

Secure Sockets Layer

Secure Sockets Layer (SSL)

Morag Hughson (z/OS) and Mike Horan (Other platforms)

Channel Security - Agenda

- **Security Problems**
- **Current MQSeries Channel Security**
- **Secure Sockets Layer (SSL)**
- **Performance Data Summary**
- **WebSphere MQ and SSL**
 - **Planning Tasks**
 - **WebSphere MQ Configuration Tasks**
 - **Security Administration Tasks**

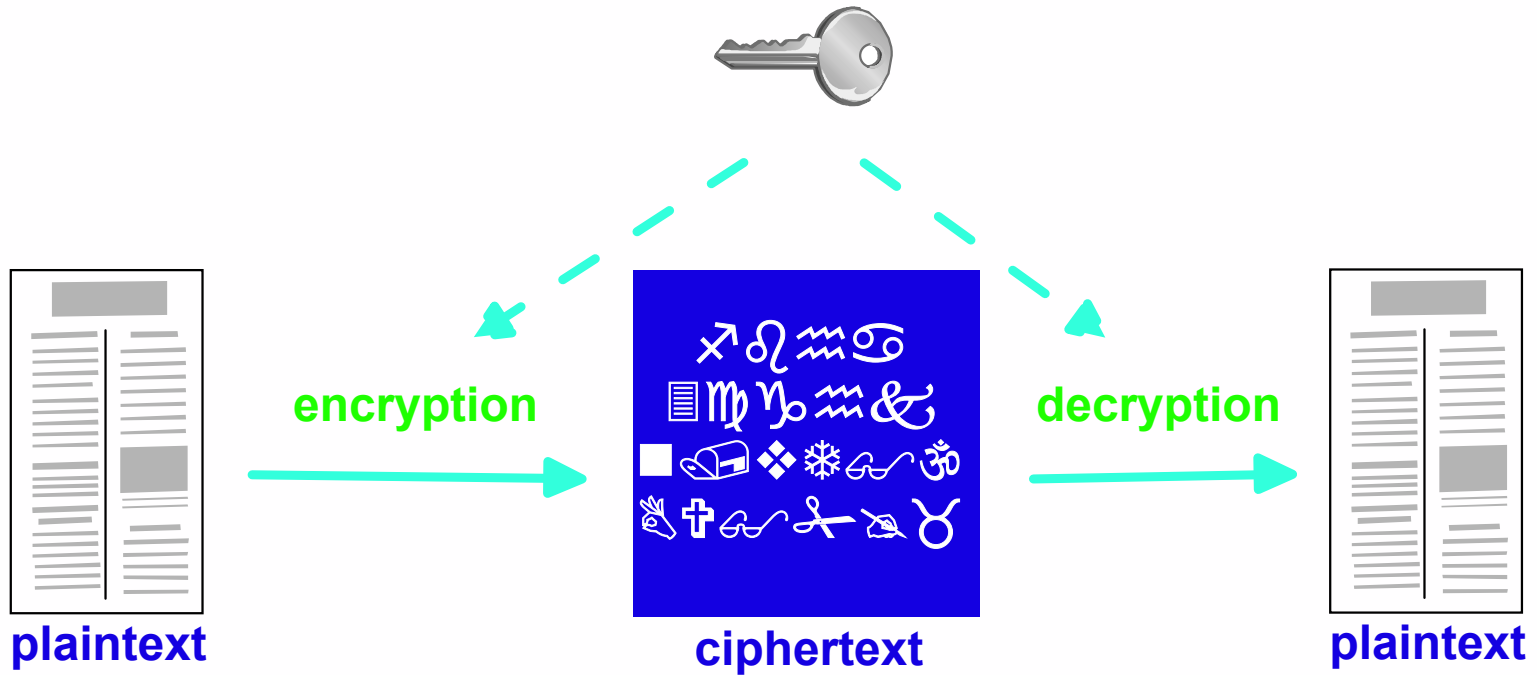


Security Problems

Security Problems

- **Eavesdropping**
 - How do I stop someone from seeing the information I send?
- **Tampering**
- **Impersonation**

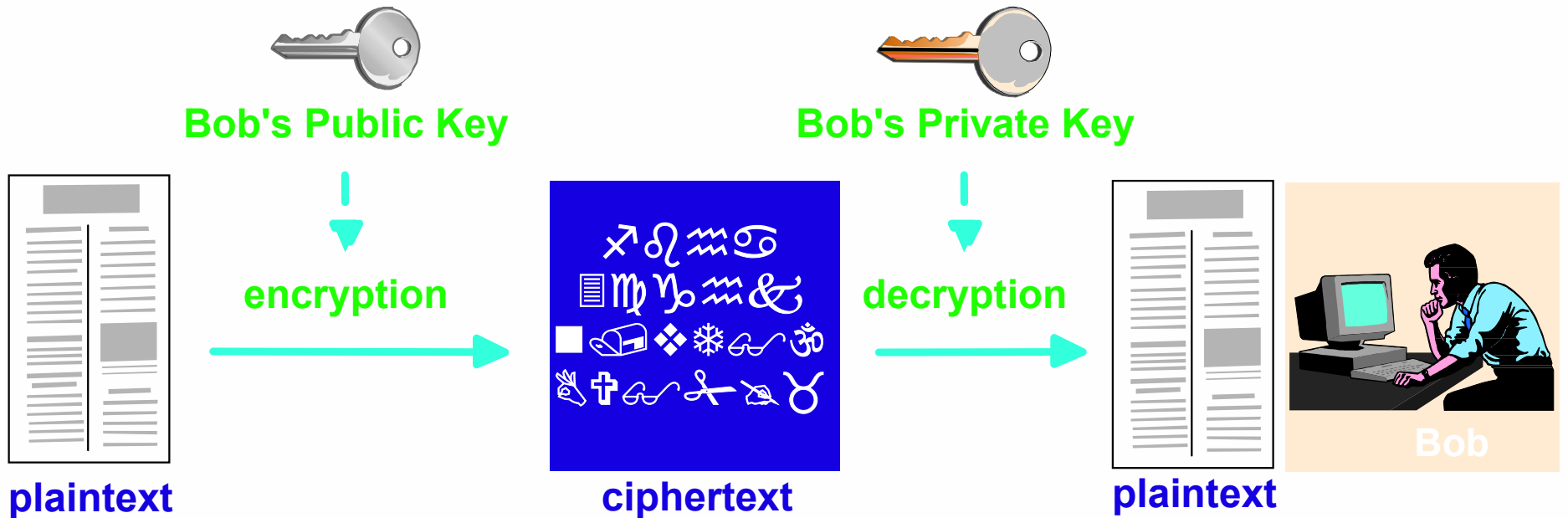
Cryptography



Symmetric Key

Secret Key

Cryptography



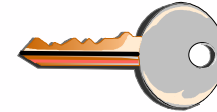
Asymmetric Key

Public/Private Key Pairs

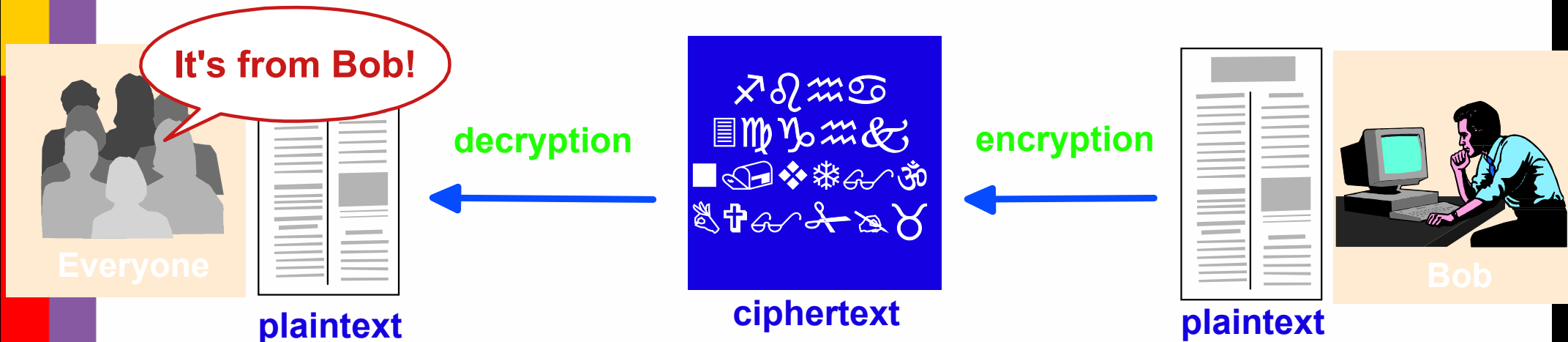
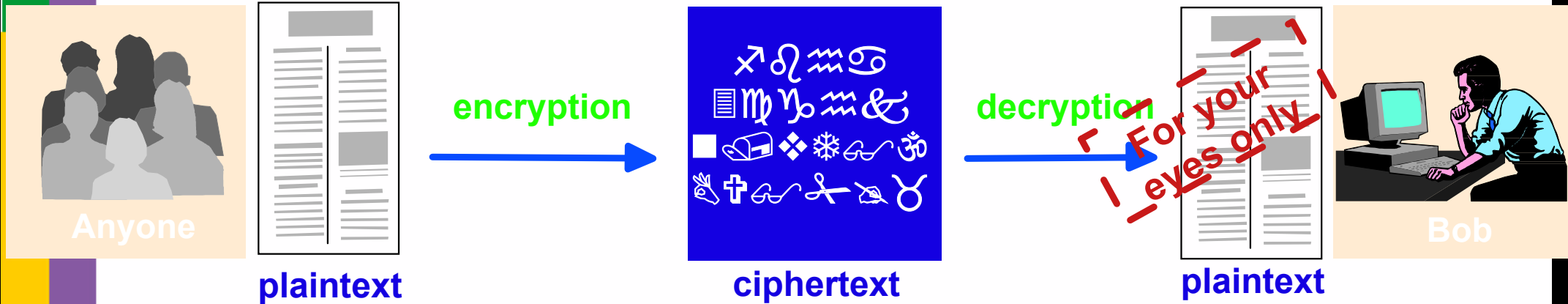
Asymmetric Key - Use both ways



Bob's Public Key



Bob's Private Key



Cryptography

- **Symmetric Keys - Secret Keys**

- Relatively fast
- Poses key delivery challenges when faced with large numbers of senders/receivers
- The key has to be known only by the sender/receiver...

- **Asymmetric Keys - Public/Private Key Pairs**

- Message encrypted with one key can only be decrypted by the other one
- Slower than secret-key cryptography
- Designed to accommodate key delivery and scalability

- Asymmetric Keys can be used to solve the key distribution challenges associated with symmetric keys

- **Standard Key Sizes**

- 512 bits Low-strength key
- 768 bits Medium-strength key
- 1024 bits High-strength key

Security Problems

- **Eavesdropping**

- How do I stop someone from seeing the information I send?

