



TCP/IP for VSE: Native SSL for VSE

Connectivity Systems
Product Development
Don Stoeber



Agenda

- The Need
 - Secure communications with applications on VSE
- The Solution
 - The SSL protocol
- The Tools
 - Standard cryptographic algorithms
- The Implementation
 - Installing SSL on VSE
- The Benefits
 - Creating secure applications for VSE



The Need

- E-Business
- E-Commerce
- Secret web sites and ports
- Viruses
- Hackers
- Denial of Service attacks
- Authentication
- Confidentiality
- Data Integrity



The Need: IP problems

- IP packets have no inherent security
 - Relatively easy to forge the addresses
 - Modify the contents
 - Replay old packets
 - Contents easy to inspect
- No guarantee that IP packets are:
 - From the claimed sender
 - Contain the original data set by sender
 - Not inspected by a third party



The Need: TCP problems

- TCP provides a reliable connection
 - Lost packets are retransmitted
 - But no:
 - Authentication
 - Confidentiality
 - Integrity
 - Repudiation



Application Message Integrity

- Messages
 - contain sensitive data
 - travel a complex path
 - must be authenticated
 - must be kept confidential
 - must not be altered
- Why not AMI for VSE ???



Why not just front end VSE ?

- Native solution is:
 - More secure
 - Efficient
 - Cheaper
 - Easy to maintain
 - Less complicated
- VSE can now do it all too...
- So, why not have secure messaging applications on VSE ?

