...	GeoTrust
GeoTrust Gl...	GeoTrust Glob...	5/21/...	GeoTrust Global CA
GeoTrust Pri...	GeoTrust Prim...	7/16/...	GeoTrust
GeoTrust Pri...	GeoTrust Prim...	12/1/...	GeoTrust Primary Certification Aut...
GlobalSign	GlobalSign	3/18/...	GlobalSign Root CA - R3
GlobalSign	GlobalSign	12/15...	Google Trust Services - GlobalSign...
GlobalSign R...	GlobalSign Ro...	1/28/...	GlobalSign Root CA - R1
Go Daddy Cl...	Go Daddy Cla...	6/29/...	Go Daddy Class 2 Certification Aut...
Go Daddy R...	Go Daddy Roo...	12/31...	Go Daddy Root Certificate Authorit...
Hotspot 2.0 ...	Hotspot 2.0 T...	12/8/...	Hotspot 2.0 Trust Root CA - 03

Certificate intended purposes

Server Authentication, Time Stamping, OCSP Signing, Encrypting File System, IP security tunnel termination, IP security user, IP security IKE intermediate, Client Authentication, Code Signing, Secure Email

A server wants to establish a secure session with a client using server authentication.

What are the steps?

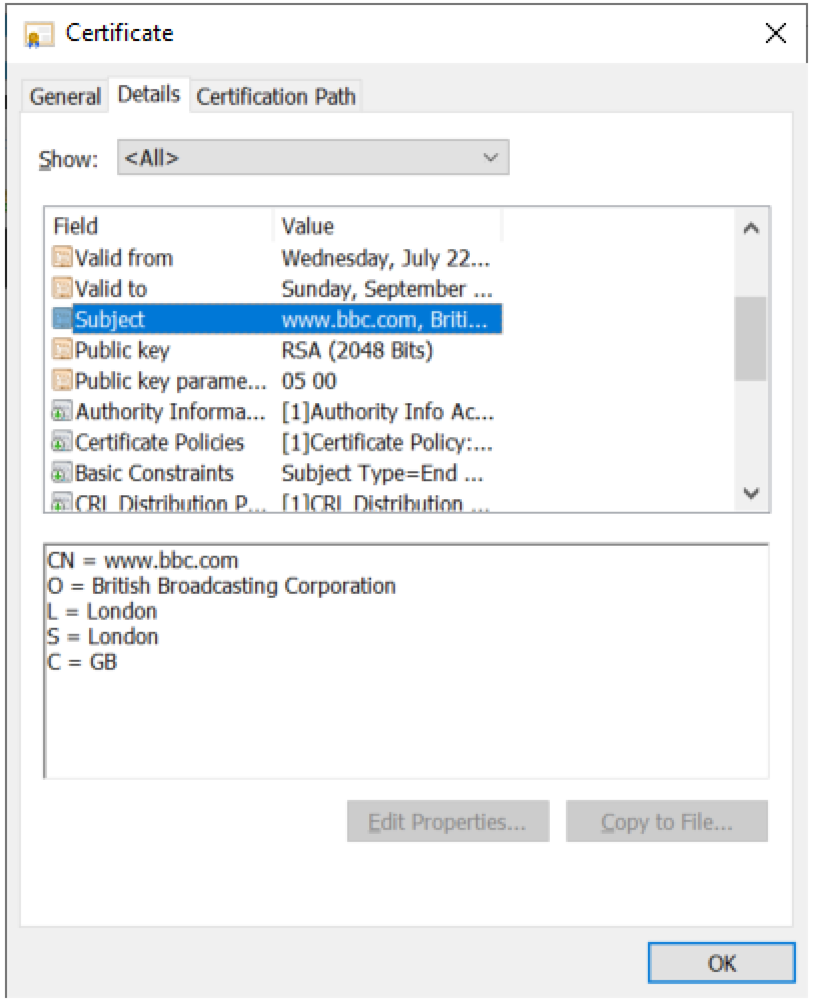
Step 1: Server needs a certificate

- The server needs to obtain a certificate to identify itself. There are different options:
  a) Use utilities from z/OS or other platforms – RACF RACDCERT or System SSL gskkyman, openSSL
     - Simple, but they do not provide any revocation status on the certificate
     - RACDCERT certificates do not have full support on certificate extensions

  b) Buy one from some commercial CAs
     - Pretty expensive
     - Preferred choice if the server is to serve worldwide clients since the root CA is preloaded in most of the browsers

