MQ and SSL

Neil Kolban
IBM Corp
kolban@us.ibm.com

October 31st 2002
Overview

- Part I – Overview of security goals and SSL
- Part II – The MQ SSL story
Security

- Goals of security
  - Confidentiality
  - Message integrity
  - Endpoint Authentication
Encryption (1)

- Encryption
  - Data confidentiality
  - Plain text vs Cipher text

Plaintext → Cyphertext → Plaintext
Encryption (2)

- Encryption
  - Data confidentiality
  - Plain text vs Cipher text

- Encryption
  - $f_E(\text{Plain}) = \text{Cipher}$
    - Example: $f_E(\text{“HEAD”}) = \text{“BQTN”}$

- Decryption
  - $f_D(\text{Cipher}) = \text{Plain}$
    - Example: $f_D(\text{“BQTN”}) = \text{“HEAD”}$

<table>
<thead>
<tr>
<th>Plain</th>
<th>Cipher</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>B</td>
<td>M</td>
</tr>
<tr>
<td>C</td>
<td>I</td>
</tr>
<tr>
<td>D</td>
<td>N</td>
</tr>
<tr>
<td>E</td>
<td>Q</td>
</tr>
<tr>
<td>F</td>
<td>C</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
</tr>
<tr>
<td>H</td>
<td>B</td>
</tr>
<tr>
<td>I</td>
<td>A</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Z</td>
<td>R</td>
</tr>
</tbody>
</table>
Cipher keys (1)
Cipher keys (2)

- Keys
  - Shared secret key
  - Symmetric cryptography
  - Common algorithms
    - DES
    - RC2
    - RC4

- Encryption
  \[ f_E(\text{Plain, Key}) = \text{Cipher} \]
  \[ f_E(\text{“HEAD”, 2}) = \text{“LPNC”} \]

- Decryption
  \[ f_D(\text{Cipher, Key}) = \text{Plain} \]
  \[ f_D(\text{“LPNC”, 2}) = \text{“HEAD”} \]

<table>
<thead>
<tr>
<th>Plain</th>
<th>Cipher K=1</th>
<th>Cipher K=2</th>
<th>Cipher K=n</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>T</td>
<td>N</td>
<td>O</td>
</tr>
<tr>
<td>B</td>
<td>M</td>
<td>T</td>
<td>W</td>
</tr>
<tr>
<td>C</td>
<td>I</td>
<td>Y</td>
<td>E</td>
</tr>
<tr>
<td>D</td>
<td>N</td>
<td>C</td>
<td>T</td>
</tr>
<tr>
<td>E</td>
<td>Q</td>
<td>P</td>
<td>S</td>
</tr>
<tr>
<td>F</td>
<td>C</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>U</td>
<td>I</td>
</tr>
<tr>
<td>H</td>
<td>B</td>
<td>L</td>
<td>N</td>
</tr>
<tr>
<td>I</td>
<td>A</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Z</td>
<td>R</td>
<td>M</td>
<td>H</td>
</tr>
</tbody>
</table>
Public Key Cryptography (1)

Plaintext → Encryption → Ciphertext → Decryption → Plaintext

Public key

Private key
Public Key Cryptography (2)

- Two keys
  - One public (known to everyone)
  - One private (known only to you)
  - Common algorithms
    - RSA
    - Diffie-Hellman
    - Asymmetric cryptography
- \( f_E(\text{Plain}, \text{Key}_{\text{public}}) = \text{Cipher} \)
- \( f_D(\text{Cipher}, \text{Key}_{\text{private}}) = \text{Plain} \)
- Keys are asymmetric
- Relatively expensive to use
Security

- Goals of security
  - Confidentiality
  - Message integrity
  - Endpoint Authentication
Message Digest (1)

- Input → arbitrary length message
- Output → fixed length string
- Attributes
  - Irreversibility
  - Collision resistance
- Other names for this
  - Hashing
  - Checksum
- Common algorithms
  - MD5
  - SHA
Message Digest (2)

- $f_H(\text{Message}) = \text{HashData}$
- $f_H(\text{Message}_1) \neq f_H(\text{Message}_2) \rightarrow \text{Message}_1 \neq \text{Message}_2$
Digital Signature (1)

- Digital Signature built from
  - Message Digest
  - Public key encryption
- Used to prove that a message has not been tampered with.
Digital Signature (2)
Digital Signature (3)
Security

- Goals of security
  - Confidentiality
  - Message integrity
  - Endpoint Authentication
Man in the middle attack
Certificate Authority
Certificates

- Issued by CA
  - VeriSign
  - Entrust
  - CyberTrust
  - etc

- Contains
  - Subject Name
  - Issuer Name
  - X.500 distinguished names

- X.509
  - Common certificate exchange format
Security

- Goals of security
  - Confidentiality
  - Message integrity
  - Endpoint Authentication