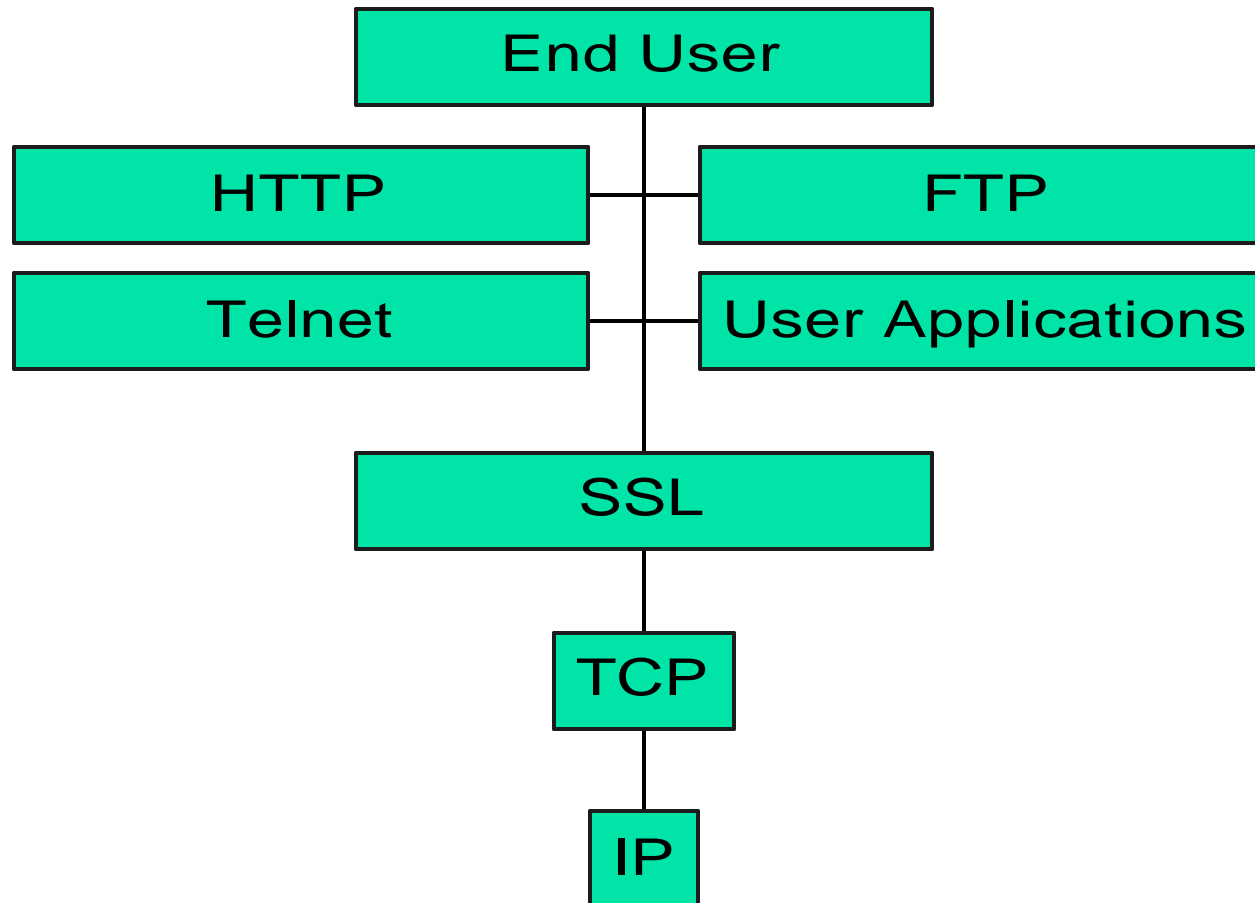


The Solution: SSL for VSE

- SSL provides secure messaging for TCP/IP applications on VSE by using:
 - Public Key Infrastructure for server and client authentication
 - Data Encryption for confidentiality
 - One-way keyed hash functions for message integrity
 - Digital Signatures for proof of authorship

The Solution: TCP with SSL Protocol

Secure



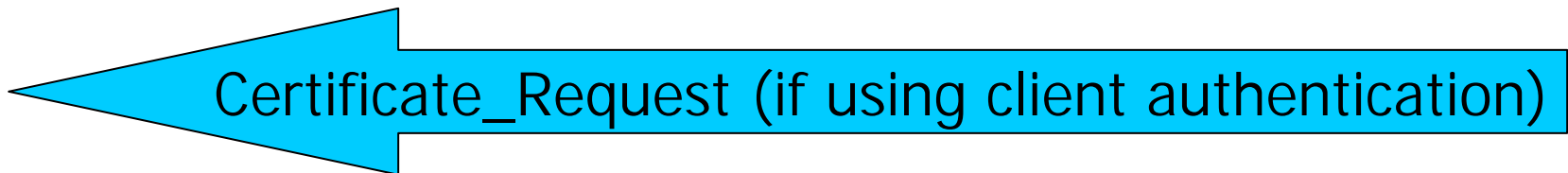
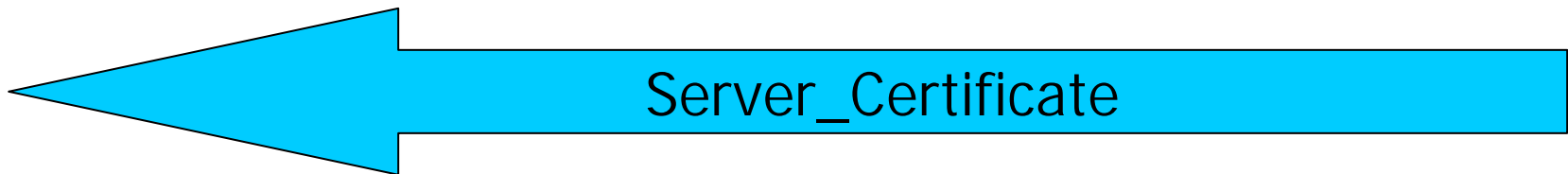
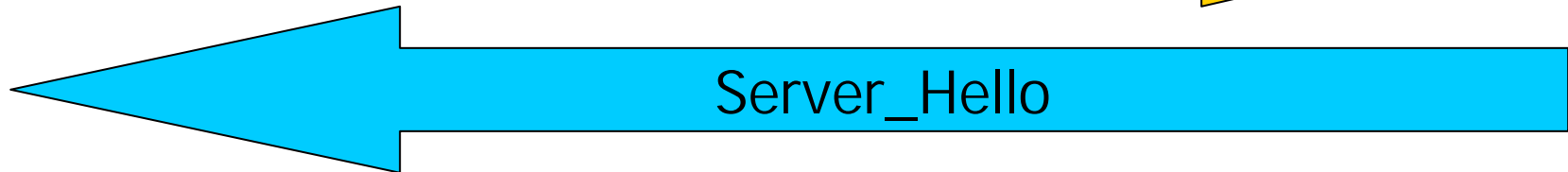
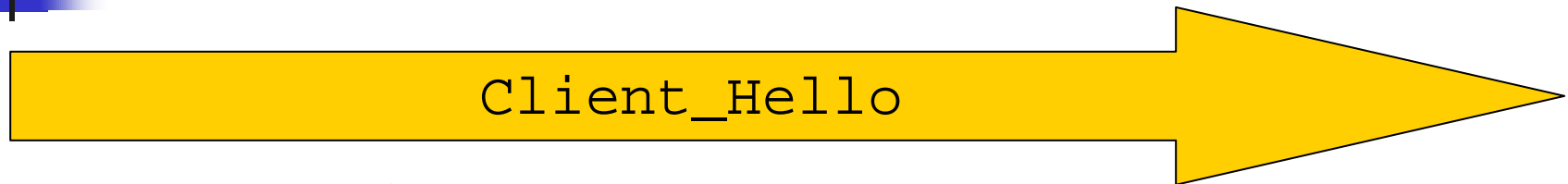