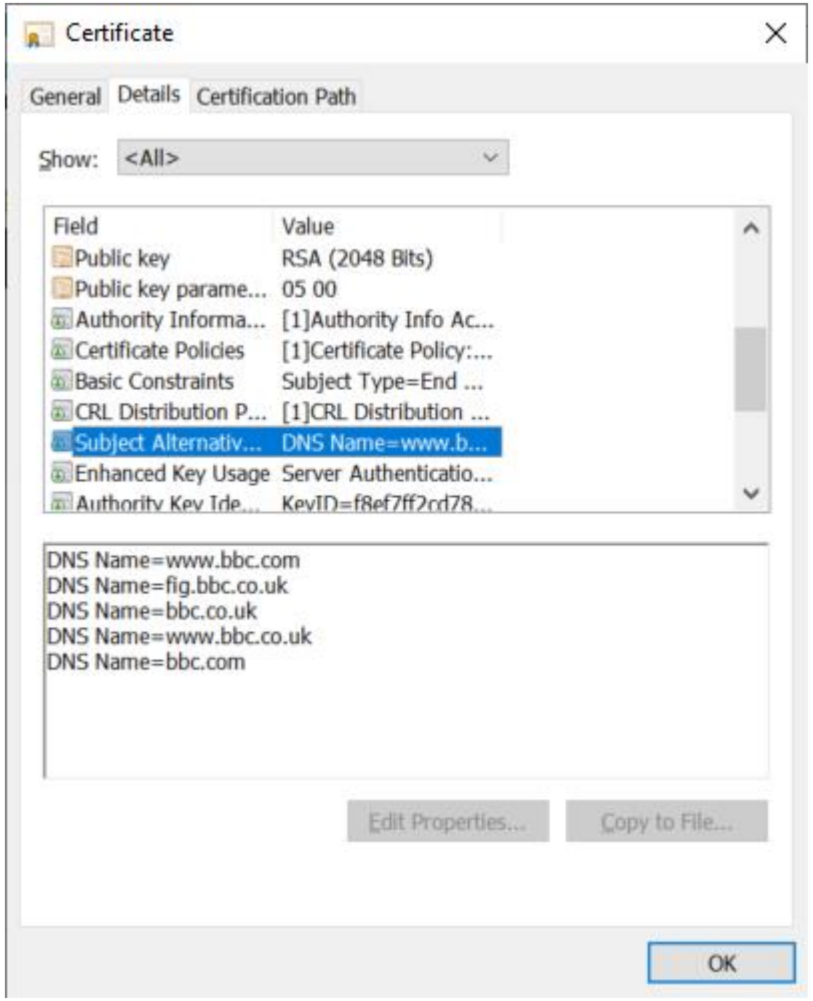
  c) Request one from some internal CA, eg. z/OS PKI Services
     - Needs set up. But if a large number of certificates are needed, it is worth the effort
     - Full support on extensions like Subject Alternate Name

Two fields to match the URL: Common Name, Subject Alternate Name



Field	Value
Valid from	Wednesday, July 22...
Valid to	Sunday, September ...
Subject	www.bbc.com, Briti...
Public key	RSA (2048 Bits)
Public key parame...	05 00
Authority Informa...	[1]Authority Info Ac...
Certificate Policies	[1]Certificate Policy:...
Basic Constraints	Subject Type=End ...
CRL Distribution P...	[1]CRL Distribution ...

```
CN = www.bbc.com
O = British Broadcasting Corporation
L = London
S = London
C = GB
```

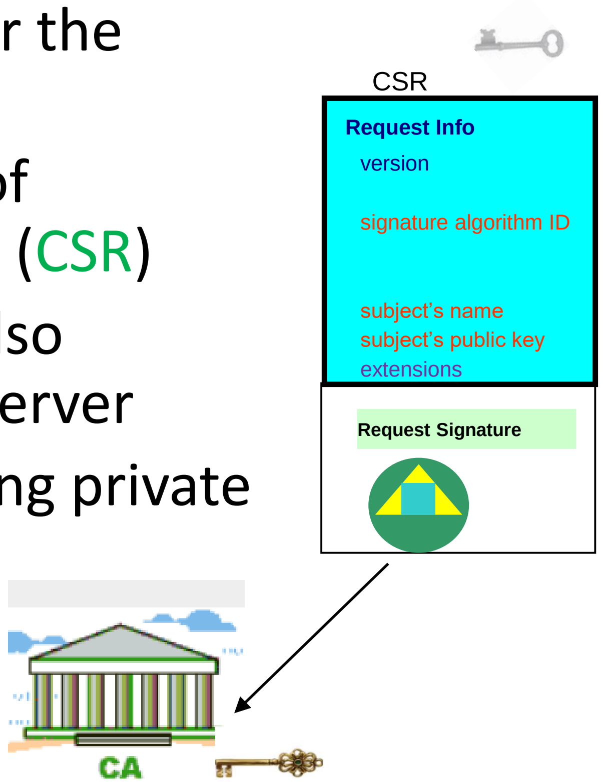


Field	Value
Public key	RSA (2048 Bits)
Public key parame...	05 00
Authority Informa...	[1]Authority Info Ac...
Certificate Policies	[1]Certificate Policy:...
Basic Constraints	Subject Type=End ...
CRL Distribution P...	[1]CRL Distribution ...
Subject Alternativ...	DNS Name=www.b...
Enhanced Key Usage	Server Authenticatio...
Authority Key Ide...	KeyID=f8ef7ff2cd78...

```
DNS Name=www.bbc.com
DNS Name=fig.bbc.co.uk
DNS Name=bbc.co.uk
DNS Name=www.bbc.co.uk
DNS Name=bbc.com
```