- Implement this design and you have SSL!!
Part II MQ and SSL
Data movement between queue managers

- Queue Manager to Queue Manager without SSL
- Queue Manager to Queue Manager with SSL
Adding SSL Support

![Diagram](image_url)
MQ SSL Implementations

- Supports SSL V3.0
- Implemented using:

<table>
<thead>
<tr>
<th></th>
<th>JSSE (Java Secure Socket Extension)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>SChannel</td>
</tr>
<tr>
<td>Unix</td>
<td>???</td>
</tr>
<tr>
<td>z/OS</td>
<td>System SSL</td>
</tr>
</tbody>
</table>
Channel Security

- SSL can be used across channels
- All kinds of channels supported
  - Sender
  - Receiver
  - Cluster
  - Client
  - Etc
- Specified on a per channel basis
Key questions

- Which CipherSpec shall be used?
  - Cost of security
  - Performance characteristics
- Is client authentication required?
  - Uni or bidirectional authentication
- Names of accepted peers.
  - Limit the names of channel initiators (SSL clients)
Channel definitions

- SSL either enabled or disabled by channel definition
- New parameters for channel definitions
  - Cypher spec (SSLCIPH)
  - DN’s allowed (SSLPEER)
  - Client authentication required (SSLCAUTH)
SSLCipherSpec (SSLCIPH) – Channel attribute

- Name of the Cipher specification to use
- If blank, no SSL
- Same attribute value required on both ends of the channel

<table>
<thead>
<tr>
<th>CipherSpec name</th>
<th>Hash algorithm</th>
<th>Encryption algorithm</th>
<th>Encryption bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL_MD5</td>
<td>MD5</td>
<td>None</td>
<td>0</td>
</tr>
<tr>
<td>NULL_SHA</td>
<td>SHA</td>
<td>None</td>
<td>0</td>
</tr>
<tr>
<td>RC4_MD5_EXPORT</td>
<td>MD5</td>
<td>RC4</td>
<td>0</td>
</tr>
<tr>
<td>RC4_MD5_US</td>
<td>MD5</td>
<td>RC4</td>
<td>40</td>
</tr>
<tr>
<td>RC4_SHA_US</td>
<td>SHA</td>
<td>RC4</td>
<td>128</td>
</tr>
<tr>
<td>RC2_MD5_EXPORT</td>
<td>MD5</td>
<td>RC2</td>
<td>128</td>
</tr>
<tr>
<td>DES_SHA_EXPORT</td>
<td>SHA</td>
<td>DES</td>
<td>40</td>
</tr>
<tr>
<td>RC4_56_SHA_EXPORT1024</td>
<td>SHA</td>
<td>RC4</td>
<td>56</td>
</tr>
<tr>
<td>DES_SHA_EXPORT1024</td>
<td>SHA</td>
<td>DES</td>
<td>56</td>
</tr>
<tr>
<td>TRIPLE_DES_SHA_US</td>
<td>SHA</td>
<td>3DES</td>
<td>128</td>
</tr>
<tr>
<td>TLS_RSA_WITH_AES_128_CBC_SHA</td>
<td>SHA</td>
<td>AES</td>
<td>128</td>
</tr>
<tr>
<td>TLS_RSA_WITH_AES_128_CBC_SHA</td>
<td>SHA</td>
<td>AES</td>
<td>256</td>
</tr>
</tbody>
</table>
SSLClientAuth (SSLCAUTH) - Channel attribute

- Requestor to form channel considered the SSL Client
- Defines if certificate from client is needed to form channel
- Values:
  - Required – Client authentication required
  - Optional – Client authentication optional
SSLPeerName (SSLPEER) - Channel attribute

- Distinguished names of the allowed partners
Obtaining certificates

- Certificates obtained from Commercial CA
- Certificates for test environments
  - OpenSSL
  - MakeCert
  - Java 1.4 Keytool
  - IKeyMan
Certificate Stores

- Certificates stored in *key repositories*
- Queue manager SSLKeyRepository (SSLKEYR) attributes specifies Queue Manager’s location of its own certificate
- MQ Client uses the MQSSLKEYR environment variable to specify location of certificate store
The amqmcert command

- Used to manage MQSeries certificate store
- Adds certificates to store
- Removes certificates from store
- Lists certificates in store
- Assigns certificate to queue manager
Performance

- Nothing for nothing …
- Extra CPU overhead for encrypted data
- No *official* IBM numbers yet published
- Performance expected to be equivalent to moving same quantity of data over base SSL implementation
  - Possibly better due to single handshake and reuse
  - Overhead based on ciphersuite employed
References

- MQ Security Manual
- SSL and TLS – Eric Rescorta
- Java Secure Socket Extension (JSSE) Reference Guide
- Web sites