- **Tampering**

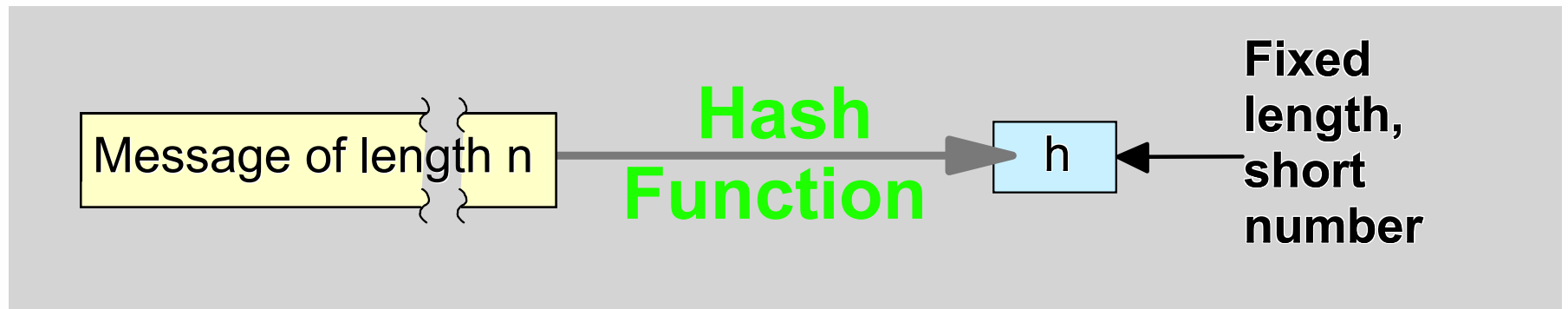
- How can I detect if someone has intercepted my information and changed it?

- **Impersonation**

Hash Function

- **Hash Function**

- **Computes the message digest or Message Authentication Code (MAC)**
- **Easy to compute**
- **Very difficult to reverse**
- **It should be computationally infeasible to find two messages that hash to the same thing.**



Security Problems

- **Eavesdropping**

- How do I stop someone from seeing the information I send?

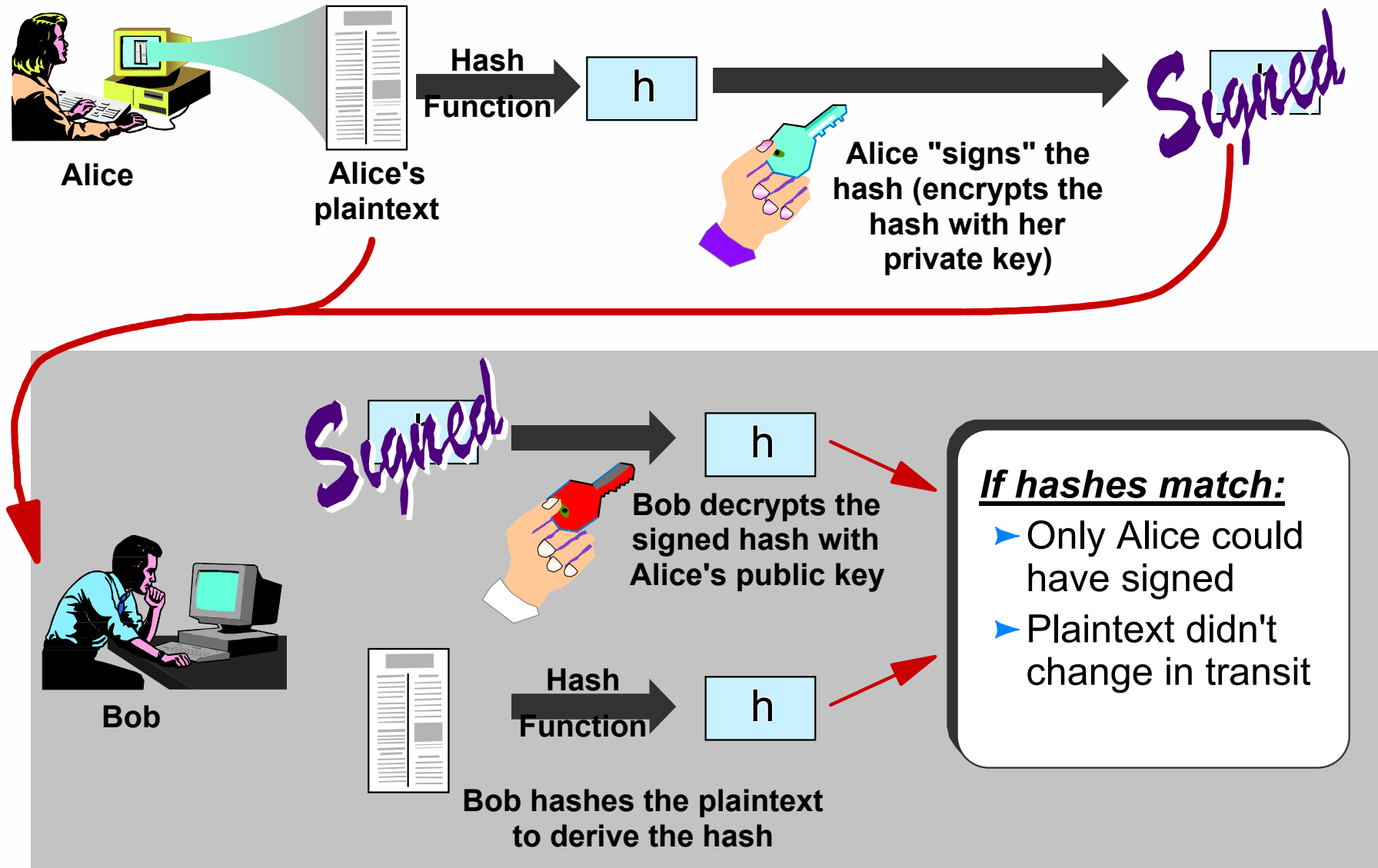
- **Tampering**

- How can I detect if someone has intercepted my information and changed it?

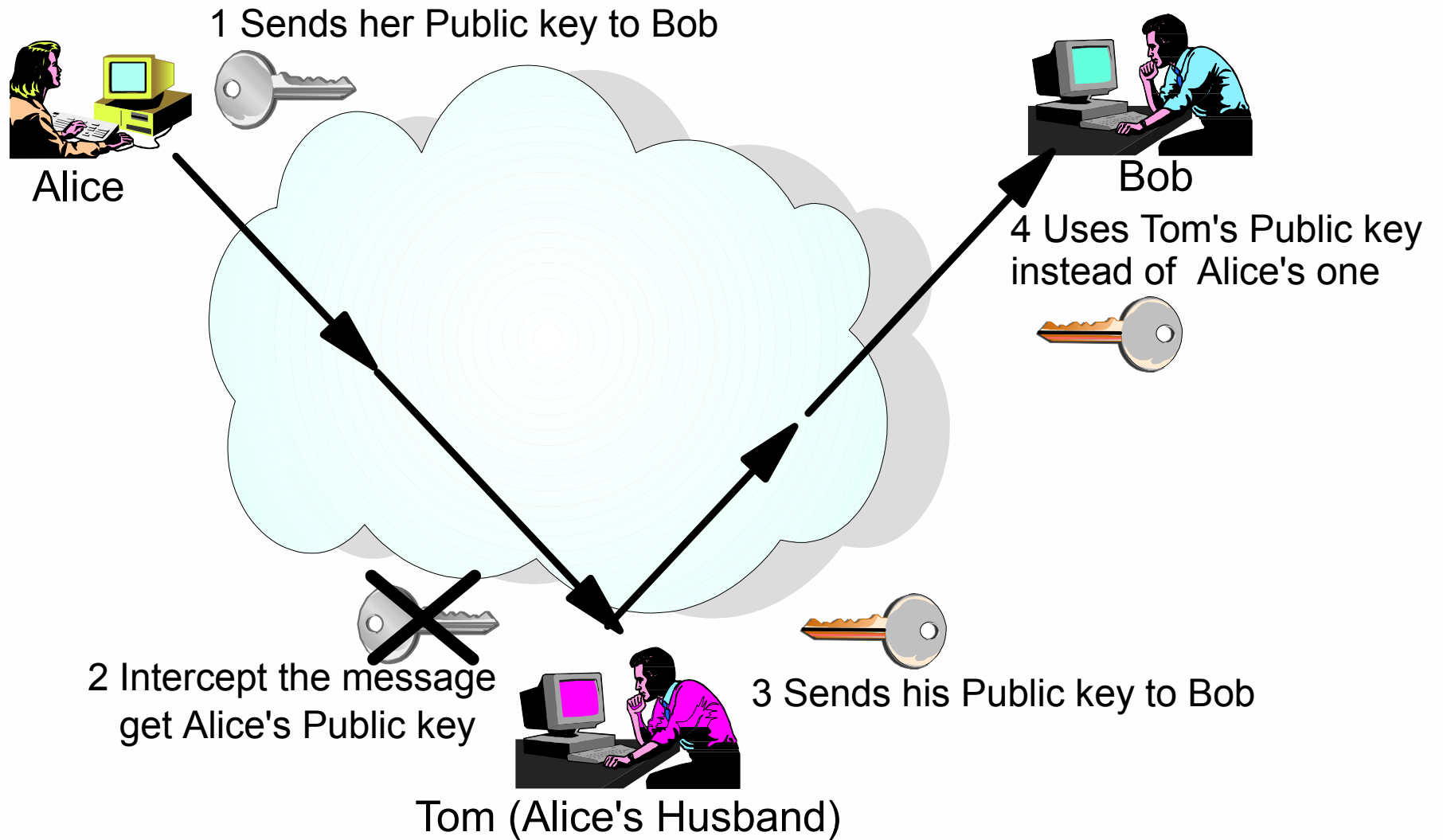
- **Impersonation**

- How can I be sure who the information is from?
- How can I be sure who I am exchanging information with?