Key pair ->CSR->certificate

- Need to have a public private key pair first for the server
  - The key pair is generated in the process of generating the certificate signing request (CSR)
  - The public key is put on the CSR, which also contains identifying information for the server
  - CSR is signed by the server's corresponding private key
  - The private key is put in a safe place!!!
- The CSR is sent to the Certificate Authority



CSR

Request Info

version

signature algorithm ID

subject's name

subject's public key

extensions

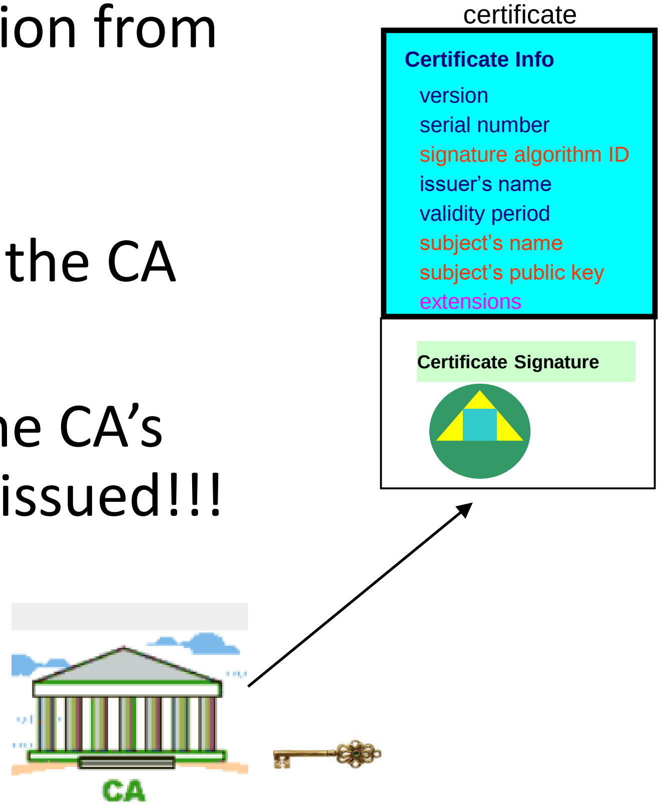
Request Signature

(For simplicity, I assume this CA is self-signed, ie. It is the root)

Key pair ->CSR->certificate

- After the CA validates the CSR, it returns a certificate that contains
  - the public key and the identifying information from the CSR
  - other content that the CA decides
  - the signature created by the private key of the CA

Note: Secure the private key associated with the certificate, especially the CA's. Compromise of the CA's private key invalidates ALL the certificates it has issued!!!



certificate

Certificate Info

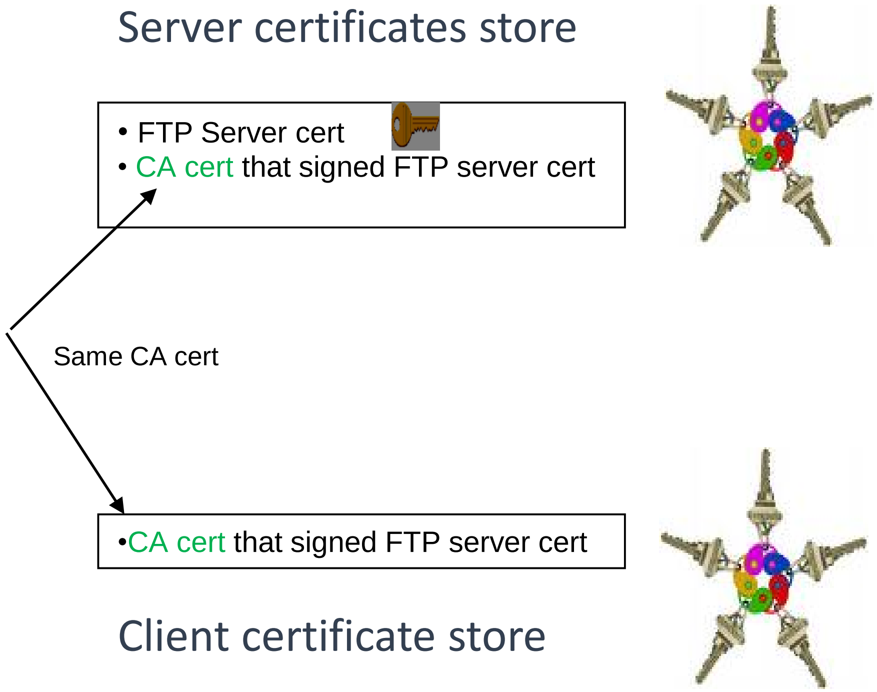
version
serial number
signature algorithm ID
issuer's name
validity period
subject's name
subject's public key
extensions

Certificate Signature

Step 2: Both server and client need certificate stores

- Certificate must be placed in a certificate store / key ring/ key database before it can be used by an application to perform identification and validation
- The server admin
  - sets up a certificate store /key ring / key database with the whole chain (this is a chain of 2):
    - the server certificate
    - the issuer CA certificate (it is also the root in this case)
  - sends the root CA certificate to the client admin (not the server certificate !!!)
- The client admin
  - sets up a key ring / key database / certificate store with this root CA certificate:
    - the root CA certificate (this is also the issuer certificate in this case)

- Notice that in the case of a chain of N, the server keyring should contain N certificates, but the client keyring only needs the root CA certificate no matter how long the chain is.