SSL Overview

- Two sockets connected
 - One must be a Server the other a Client
- Server always authenticated
- Client authentication optional
- Client and Server must:
 - Agree on cipher algorithms
 - Establish crypto keys



SSL Handshake Hello's



SSL Handshake Client Messages



`Client_Certificate(Optional)`

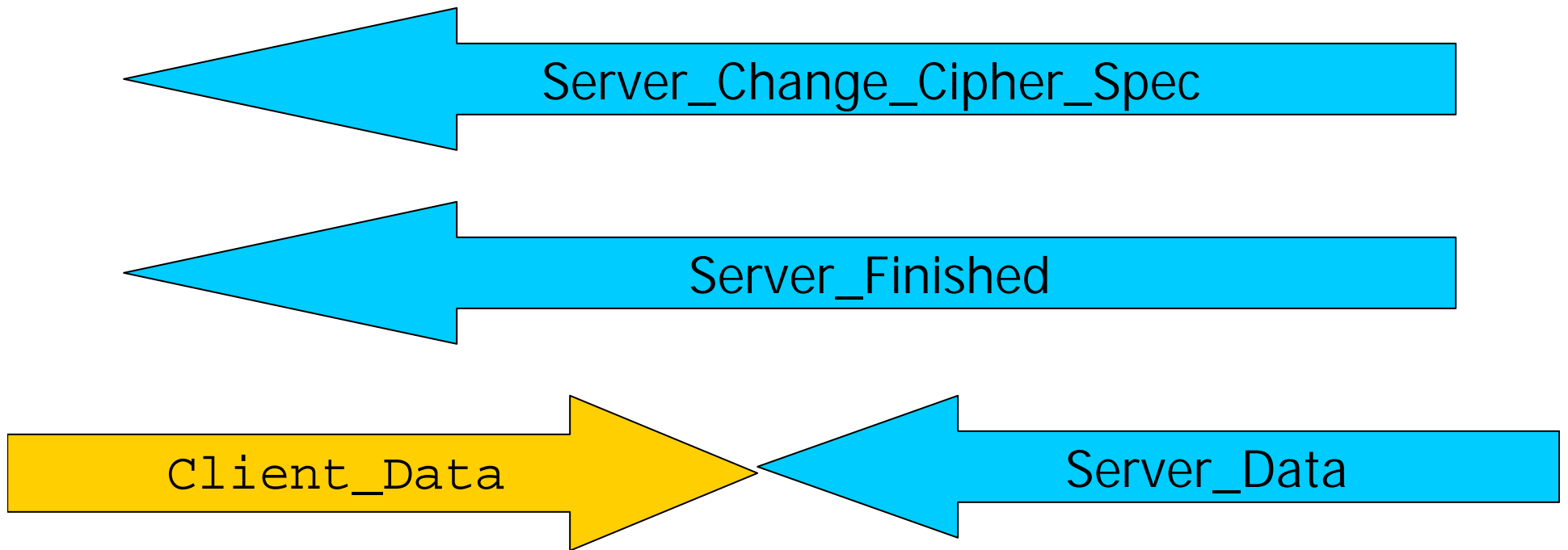
`Client_Key_Exchange`

`Client_Certificate_Verify (Optional)`

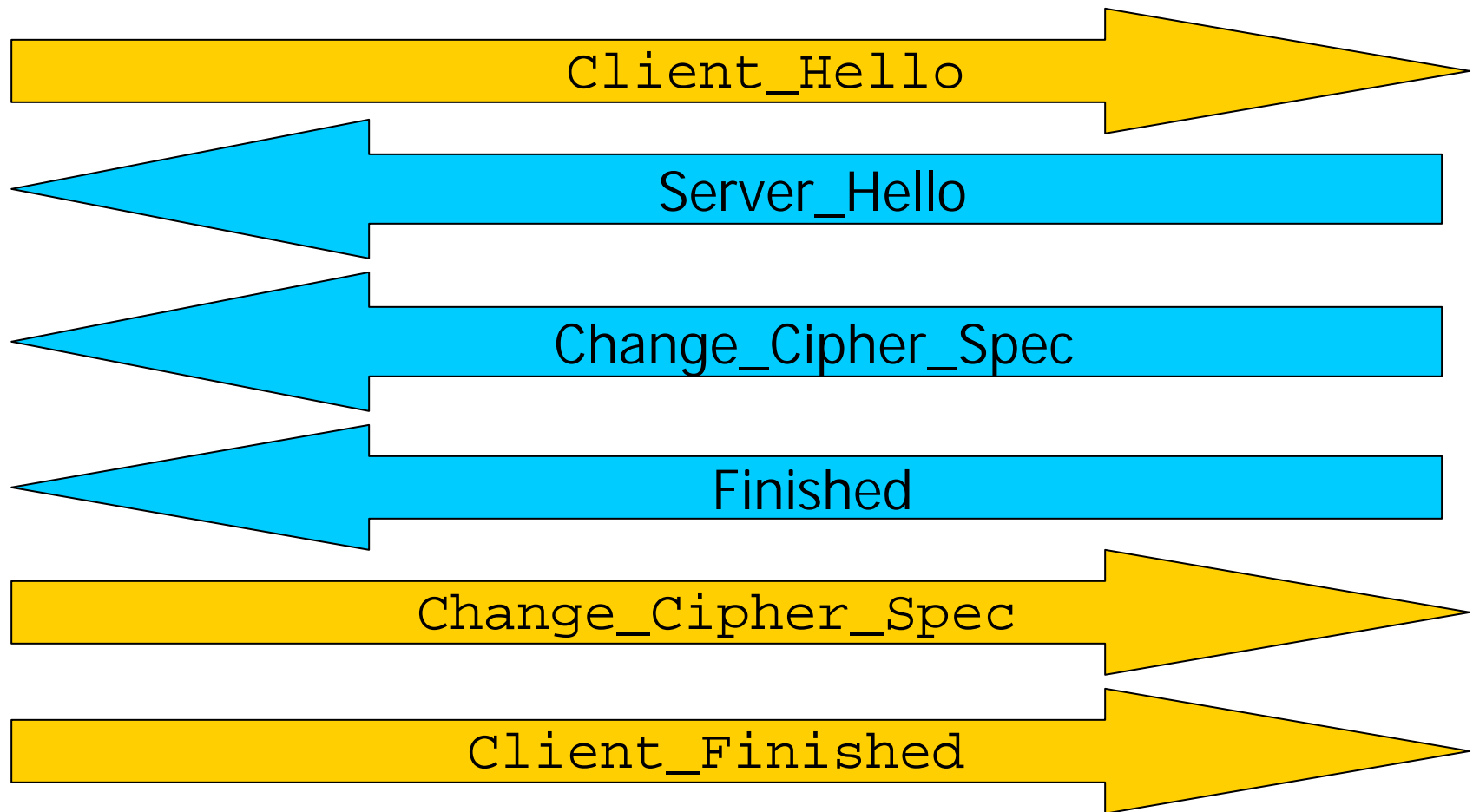
`Change_Cipher_Spec`

`Client_Finished`

SSL Handshake Server Messages



SSL Handshake Resuming a Previous Session





SSL Alerts



Close_Notify_Warning



Handshake_Failure



Bad_Certificate_Fatal



Bad_Record_MAC_Fatal



SSL Enabled Server on VSE

- Server allocates a socket binds to a port, listens, and issues a accept.
- Client connects to the VSE server and sends a "client hello".
- Server passes control to the SSL4VSE secure socket initialization routine which performs the actual SSL handshake.
- Server responds to the "client hello" by choosing the cipher algorithms that will be used during the session and sending the clients its x.509v3 PKI certificate.



SSL Enabled Server on VSE

- Key material is generated that will be used for encryption, decryption, and message authentication.
- Once the handshake is completed a secure connection is ready, and the server and client can then use secure socket read and write functions of the SSL4VSE API.



The Tools: Cryptography Algorithms

- SSL requires cryptography functions
 - X509v3 PKI certificates for identification
 - RSA for key exchange
 - DES for data encryption
 - MD5 and SHA-1 for message hashing
 - HMAC for message authentication



Crypto Toolkit for VSE

- API for cryptography standards
 - Message Digest algorithms
 - MD5 RFC1321
 - SHA-1 FIPS Pub 180-1
 - Bulk Data encryption
 - DES FIPS Pub 46-3
 - Triple DES Ansi x9.52 Triple DES
 - Message authentication
 - HMAC RFC2104
 - Digital Signatures
 - RSA PKCS#1



RSA Public Key encryption

- Used by SSL for initial key exchange and digital signatures
- Separate keys used for encrypt and decrypt
- Public key shared with others in signed certificate
- Private key used to decrypt and for creating digital signatures
- RSA patent expired in September, 2000