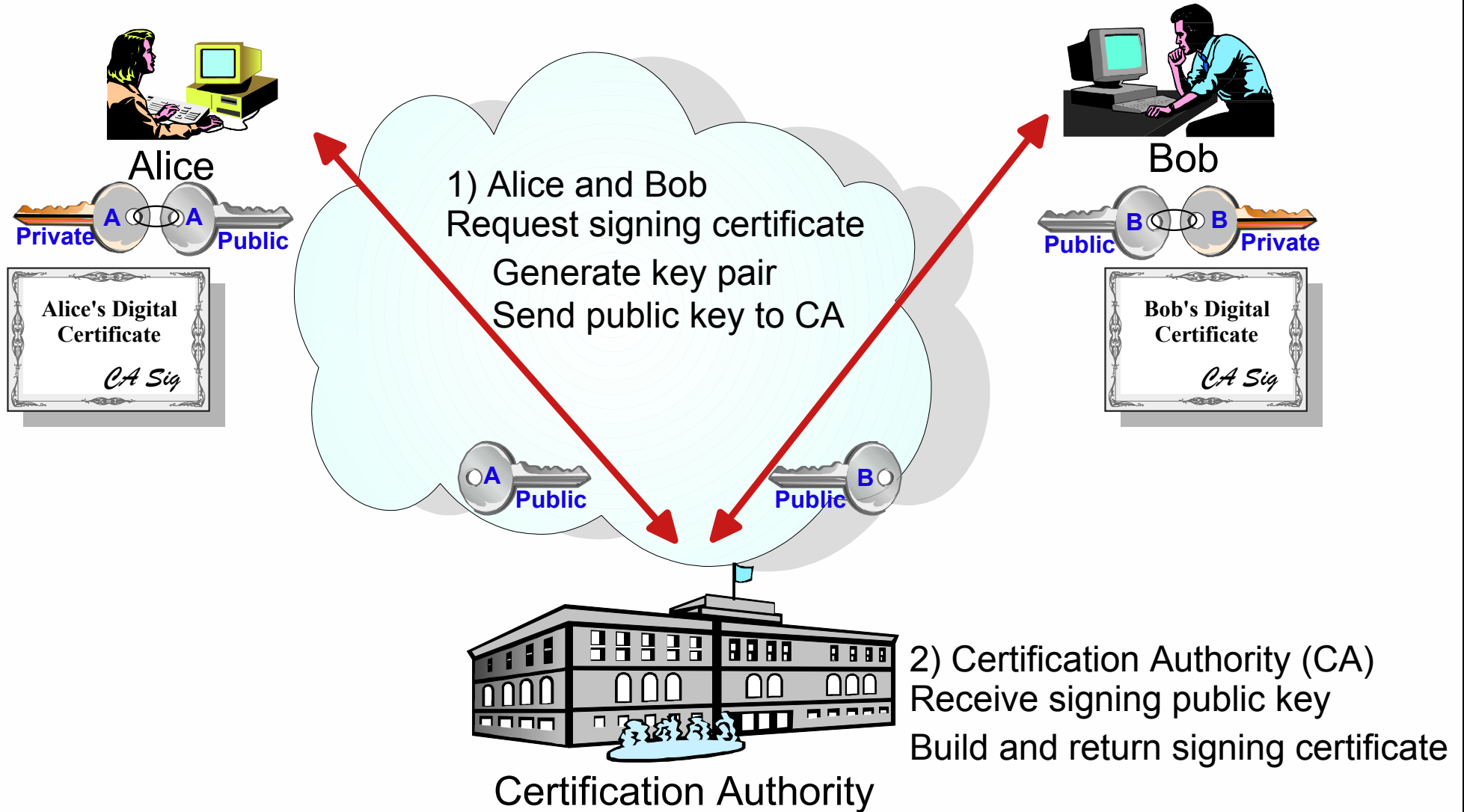
Digital Signature



Jealous Husband



Digital Certificate



Digital Certificates - More info

- **Certificate request**

- the sender's identity
 - **Distinguished Name, well known format X.500 series**
- the sender's public key
- generally money (though sometimes internal certification)

- **Certificate, X.509 standard**

- the sender's verified identity
- the sender's public key
- the Certification Authority's digital signature
- Expiry Date

- **User Certificate**

- Create using Digital Certificate Management tool of choice
- Binds an identity to a public key

- **Certification Authority**

- Trustworthy Authority
- "Well known" public key, to use for encryption of request for certificate

Distinguished Name

- Well defined format

**CN="Morag Hughson" L=Hursley O=IBM
OU="WebSphere MQ Development" C=England**

CN - Common Name

T - Title

L - Locality name

ST/SP/S - State or Province name

O - Organisation name

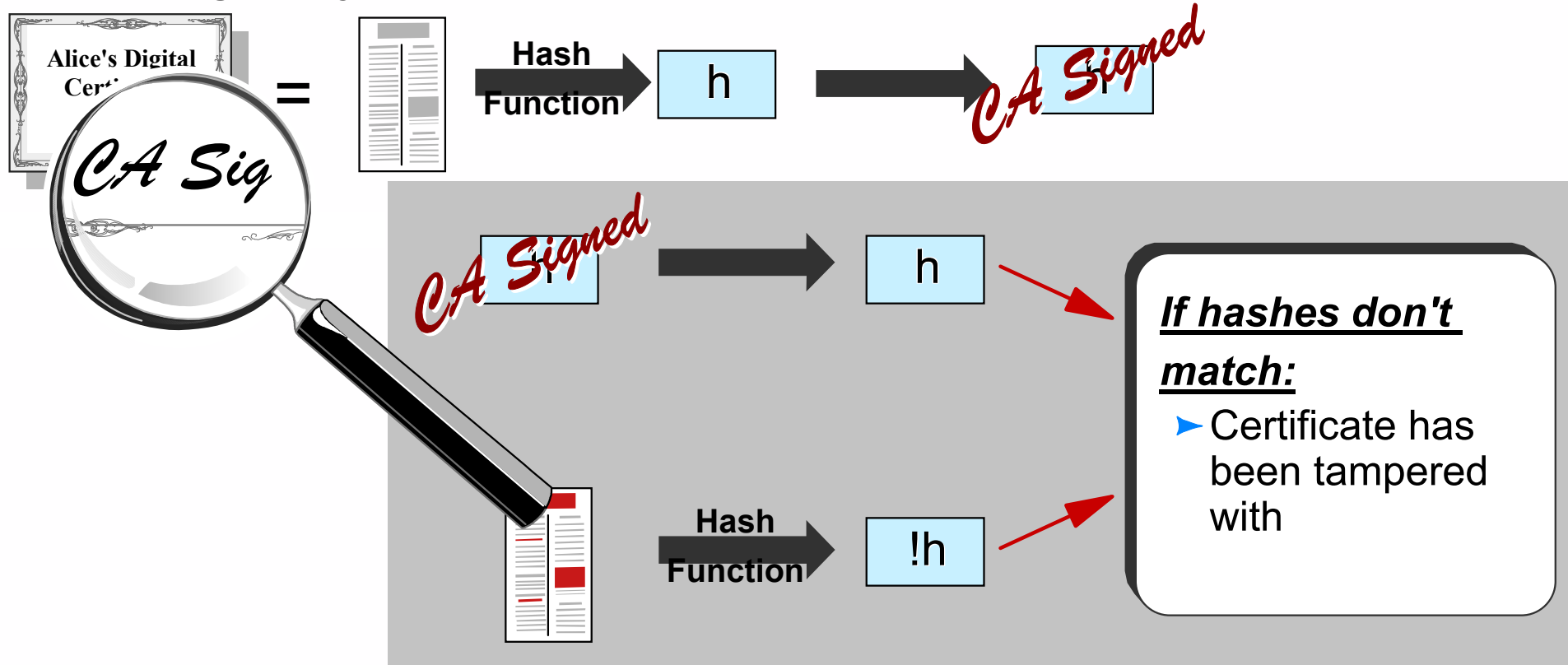
OU - Organisational Unit name

C - Country

Trusting a Digital Certificate

- **Digital Certificate = Plaintext**

- Can be subject to tampering
- Signed by CA at creation



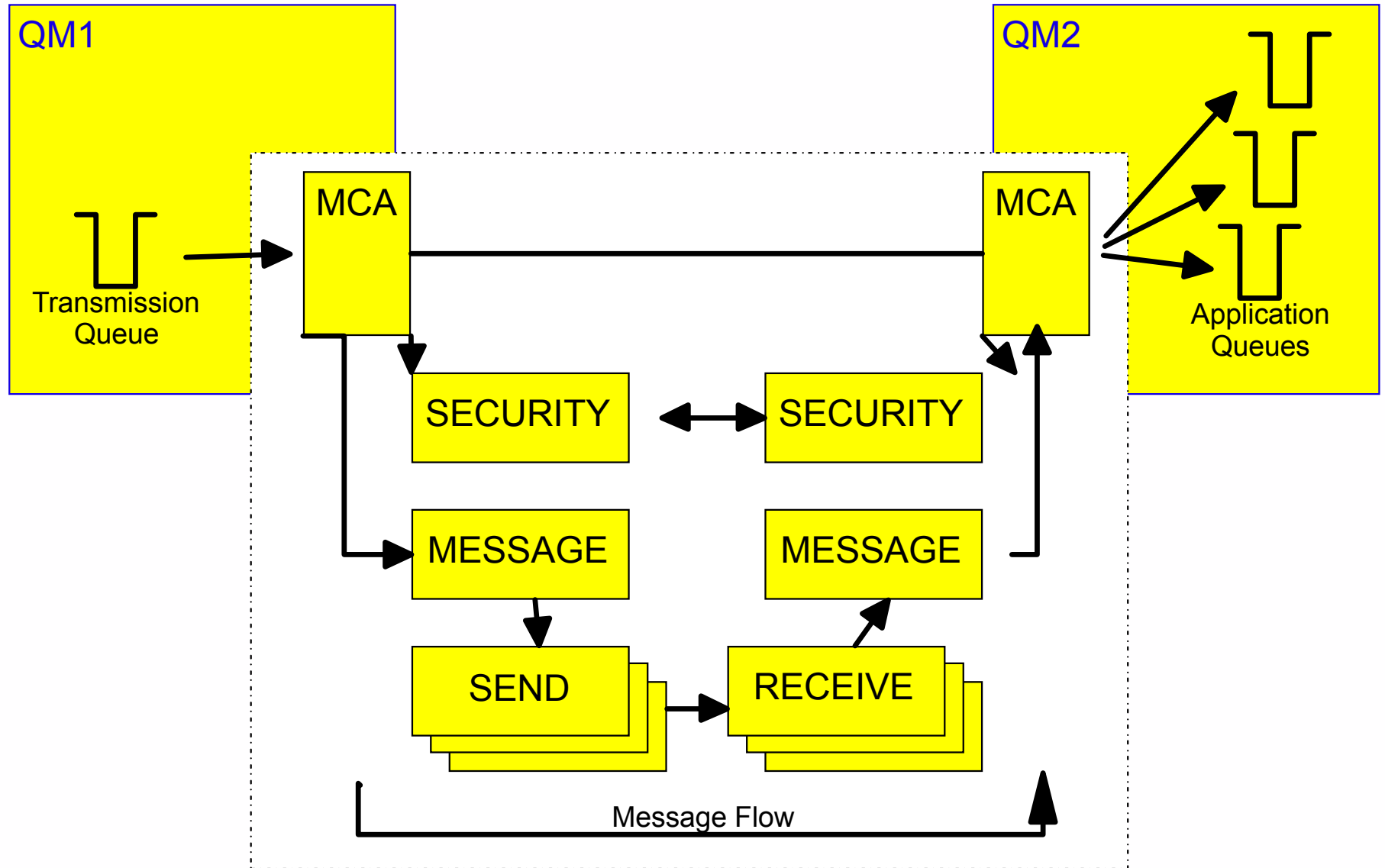
- **CA's Digital Signature**

- Allows tampering to be detected



Current MQSeries Channel Security

Current MQSeries Channel Security



MQSeries Channel Exits

- **Security Exit**
 - Called after initial channel negotiation
 - Useful for partner authentication
- **Message Exit**
 - Operates on the full message
 - Useful for encryption of the full message before transmission
- **Send/Receive Exit**
 - Operates on the transmission buffer
 - Useful for compression and decompression
 - Useful for encryption
- **Significant setup required for user exits**
- **Not 'out of the box' security**
 - Does not feel integrated with the product