Server certificates store

- FTP Server cert
- CA cert that signed FTP server cert

Same CA cert

- CA cert that signed FTP server cert

Client certificate store

Certificate verification

Client perform these checks on the **server** certificate: (for server authentication)

- **Validation checks**
  - Check the certificate’s integrity by verifying the signature on the certificate – is it really issued by the CA it claims?
  - Check if the certificate is expired by verifying the expiration date on the certificate
  - Check if the certificate has been revoked

  Note: The validation checks apply to the issuer certificate(s) too. All the certificates in the chain have to pass the above checks

- **Trust check** - check if the root CA certificate is trusted
  - Is the root CA certificate of the server certificate in the client’s keyring?

Types of z/OS certificate stores

- RACF Key Ring – real or virtual
- ICSF PKCS11 Token
- System SSL Key database
- PKCS12 package

RACF keyring is the most popular certificate store on z/OS

- **Real keyring**
  - created by RACDCERT id(<ring owner>) ADDRING (<ring name>)
  - specified its name on application configuration with
    - <ring owner>/<ring name>, eg. **FTPID/ftpRing**
  - can be created before or after the certificates have been obtained
- **Virtual keyring,** no need to create, just a collection of certificates under the owner ID
  - specified its name on application configuration with
    - <ring owner>/*, eg. ***AUTH*/*** , CA's virtual key ring
- Key rings are protected by RACF resource profiles
  - application ID needs read access to the profiles in the RDATALIB or FACILITY class
    - RDATALIB: <ring owner>.<ring name>.LST – Granular control (Since 2008)
      - RDATALIB must be raclisted
    - FACILITY: IRR.DIGTCERT.LISTRING, IRR.DIGTCERT.GENCERT – Global control (Original support)

Some useful RACDCERT command tips

- RACDCERT <owner> <function> <other function specific sub keywords>
  - Owner: ID(RACF id), eg. ID(ftpserver), or predefined owner - CERTAUTH, SITE, MULTIID
  - Function: 26 functions - GENCERT, GENREQ, ADD, ADDRING, CONNECT, LISTCHAIN…
- If owner is not specified, it defaults to the command issuer. If Mary issues the commands:
  - RACDCERT ID(John) LISTCHAIN(LABEL(‘mycert’))
    - Display John’s mycert and its issuer(s) cert(s)
  - RACDCERT LISTCHAIN(LABEL(‘mycert’))
    - Display Mary’s mycert and its issuer(s) cert(s)
- Don’t confuse RACDCERT ADD with RACDCERT IMPORT – ADD a cert in a dataset to RACF, IMPORT a cert from ICSF PKCS11 token to RACF

Certificate Formats

- X.509 certificates can be packaged differently
  - Single certificate (eg. .cer, .crt, .pem)
  - PKCS#7 certificate package (eg. .p7b)
    - Contains end entity certificate and its issuer(s)
  - PKCS#12 certificate package (eg. .p12, .pfx)
    - Similar to PKCS#7, but also contains the private key associated with the end-entity certificate.
    - Packaged protected by a password
- Package can be in binary or Base64 encoded format (containing Aa-Zz,0-9,/,+ (= is for padding) for easy cut and paste)

```
-----BEGIN CERTIFICATE-----
MIICPTCCAaagAwIBAgIIR49S4QANLvEwDQYJKoZIhvcNAQEFBQAwNzELMAkGA1UE
BhMCVVMxDTALBgNVBAoTBFRlc3QxGTAXBgNVBAMMEFRlc3Rfc2VsZl9zaWduZWQw
HhcNMDgwMTE3MTMwNjQxWhcNMDkwMTE2MTMwNjQxWjA3MQswCQYDVQQGEwJVUzEN
MAsGA1UEChMEVGVzdDEZMBcGA1UEAwwQVGVzdF9zZWxmX3NpZ25lZDCBnzANBgkq
hkiG9w0BAQEFAAOBjQAwgYkCgYEA9tKOv5gLaceozMfMeVd891fCjBVoR+dpzhwK
R2B/QcQYBGLfqS4YM/wGSh6YrmVygO0VxocriySbcxRuBayw3pE4/3JI2myINmLp
bFIdPCnqk/qvFK+1N+nrEnBK9yls7NmxDIuQQfFsX/o/DpoxwxzwXf+JbWDwirQR
NyLiTGMCAwEAAaNSMFAwHQYDVR0OBBYEFAwDFLjOUCRa62BVs3jVyHewuOWEMB8G
A1UdIwQYMBaAFAwDFLjOUCRa62BVs3jVyHewuOWEMA4GA1UdDwEB/wQEAwIE8DAN
BgkqhkiG9w0BAQUFAAOBgQAC5sW1f3EdE0k9zc8wKNt1sczWkQBrVy4Rdrl7ERqN
D2OfkBJQuXiNwN18pF6WPWfYG80MNwhP4oJSVePnzElh4Wzi2wl/zI8rINSW7px3
w16lz+8jEI84q/N0q0toPTAtEb6fIzwjkLtctt3oF+IjunvE5QoRsXRJbbTMD/EG
jw==
-----END CERTIFICATE-----
```