Implementing SSL on VSE

- Install a SSL enabled client
 - MS-IE, QWS3270 Secure, Zephyr Passport, etc
- Create a RSA private key file
- Submit a CSR request to a Certificate Authority.
 - It will contain your public key and is digitally signed with your private key



Implementing SSL on VSE

- Install the CA signed certificate
- Install the CA root certificate
- Configure the SSL daemon on VSE



Example SSL enabled client

- Microsoft Internet Explorer
 - Must be version 5 or higher
 - Under the the "Tools - Internet Options - Advanced - Security" the "Use SSL/TLS" checkbox must be checked
 - Under the the "Tools - Internet Options – Content - Certificates the CA root certificate must be installed



Creating a RSA Key file

- Based on RSA PKCS#1

```
// EXEC CIALSRVR
```

```
SETPORT 5622
```

```
/*
```

```
* * RSA private key created on PC and sent to VSE
```

```
/&
```




Creating a CSR Request

- Based on RFC2314

// EXEC CIALCREQ

Webmaster: dstoever@tcpip4vse.com

Phone: xxx-xxx-xxxx

Server: TCP/IP for VSE 1.4

Common-name: www.dstoever.com

Organization Unit: Development

Organization: Connectivity Systems

Locality: Columbus

State: Ohio

Country: US

/*



Install Certificate Authority Root Certificate

```
// EXEC CIALROOT
```

```
-----BEGIN CERTIFICATE-----
```

```
MIICpDCCAg2gAwIBAgIDPItCMA0GCSqGSIb3DQEEBBAUAMIGHMQswCQYDVQQGEwJa  
QTEiMCAGA1UECBMZRk9SIFRFU1RJTkcgUFVSUE9TRVMgT05MWTEdMBsGA1UEChMU  
VGhhd3RIIENlcnRpZmljYXRpb24xZzAVBgNVBAsTDIRFU1QgVEVTVCBURVNUMRww
```

```
...
```

```
lyqW1vNOcNo=
```

```
-----END CERTIFICATE-----
```

```
/*
```



Install CA Signed certificate

```
// EXEC CIALCERT
```

```
-----BEGIN CERTIFICATE-----
```

```
MIICpDCCAg2gAwIBAgIDPItCMA0GCSqGSIb3DQEEBBAUAMIGHMQswCQYDVQQGEwJa  
QTEiMCAGA1UECBMZrk9SIFRFU1RJTkcglUFVSUE9TRVMgT05MWTEdMBsGA1UEChMU  
VGhhd3RIIENlcnRpZmljYXRpb24xFzAVBgNVBAsTDIRFU1QgVEVTVCBURVNUMRww
```

```
...
```

```
IyqW1vNOcNo=
```

```
-----END CERTIFICATE-----
```

```
/*
```



Configure the SSL VSE server

- DEFINE TLSD,
 - ID=TLSD01, Identifier
 - PORT=7723, We listen here
 - PASSPORT=23 Pass to real daemon
 - CIPHER=0A096208 Allowed ciphers
 - CERTLIB=KEYLIB Library name
 - CERTSUB=SSLKEYS Sublibrary name
 - CERTMEM=SSL4VSE Member name
 - TYPE=1 Server application
 - MINVERS=0300 Protocol version



SSL for VSE API

- Based on IBM OS/390 SSL Programming Guide and Reference, manual number SC24-5877
 - Easy porting for OS/390 SSL applications
 - Callable from either C or BAL



IETF Standards Implemented

<http://www.ietf.org/html.charters/tls-charter.html>

- RFC2246 The TLS Protocol
 - Handshake requires server certificate from VSE
 - RSA used for generating key material
 - DES used for application data encryption
 - HMAC-SHA1 used for message authentication



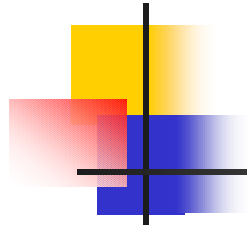
IETF Standards Implemented

- RFC1321 The MD5 Message-Digest Algorithm
- RFC2104 HMAC: Keyed hashing for message authentication
- RFC2202 Test Cases for HMAC-MD5 and HMAC-SHA-1
- RFC1113 Universal Printable Character encoding
- RFC2459 Internet x509v3 PKI certificates
- Internet draft HTTP over TLS



FIPS Standards

- PUB 46-3 Data Encryption Standard (DES)
- PUB 81 DES Modes of Operation
- Cipher Block Chaining mode
- PUB 180-1 Secure Hash Standard (SHA-1)
- <http://www-08.nist.gov/cryptval/des.htm>
- http://csrc.nist.gov/pki/nist_crypto/welcome.html



Questions ?