Secure Sockets Layer

Secure Sockets Layer

- **Protocol to allow transmission of secure data over an insecure network**
- **Combines these techniques**
 - Symmetric / Secret Key encryption
 - Asymmetric / Public Key encryption
 - Digital Signature
 - Digital Certificates
- **to combat security problems**
 - Eavesdropping
 - Encryption techniques
 - Tampering
 - Digital Signature
 - Impersonation
 - Digital Certificates

SSL Terms

Encryption
+
Hash Function = **CipherSpec**

CipherSpec
+
Authentication/Key Exchange = **CipherSuite**

CipherSpecs

- **Encryption**

- **Block Cipher**

- **RC2**
- **DES**
- **Triple DES**

- **Stream Cipher**

- **RC4**

- **Hash Function**

- **SHA**
- **MD5**

- **CipherSpec**

- **NULL_MD5**
- **NULL_SHA**
- **RC4_MD5_EXPORT**
- **RC4_MD5_US**
- **RC4_SHA_US**
- **RC2_MD5_EXPORT**
- **DES_SHA_EXPORT**
- **RC4_56_SHA_EXPORT1024**
- **DES_SHA_EXPORT1024**
- **TRIPLE_DES_SHA_US**

Combining these techniques

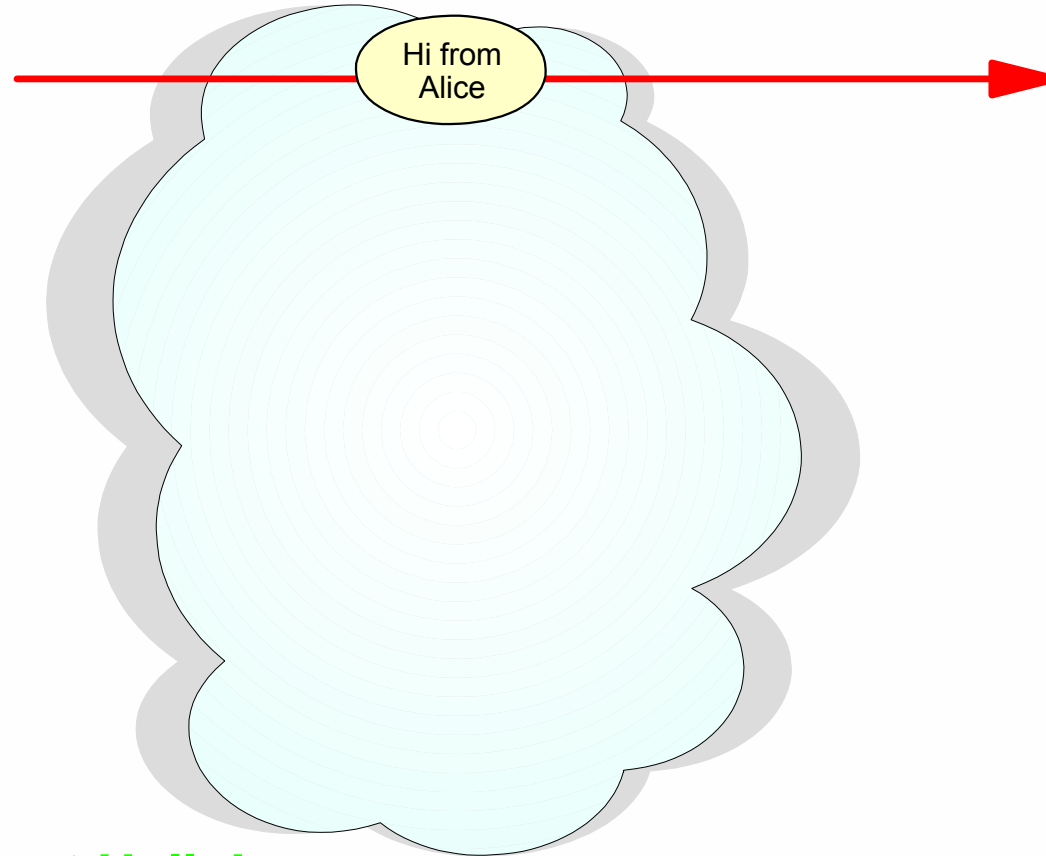
- **SSL Handshake**

- Negotiate level of SSL being used
- Exchange random numbers that are used to build one-time keys
- Negotiate cryptographic algorithms
- Authenticate parties

SSL Handshake



Alice

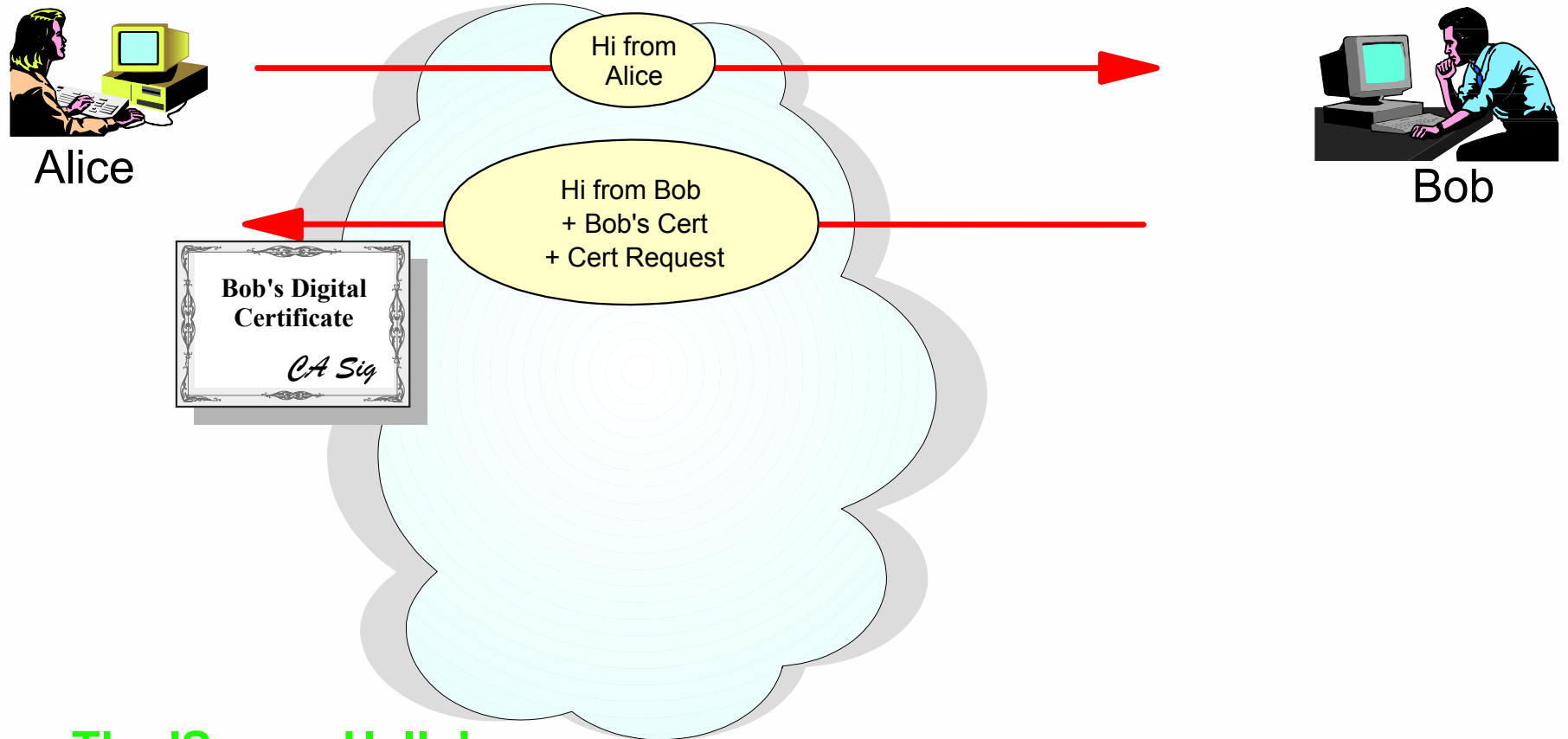


Bob

• The 'Client Hello'

- Alice sends Bob some random text
- Also sends what CipherSpecs and compression methods she can use
- Alice is considered the client since she started the handshake