Using what you have learnt to solve a handshake problem from a certificate perspective

Steps to tackle from server side

- Find out which party is the server, which party is the client

- Server side:

1. What is the configuration file which include the keyring / database information?
2. What is the keyring name? Who is the keyring owner?
3. Does the keyring contain all the needed certificates?
4. Which one is the server certificate? Who owns it?

   Usage is Personal

   Marked as DEFAULT (the most popular setting)
5. Does the server certificate have a private key associated with it and is its status TRUST?
6. What ID will be using the keyring? Does it have access to the private key?
   - Access to keyring means access to certificates in the keyring, but not the access to their private keys
   - Simpler set up if the accessing ID is the owner of the certificate, and owner of the keyring
   - If the access control is through RDATALIB, make sure it is active and raclisted

Example on tracing AT-TLS handshake problem based on RACF key ring

Server side:

From PROFILE.TCPIP.SERVER 1

2 2

```
TTLSKeyRingParms
  {
     Keyring                XXServer/XXServerRing
  }
```

2 2

RACDCERT ID(XXServer) LISTRING(XXServerRing)

```
Digital ring information for user XXServer:
```

3

```
 Ring:
      >XXServerRing<
 Certificate Label Name                        Cert Owner     USAGE        DEFAULT
 --------------------------------            ------------   --------     -------
 SSL Cert                                      ID(XXServer)   PERSONAL       YES
 Local Intermediate CA                         CERTAUTH       CERTAUTH       NO
 Local Root CA                                 CERTAUTH       CERTAUTH       NO
```

4 4 👍

RACDCERT ID(XXServer) LISTCHAIN(LABEL(‘SSL Cert’))

```
Certificate 1:
Digital certificate information for user XXServer:

    Label: SSL Cert
    Certificate ID: 2QbmxcLi2eXi4tNAw4WZo0BA
5   Status: TRUST 👍
    Start Date: 2020/04/17 01:00:00
    End Date:   2021/04/16 00:59:59
    …
5   Private Key: YES 👍
    Ring Associations:
      Ring Owner: XXServer
    Ring:
       >XXServerRing<

Certificate 2:
Digital certificate information for CERTAUTH:

    Label: Local Intermediate CA
    Certificate ID: 2QinxcLi2eYj4tMAw4WZo0BD
    Status: TRUST
    Start Date: 2015/02/17 01:00:00
    End Date:   2025/12/31 00:59:59
    …
    Private Key: NO
    Ring Associations:
      Ring Owner: XXServer
    Ring:
       >XXServerRing<
```

```
Certificate 3:
Digital certificate information for
CERTAUTH:

  Label: Local Root CA
  Certificate ID: 2QkkxcLi2eZj4tMAw4WZo0BE
  Status: TRUST
  Start Date: 2015/01/01 01:00:00
  End Date:   2035/12/31 00:59:59
  …
  Private Key: NO
  Ring Associations:
    Ring Owner: XXServer
  Ring:
     >XXServerRing<

 Chain information:
   Chain contains 3 certificate(s), chain
is complete 👍
   Chain contains ring in common:
XXServer/XXServerRing 👍
```

RLIST **RDATALIB** XXServer.XXServerRing.LST 6

```
CLASS       NAME
-----       ----
RDATALIB    XXSERVER.XXSERVERRING.LST

LEVEL   OWNER        UNIVERSAL ACCESS   YOUR ACCESS   WARNING
-----   --------     ----------------   -----------   -------
…
USER        ACCESS
----        ------
XXSERVER    READ        👍
YYSERVER    UPDATE      ←  if YYSERVER accesses XXSERVER's keyring,
                          XXSERVER's private key is involved, need
                          UPDATE
…
```

** Make sure the RDATALIB class is active and raclisted!!!

SETR LIST

…

ACTIVE CLASSES =…..RDATALIB… 👍

…

SETR RACLIST CLASSES = … RDATALIB… 👍

SETR RACLIST(RDATALIB) REFRESH 👍

Steps to tackle from client side

- Client side:
  1. What is the configuration file which include the keyring / database information?
  2. What is the keyring name? Who is the keyring owner?
  3. Are the certificates CERTAUTH certificates?
  4. Which one is the root CA certificate of the server? Is its status TRUST?
  5. What ID will be using the keyring? Does it have access to the keyring?
     - Access to keyring means access to certificates in the keyring
     - If the access control is through RDATALIB, make sure it is active and raclisted

Client uses a real RACF keyring

1

Client side:

From PROFILE.TCPIP.CLIENT

2 2

```
TTLSKeyRingParms
   {
      Keyring                 XXClient/XXClientRing
   }
```

2 2

RACDCERT ID(XXClient) LISTRING(XXClientRing)

```
Digital ring information for user XXClient:

  Ring:
       >XXClientRing<
  Certificate Label Name                   Cert Owner     USAGE      DEFAULT
  --------------------------------         ------------   --------   -------
  XXServer Root CA                         CERTAUTH       CERTAUTH     NO
```

4 3

RACDCERT CERTAUTH LIST(LABEL(‘XXServer Root CA’))

```
Digital certificate information for
CERTAUTH:

  Label: XXServer Root CA
  Certificate ID: 2QkkxcLi2eZj4tMAw4WZo0BE
  Status: TRUST 👍
  Start Date: 2015/01/01 01:00:00
  End Date:   2035/12/31 00:59:59
  Serial Number:
    …
  Issuer’s Name:
    …
  Subject’s Name:
  …
  Private Key: NO
  Ring Associations:
    Ring Owner: XXClient
  Ring:
     >XXClientRing<
```

4

Make sure this is the server’s root CA sent by the server side by checking fields like:

- serial number,
- issuer’s name,
- subject's name

RLIST **RDATALIB** XXClient.XXClientRing.LST 5

```
CLASS        NAME
-----        ----
RDATALIB     XXCLIENT.XXCLIENTRING.LST

LEVEL   OWNER        UNIVERSAL ACCESS   YOUR ACCESS   WARNING
-----   --------     ----------------   -----------   -------
…
USER        ACCESS
----        ------
XXCLIENT    READ       👍
YYCLIENT    READ  ←   if YYCLIENT accesses XXCLIENT’s keyring,
                          also just need READ
…
```

**** Make sure the RDATALIB class is active and raclisted!!!**

SETR LIST

…

ACTIVE CLASSES =…..RDATALIB… 👍

…

SETR RACLIST CLASSES = … RDATALIB… 👍

SETR RACLIST(RDATALIB) REFRESH 👍

Client uses a virtual RACF keyring

Client side:

From PROFILE.TCPIP.CLIENT

1

```
TTLSKeyRingParms
   {
      Keyring                    *AUTH*/*
   }
```

2

3

RACDCERT CERTAUTH LIST

4

```
Digital certificate information for CERTAUTH:
Label: Verisign Class 3 Primary CA
…
Label: XX Root CA
…
Label: YY Root CA
…
Label: XXServer Root CA
```

Make sure one of these CA certificates is the root CA certificate that the server side sent this by checking the fields like:

- serial number,
- issuer’s name,
- subject's name

RLIST **RDATALIB** CERTIFAUTH.IRR_VIRTUAL_KEYRING.LST 5

```
CLASS         NAME
-----         ----
RDATALIB      CERTIFAUTH.IRR_VIRTUAL_KEYRING.LST

LEVEL   OWNER         UNIVERSAL ACCESS   YOUR ACCESS   WARNING
-----   --------      ----------------   -----------   -------
…
USER         ACCESS
----         ------
XXCLIENT     READ     👍
…
```

OR

*(**For the client side, use the old FACILITY class for control is fine)*

RLIST **FACILITY** IRR.DIGTCERT.LISTRING 5

```
CLASS         NAME
-----         ----
FACILITY      IRR.DIGTCERT.LISTRING

LEVEL   OWNER         UNIVERSAL ACCESS   YOUR ACCESS   WARNING
-----   --------      ----------------   -----------   -------
…
USER         ACCESS
----         ------
XXCLIENT     READ     👍
…
```

Some key points

- Keyring set up is the first area to debug in TLS problem
- Three IDs you need to find out for the server side
  - Keyring owner
    - from System SSL log
      - gsk_open_keyring: Keyring '<ring owner>/<ring name>' (if you include the ring owner in the configuration file)
      - gsk_dll_init_once(): Job name <jobname>. The owner is indirectly found from the job submitter based on the job name
  - Certificate owner
    - from RACDCERT LISTRING and LIST
  - Access ID that accesses the keyring and private key (ie the ID reads the configuration setup)
    - from TLS log: message EZD1286I USERID:<userid>