SSL Handshake



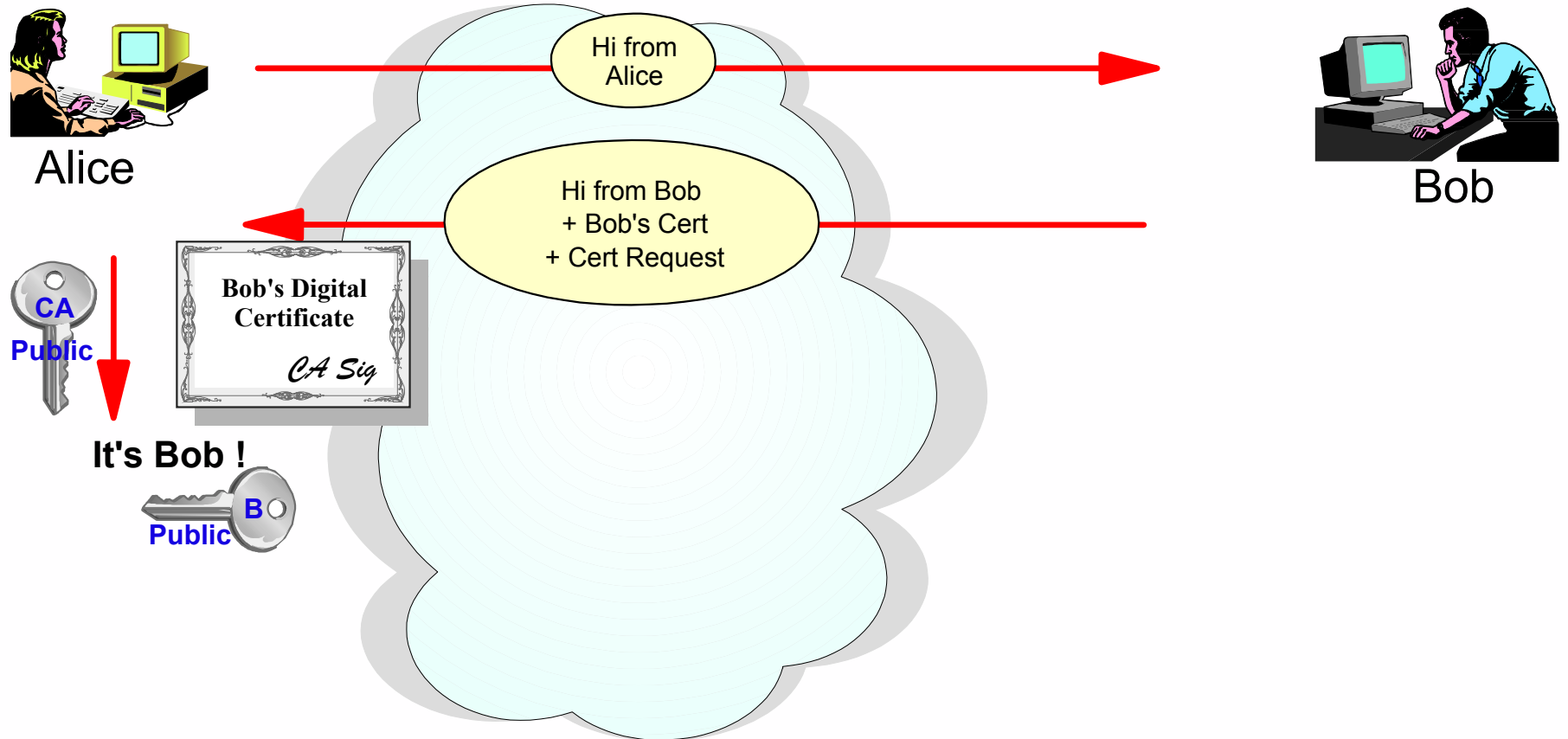
- **The 'Server Hello'**

- Bob sends Alice some random text
- Bob chooses the CipherSpec and compression method to be used, from Alice's list

- **The Server Certificate**

- **The Client Certificate Request**

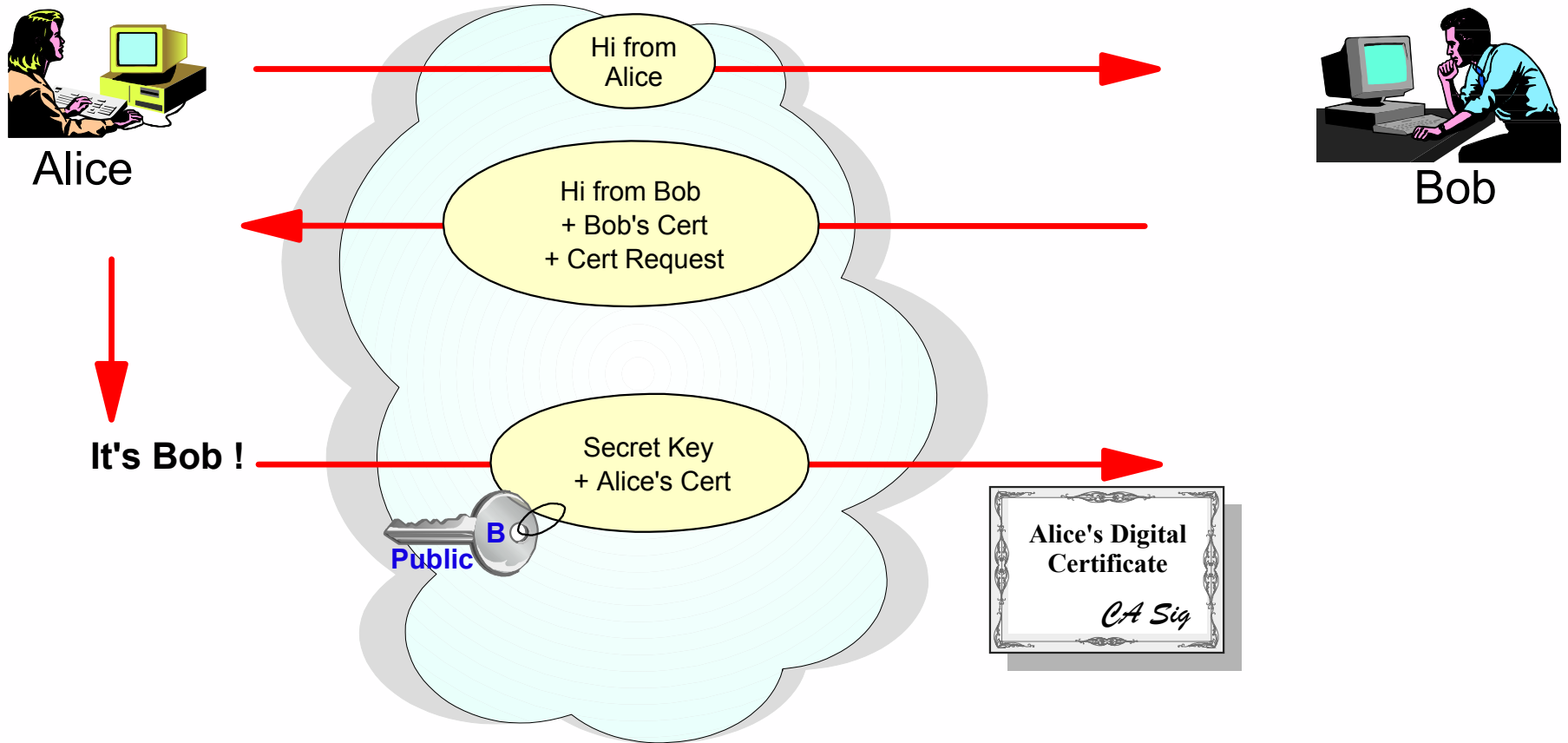
SSL Handshake



• Verify Server Certificate

- Check Validity Period
- Decrypt using CA's Public Key - verifies that CA is trusted
- Check Domain Name and/or Distinguished Name
- Also receives Bob's Public Key

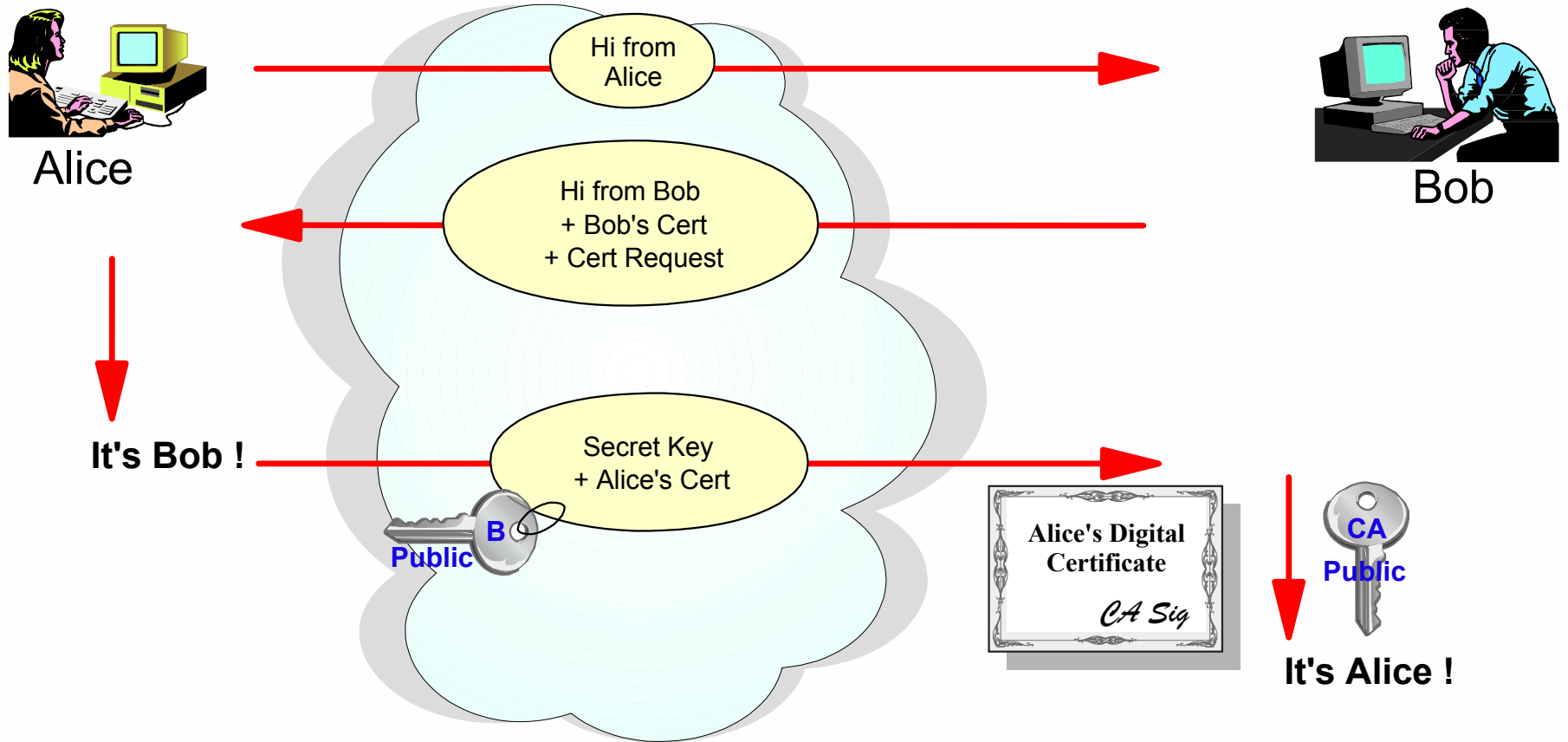
SSL Handshake



• Client Key Exchange

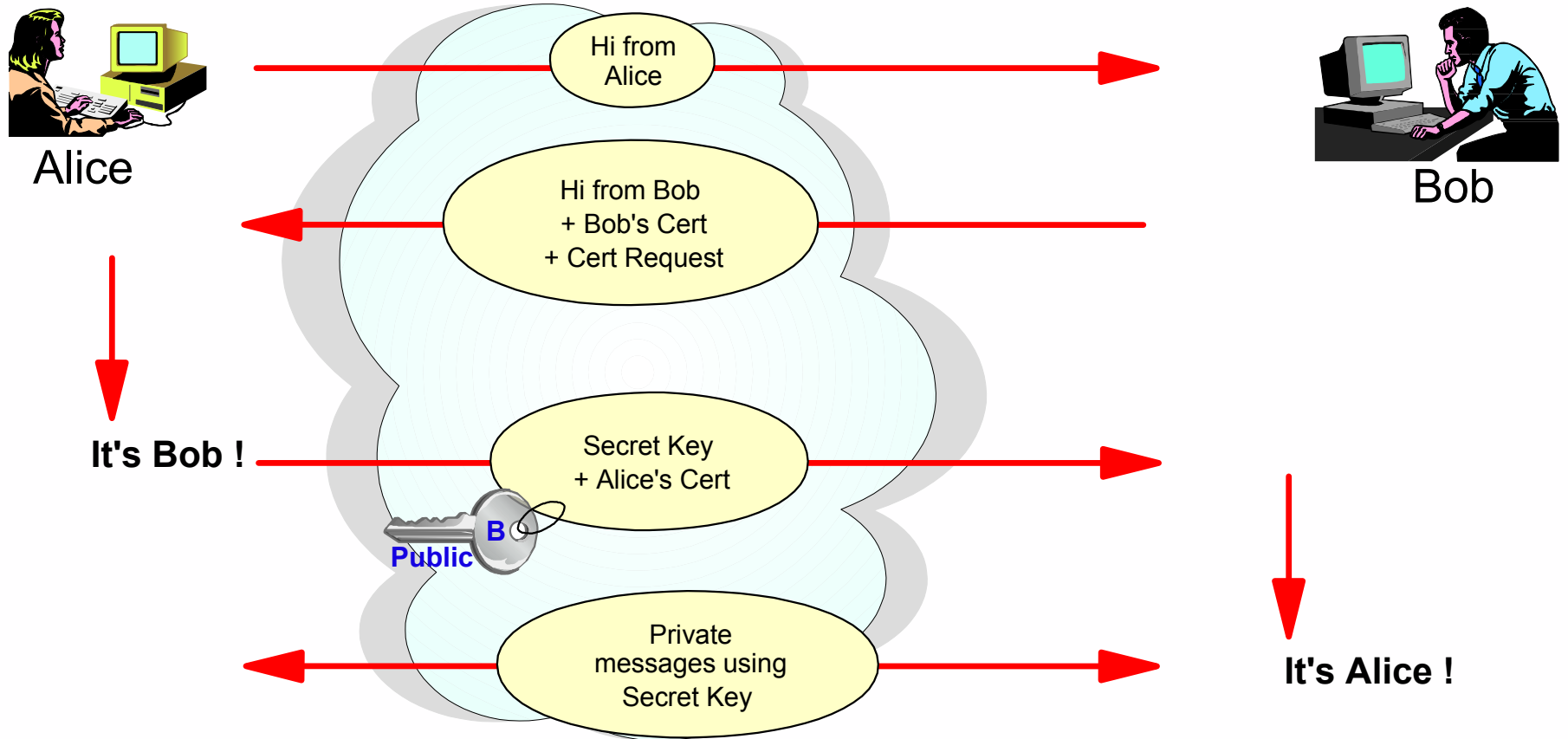
- Alice sends Bob the Secret Key to use
- This is encrypted with Bob's Public Key
- Also sends her certificate

SSL Handshake



- **Verify Client Certificate**
 - Decrypt using CA's public Key

SSL Handshake



- **Send Information using agreed Secret Key**
 - Randomly generated 1-time key
- **This is now a 'secure line'**

Certificate Revocation

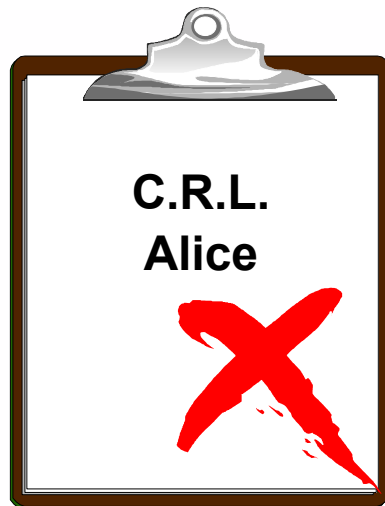
- What happens if a Certificate is no longer trusted?



Valid From 01/04/2002
Valid To 01/10/2002



- Certification Authority revokes it on a Certificate Revocation List (CRL)



Benefits of SSL

- **Provides a protocol for the function we need**
 - Encryption
 - Message Integrity Checking
 - Authentication
- **Supports a range of cryptographic algorithms**
- **Uses Public/Private Keys**
 - No key distribution problem
- **Widely accepted in the Internet community**
- **Subjected to significant testing by the hacker community**



Performance Data

Performance Costs - z/OS

- **SSL Handshake**

- Channel Startup
- Dependant on...
 - Client Authentication
 - Cryptographic Hardware

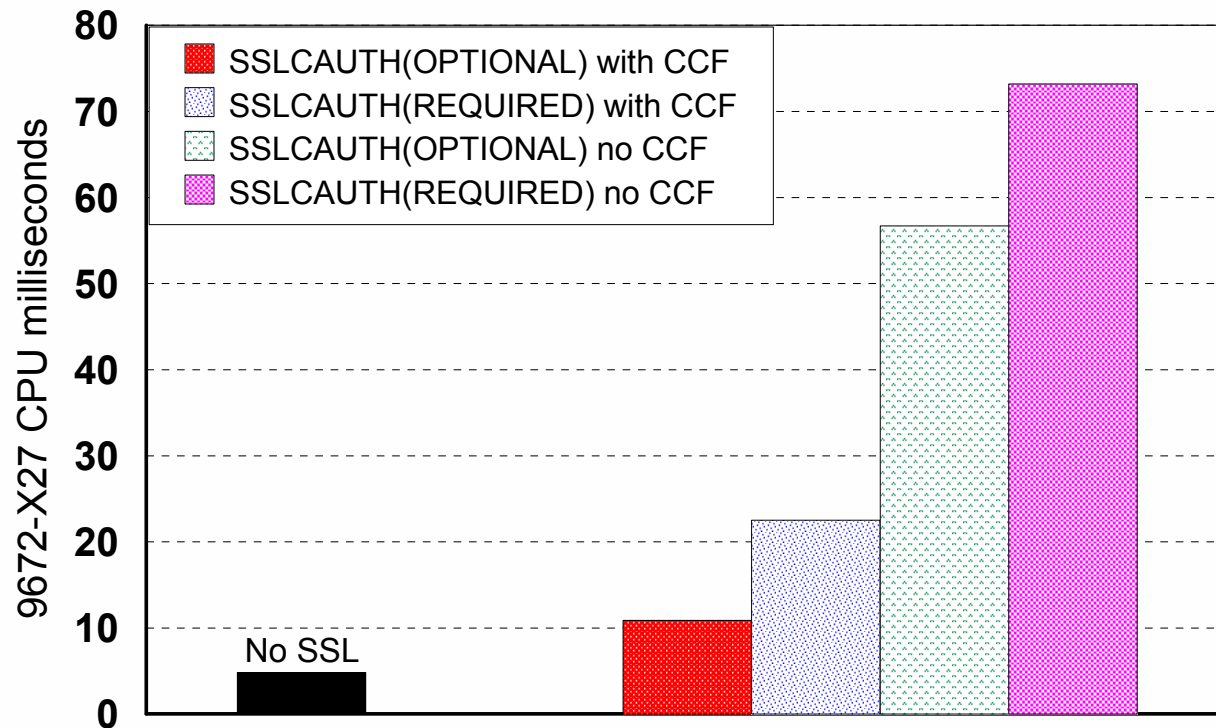
- **SSL Data Transmission**

- Moving Messages
- Dependant on...
 - Message Size
 - Different Encryption Algorithms
 - Cryptographic Hardware