They don't need to be the same, but simpler if all of them are the same

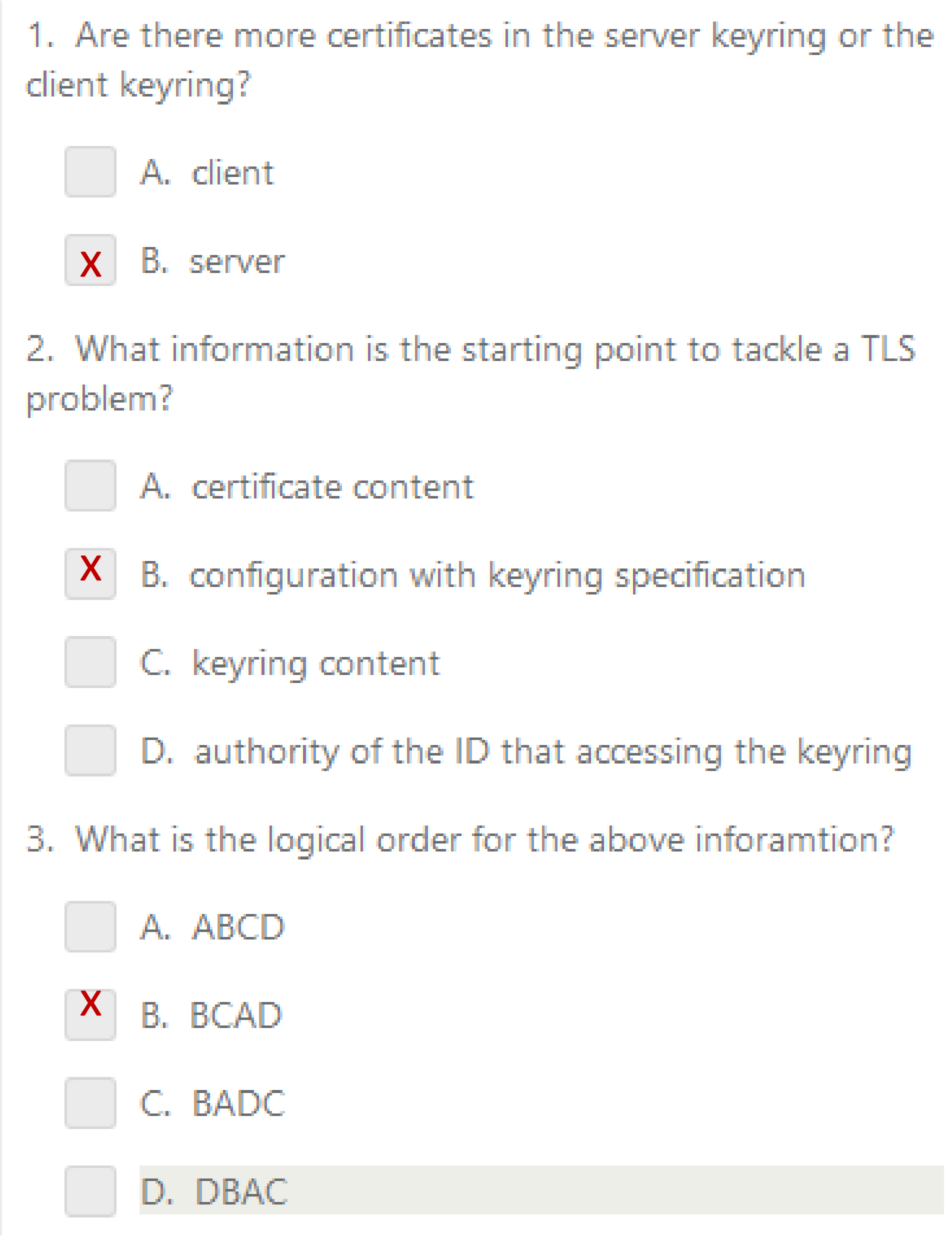
Some key points

- Before adding certificate(s) to RACF, use RACDCERT CHECKCERT on the dataset containing the certificate(s) to check if they already exist
- Use RACDCERT LISTCHAIN to list the certificate chain. But if there are more than one chain, it may not display the one you expected. It uses the one exists earlier to form the chain
- Keep the minimum number of certificates in a keyring. Unnecessary certificates affect handshake performance and may even cause outage
- RACF provides the RACF_CERTIFICATE_EXPIRATION Health Check showing expiring and expired certificates
  - Don’t wait till the last minute
  - Remove the expired one from the keyring, and:
    - Delete it from RACF DB if it is only used for TLS process, or
    - RACDCERT ALTER its status to NOTRUST if you want to keep it (for a while)

Some key points

- It is the responsibility of the server side to send the root certificate (in a file) to the client side **before** the communication occurs
- Once you are sure keyring is set up correctly, then you can proceed to debug the other areas like the cipher suite

How much do you remember?



1. Are there more certificates in the server keyring or the client keyring?

□ A. client

☒ B. server

2. What information is the starting point to tackle a TLS problem?

□ A. certificate content

☒ B. configuration with keyring specification

□ C. keyring content

□ D. authority of the ID that accessing the keyring

3. What is the logical order for the above inforamtion?

□ A. ABCD

☒ B. BCAD

□ C. BADC

□ D. DBAC

References

- **Cryptographic Server Manual**

  Cryptographic Services PKI Services Guide and Reference

  https://www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/zOSV2R4sa232286/$file/ikya100_v2r4.pdf

  Cryptographic Services System Secure Sockets Layer Programming

  https://www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/zOSV2R4sc147495/$file/gska100_v2r4.pdf

- **Security Server Manuals:**

  RACF Command Language Reference

  https://www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/zOSV2R4sa232292/$file/icha400_v2r4.pdf

  RACF Security Administrator's Guide

  https://www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/zOSV2R4sa232289/$file/icha700_v2r4.pdf