- **Numbers from Performance Report**

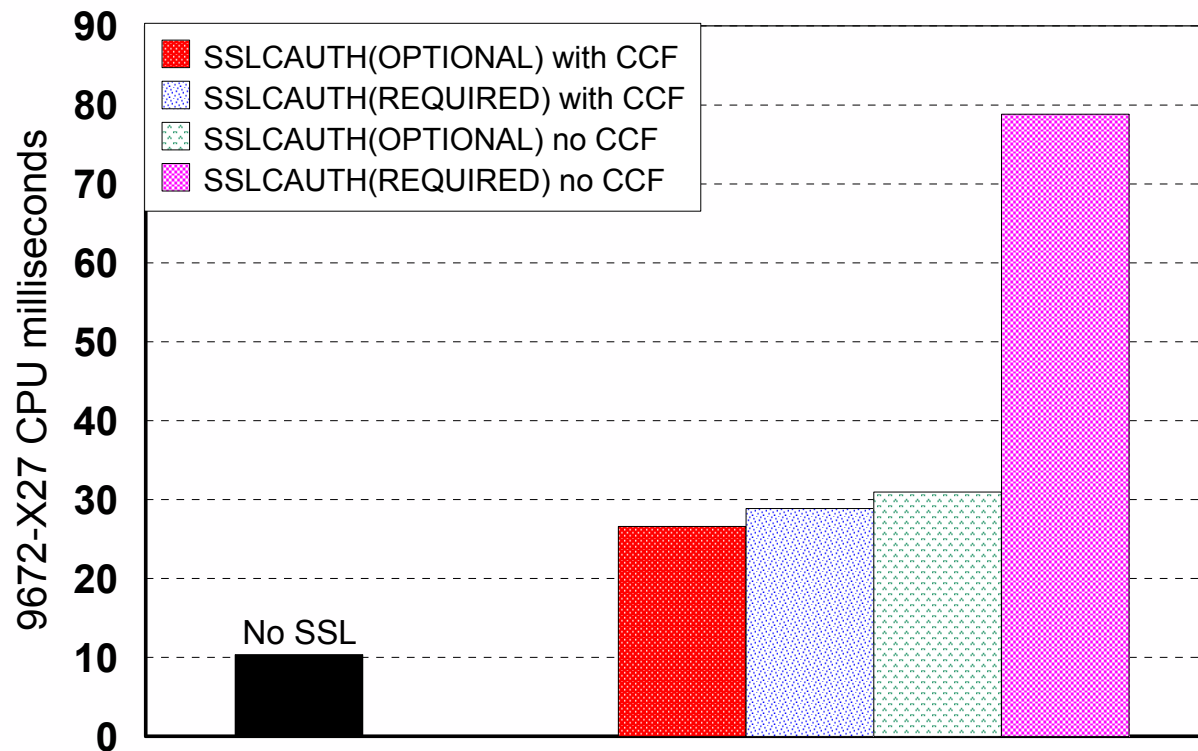
SSL Handshake - Receiver - Figures

CPU ms per Channel Start Receiver



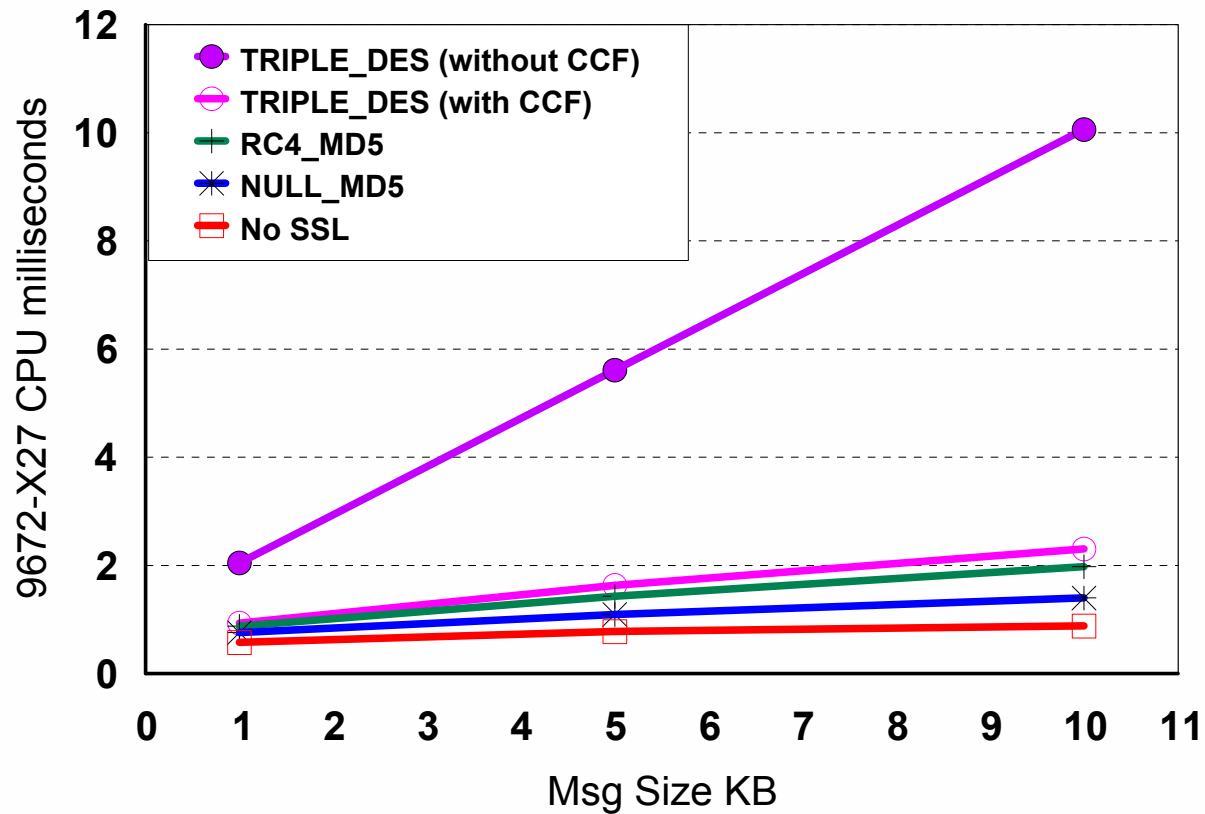
SSL Handshake - Sender - Figures

CPU ms per Channel Start Sender



SSL Data Transmission - Figures

CPU ms per Message - PUT

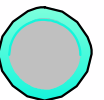


SSL Data Transmission - Performance Conclusion

Cipherspec	Action	Constant Factor	Coefficient per 1KB
NULL_MD5	Put + Send	0.15	0.045
	Receive + Get	0.12	0.49
RC4_MD5_EXPORT	Put + Send	0.15	0.1
	Receive + Get	0.2	0.105
TRIPLE_DES_SHA_US with CCF	Put + Send	0.2	0.14
	Receive + Get	0.2	0.144
TRIPLE_DES_SHA_US without CCF	Put + Send	0.5	0.89
	Receive + Get	0.5	0.9

- If want Triple DES encryption, use cryptographic hardware
- Cost is approximately linear with message size
- Example:

- For a 5KB message using RC4_MD5_EXPORT,
- additional CPU at the SENDER = $0.15 + (0.1 * 5) = 0.65$ CPU ms
- additional cost at the RECEIVER = $0.2 + (0.105 * 5) = 0.725$ CPU ms



Performance Information - Distributed (1)

- **On UNIX the SSL support is installed with WebSphere MQ**
- **Recent performance enhancements on the UNIX SSL include:**
 - main connection set up cost is private key decryption: improved algorithm from RSA called BSAFE
 - support for a number of different cryptographic hardware devices
 - CRL caching
- **Windows uses Microsoft SSL**
- **OS/400 uses OS/400 SSL**

Performance Information - Distributed (2)

- **The SSL support on the Java client is provided by the Java Secure Socket Extension (JSSE)**
- **Comparative figures between SSL and plain TCP/IP**
 - client and server on 400 MHz PC, Windows NT
 - no crypto hardware
 - RSA key exchange (1024 bit), RC4 encryption (128 bit), MD5 hashing
 - set up and 50 round trips:
 - 100 byte message: SSL 110 ms, TCP/IP 20 ms
 - 4000 byte message: SSL 230 ms, TCP/IP 30ms

Performance Costs - Summary

- **SSL Handshaking**

- Use cryptographic hardware where available
- Only use client certificates if really necessary
- Use smallest sized server key that meets security requirement

- **SSL Data Transmission**

- Only cost after SSL handshake
- Choice of CipherSpec based on Speed vs Security
- Some ciphers with benefit from cryptographic hardware



WebSphere MQ and SSL

Planning Tasks

WebSphere MQ Configuration Tasks

Security Administration Tasks

SSL functions and WebSphere MQ

- **Supported**

- **SSL V3.0**
- **Choice to authenticate client**
- **Certificate Revocation Lists on LDAP servers**

- **Not Supported**

- **List of CipherSpecs, only one must be provided**
- **SSL session reuse**



Planning Tasks

Planning tasks

- **Cryptographic hardware on z/OS and Unix**
- **Prereqs for SSL on z/OS**
 - **Hardware**
 - **Software**
- **Prereqs for SSL on distributed**

Value of Cryptographic Hardware on z/OS and Unix

- **Specialised**

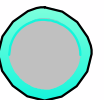
- More secure protection to maintain the secrecy of keys
- Greater transaction rates

- **Well-designed**

- Ensure the security of cryptographic keys
- Ensure the integrity of the cryptographic processes
- Limit the key-management activities to a well-defined and carefully controllable set of services

Cryptographic Hardware on z/OS

- **OS/390 Integrated Cryptographic Service Facility (ICSF) supports the following cryptographic features.**
- **Cryptographic Coprocessor Feature**
 - The Cryptographic Coprocessor Feature is available as a feature on the S/390 Enterprise Servers and S/390 Multiprise. These are complementary metal oxide semiconductor (CMOS) servers.
- **PCI Cryptographic Coprocessor Feature**
 - The PCI Cryptographic Coprocessor is a SecureWay 4758 model 2 standard PCI-bus card available as a field upgrade on the S/390 G5 Enterprise Server and on the S/390 G6 Enterprise Server.
- **Both Features supports all the cryptographic algorithms and callable services available with OS/390 ICSF.**



Cryptographic Hardware on UNIX

- **IBM FC 4963**

- Can be run on AIX 4.3.3
- PCI Cryptographic CoProcessor
- Keys and certificates stored on card
- Very secure

- **Rainbow Cryptoswift 200**

- Can be run on HP-UX 11
- No key or certificate storage
- Very fast SSL handshake

- **nCipher nFast 300**

- Can be run on Solaris 2.7
- No key or certificate storage
- Very fast SSL handshake

- **Increased coverage possible in future**

- Other platforms?
- Other cards?

Prereqs on z/OS

• Hardware

- Cryptographic Hardware is NOT a prerequisite
 - It is recommended
- If installed will be used to assist SSL processing
 - DES Encryption
 - SSL Handshake

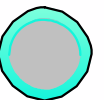
• Software

- OS/390 V2 R9 and above or z/OS
- Cryptographic Services System SSL
 - part of Cryptographic Services Base element of z/OS installed in the pdsname.SGSKLOAD PDS
- Cryptographic Services Security Level 3
 - Stronger encryption supplied in Level 3

RC4_MD5_US

RC4_SHA_US

TRIPLE_DES_SHA_US



Prereqs on Windows

- **Windows 2000 queue manager and 'C' client SSL support: integral to Windows 2000**
- **Windows NT queue manager and 'C' client: Use SSL in Internet Explorer**
 - Windows NT queue manager systems already prereq IE
 - New soft prereq on Windows NT 'C' client if want SSL
- **Windows SSL support may have a 56-bit encryption limit**
 - If want higher, we soft prereq a 128-bit upgrade
- **WebSphere MQ does not provide SSL support on Windows 98**
 - there are no plans to provide such support

Prereqs on UNIX, Java, OS/400

• UNIX

- **SSL support is installed with the WebSphere MQ queue manager and 'C' client on the following platforms:**
 - **Solaris, HP-UX 11, AIX, Linux(Intel), Linux(390)**
 - **Websphere MQ does not provide SSL support on HP-UX 11i, though this may be provided in future.**
- **Prerequisite patches on Solaris 2.7, Solaris 2.8 and HP-UX 11 (see books for Sol 2.7 and HP, GA1 README for Sol 2.8)**

• Java Client and JMS

- **prereqs a JVM at V1.4 or higher**

• OS/400 SSL support

- **OS/400 V5.1**
- **IBM Digital Certificate Manager (DCM), option 34 of OS/400**
- **IBM TCP/IP Connectivity Utilities**
- **IBM HTTP Server**
- **IBM 56-bit or 128-bit Cryptographic Access Provider**



WebSphere MQ Configuration

WebSphere MQ Configuration Tasks

- **Associating a certificate with a queue manager**
- **Associating a certificate with an WebSphere MQ client**
- **Allowing access to Certificate Revocation Lists (CRLs)**
- **Specifying cryptographic hardware (some platforms)**
- **Specifying SSL tasks (z/OS)**
- **New channel attributes**
 - **Specifying CipherSpec**
 - **Specifying permitted partners**
 - **Specifying that the partner must provide a certificate**

Queue Manager's Key Repository

- Queue Manager's own Digital Certificate

- `ibmWebSphereMQ<QMgr Name>`
(mixed case) label on z/OS
- `ibmwebspheremq<qmgr name>`
(lower case) label on UNIX and OS/400
- Selected from a GUI on Windows

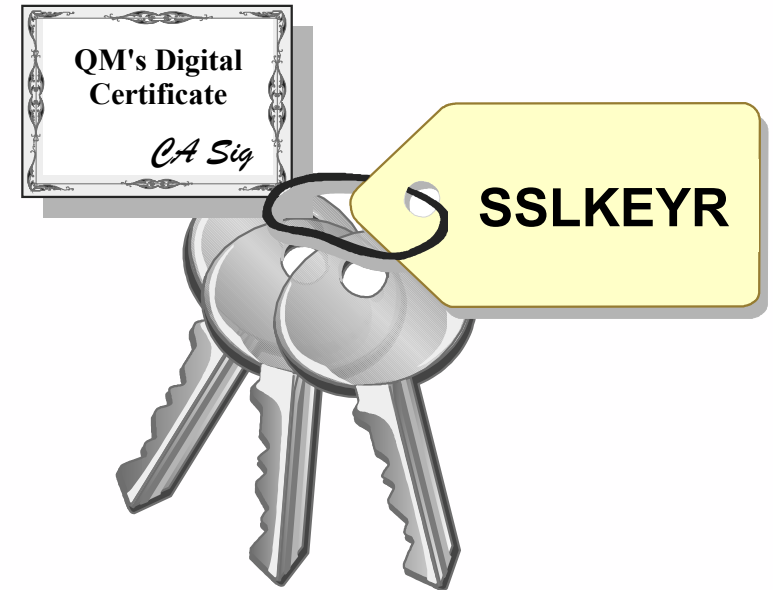
- Digital Certificates from various Certification Authorities

- On z/OS

ALTER QMGR SSLKEYR(CSQ1RING)

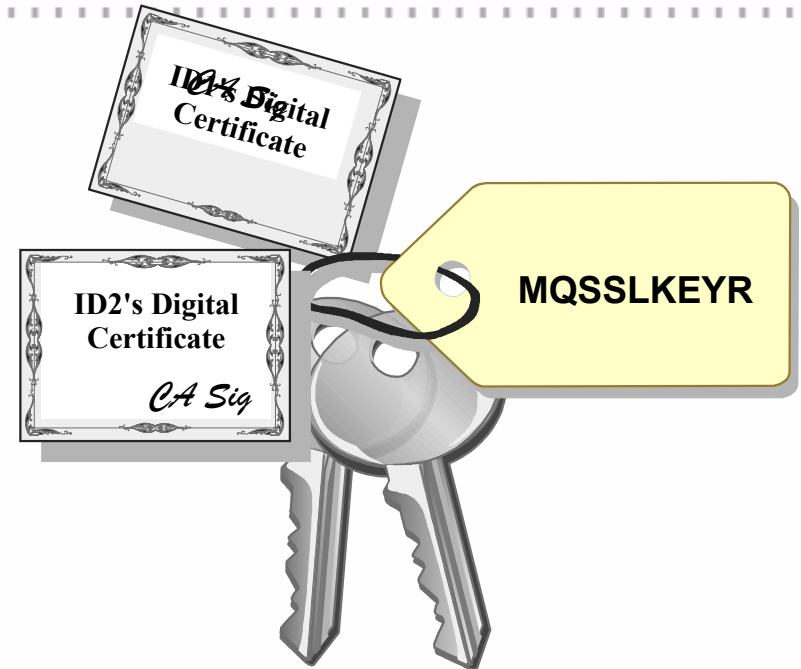
- On Unix, Windows, OS/400

ALTER QMGR SSLKEYR('var/mqm/qmgrs/QM1/ssl/key')



WebSphere MQ Client's Key Repository

- **Client's own Digital Certificate**
 - `ibmwebspheremq<logon userid>` (lower case) label on UNIX clients
 - Selected from a list on Windows
- **Digital Certificates from various Certification Authorities**



- **Specify**
 - Environment variable:

```
export MQSSLKEYR=var/mqm/ssl/key
```

- MQCONNX

SSLKeyRepository

Queue Manager: Access to Certificate Revocation Lists

- Define AUTHINFO objects

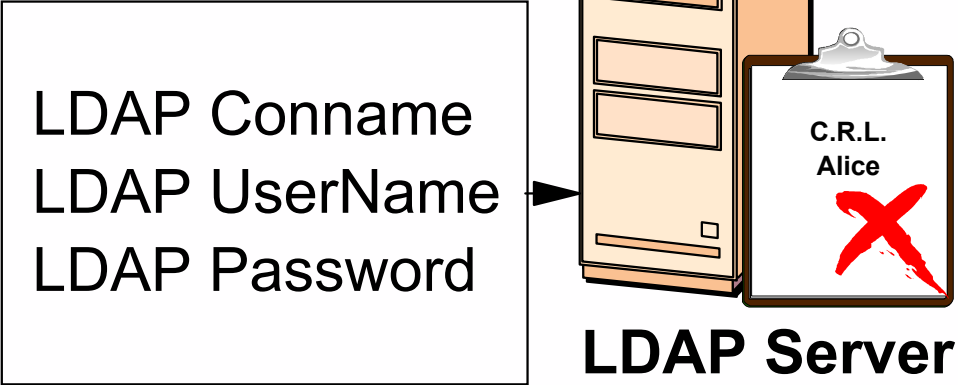
```
DEFINE AUTHINFO(LDAP1)
```

```
AUHTTYPE(CRLLDAP)
```

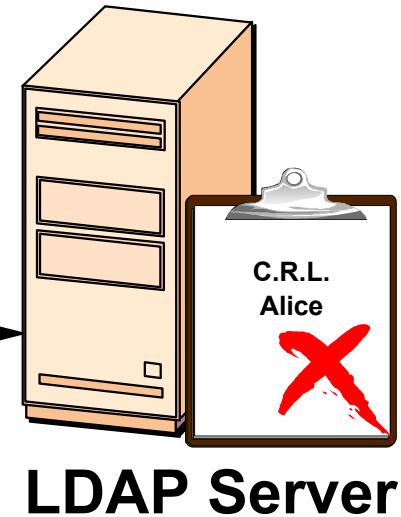
```
CONNAME(...)
```

```
LDAPUSER(...)
```

```
LDAPPWD(...)
```



LDAP Conname
LDAP UserName
LDAP Password



- Put these AUTHINFO objects into a namelist

```
DEFINE NL(LDAPNL) NAMES(LDAP1, LDAP2, ...)
```

- Associate namelist with QMGR

```
ALTER QMGR SSLCRLNL(LDAPNL)
```

WebSphere MQ client: Certificate Revocation Lists

- **Record in the client channel definition table**
 - Definitions which were current on the queue manager when the table was copied off
- **Also on MQCONNX**
- **Also, on Windows, can specify in the Active Directory**

CryptoGraphic Hardware on the UNIX platforms

- **Parameters**

- are required by the SSL support.
- required vary according to hardware used
- only apply to UNIX platforms

- **Specify**

ALTER QMGR SSLCRYP(<string>)

- <string>: hardware involved + SSL support parameters

- **On MQ client:**

- Environment variable

SET MQSSLCRYP=<string>

- MQCONNX

SSLCryptoHardware

SSL Tasks

- **MVS Tasks**

- To run SSL handshake and encryption calls
- At least 2 required to run any SSL channels

- **On z/OS**

ALTER QMGR SSLTASKS(8)

SSLCIPH

- **Only mandatory parameter on an SSL channel**
 - Without it channel is assumed not to be using SSL
- **Specify the CipherSpec to be used**
 - Both ends of the channel must specify the same CipherSpec
- **From a list of human-readable strings**
 - e.g. NULL_MD5
 - RC4_MD5_US
- **z/OS, Windows, OS/400: also SSL API numeric values**
 - Allow support of new CipherSpecs without updates to MQ Code

SSLCIPH(RC4_MD5_US)

or

SSLCIPH(04)

SSLPEER

- Specify the partner's Distinguished Name
- Can use wildcards
- Multiple Organisational Unit (OU)
 - Must be matched in order

```
SSLPEER('CN="Morag Hughson", O=IBM')
```

or

```
SSLPEER('OU=WebSphere*, O=IBM')
```

SSLCAUTH

- **Client Authentication**

- Request whether the client end is required to provide a certificate for authentication

- N.B. Client refers to SSL Client, i.e. initiating end of session

SSLCAUTH(REQUIRED)

or

SSLCAUTH(OPTIONAL)



Security Administration Tasks

Security Administration Tasks

- **Creating certificates and Certificate Requests**

- Contains a Distinguished Name
- Different tools on various platforms

- **Storing Certificates & Public Keys and Private Keys**

- Keyrings in RACF, ACF2 or TopSecret on z/OS
- Key database files on UNIX platforms and OS/400
- Certificate Stores on Windows
 - private keys are stored in the Registry on Windows

- **Managing certificates cross platform**

- Binary, must be transferred in binary
 - DER, CER, BER encodings, e.g.

PKCS #7 DER encoded X.509 certificate

- PKCS #12 DER encoded X.509 certificate (password protected).PKCS #12 files contain a personal certificate and its private key, together with the CA certificate for its signing CA(s)

- Text

- must be transferred with text conversion, e.g. ACSII -> EBCDIC
- Privacy Enhanced Mail (PEM) encoded X.509 certificate
- Base64 encoded certificate

Creating Certificates

- **Certificates Contain the Distinguished Name**

**CN="Morag Hughson" L=Hursley O=IBM
OU="WebSphere MQ Development" C=England**



- **Create internal test certificates,**

- Using RACF panels or RACDCERT commands on z/OS
 - Using iKeyMan GUI tool on Unix platforms
 - Using Microsoft MAKECERT tool on Windows
- OR**
- Using Digital Certificate Manager (DCM) on OS/400

- **Generate a certificate request**

- This request is written to a file/data set
- Send it to the Certification Authority
- This may be via their website
- Receive your signed certificate from the CA

- **Import Certificate into repository**

- labelled appropriately (z/OS, UNIX and OS/400)

- **'ibmWebSphereMQ<qmgr-name>' (z/OS)**
- **'ibmwebspheremq<lower-case-qmgr-name>' (UNIX and OS/400)**

Security Problems

Solutions

Using WebSphere MQ

- Eavesdropping

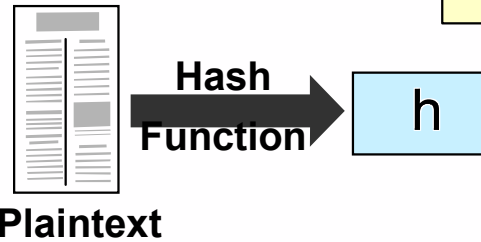
- Symmetric Key Cryptography



SSLCIPH(RC4_MD5_US)

- Tampering

- Hash Function

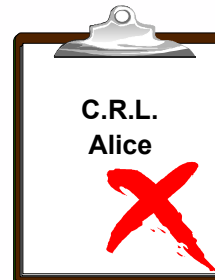
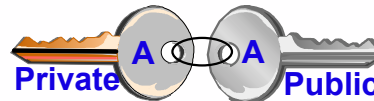


- Impersonation

- Digital Certificates
- Asymmetric Keys
- CRL checking



SSLKEYR(QM1KEYRING)
SSLPEER('O=IBM')
SSLCAUTH(REQUIRED)



SSLCRLNL(LDAPNL)