- **RFCs**

  RFC5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

  https://tools.ietf.org/html/rfc5280

References

- **IBM Enterprise Knights videos on digital certificates:**

  https://ek-ibmz.mybluemix.net/video/c57660745a547e504d54793083a97b0d

  https://ek-ibmz.mybluemix.net/video/d399cee97db684bbf4f0f4e2b42cff15

- IBM Hot Topics

Issue #29: Drowning in digital certificates? Here's a lifeline!
http://publibfp.dhe.ibm.com/epubs/pdf/e0z3n110.pdf

Issue #21: RACDCERT tipbits. x509 digital certificate technology
http://publibz.boulder.ibm.com/epubs/pdf/e0z2n1a0.pdf

Issue #19: Grow your own. Using locally generated digital certificates
http://publibz.boulder.ibm.com/epubs/pdf/e0z2n190.pdf

Issue #14: Security alert: Do you want to proceed?
http://publibz.boulder.ibm.com/epubs/pdf/e0z2n161.pdf

- **IBM PKI Redbooks**

  Managing Digital Certificates across the Enterprise

  https://www.redbooks.ibm.com/abstracts/sg248336.html?Open

  z/OS PKI Services: Quick Set-up for Multiple CAs

  https://www.redbooks.ibm.com/abstracts/sg248337.html?Open

Redbooks

Managing Digital Certificates across the Enterprise

Security

z Systems

IBM Redbooks

Redbooks

z/OS PKI Services
Quick Set-up for Multiple CAs

Security

z Systems

IBM Redbooks

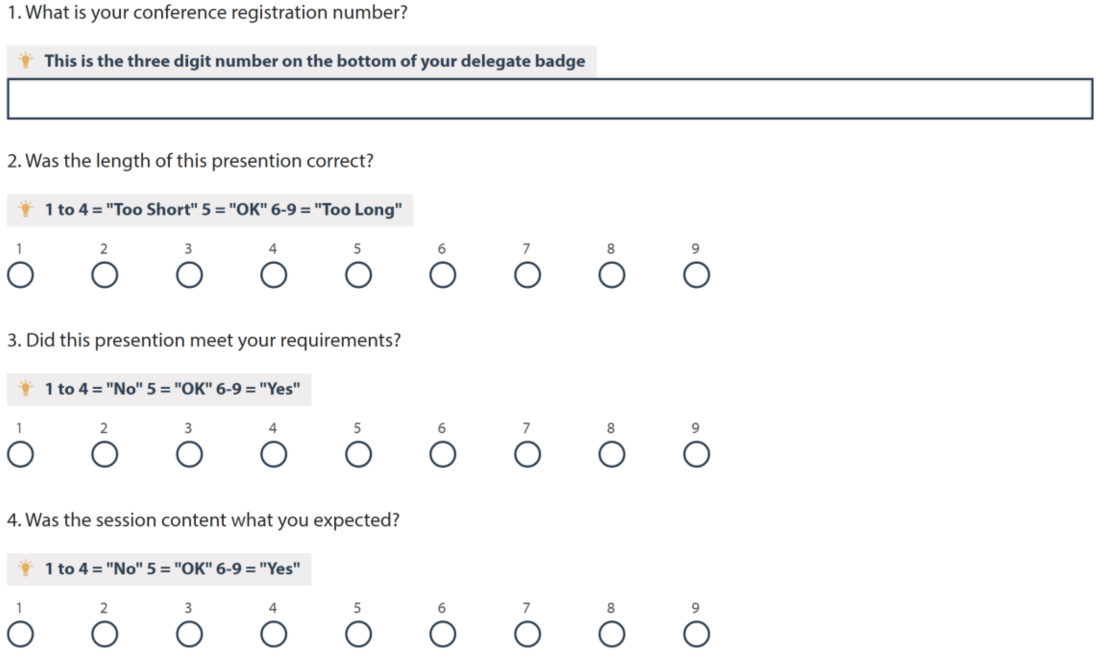
Your turn ☺
Questions?

GUIDE SHARE EUROPE
UK REGION

Please submit your session feedback!

- Do it online at http://conferences.gse.org.uk/2020/feedback/5BE

- This session is 5BE



1. What is your conference registration number?

This is the three digit number on the bottom of your delegate badge

2. Was the length of this presention correct?

1 to 4 = "Too Short" 5 = "OK" 6-9 = "Too Long"

1	2	3	4	5	6	7	8	9
○	○	○	○	○	○	○	○	○

3. Did this presention meet your requirements?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1	2	3	4	5	6	7	8	9
○	○	○	○	○	○	○	○	○

4. Was the session content what you expected?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1	2	3	4	5	6	7	8	9
○	○	○	○	○	○	○	○	○

GSE UK Conference 2020 Charity

- The GSE UK Region team hope that you find this presentation and others that follow useful and help to expand your knowledge of z Systems.
- Please consider showing your appreciation by kindly donating a small sum to our charity this year, NHS Charities Together. Follow the link below or scan the QR Code:

http://uk.virginmoneygiving.com/GuideShareEuropeUKRegion