HTTPS(SSL/TLS) - (SSL/TLS) is the crypto part

- HTTPS is HTTP-secure
  - SSL 2.0,3.0 (Netscape)
  - TLS 1.0,1.1,1.2 (IETF)
  - Most servers support 1.0 TLS where as most modern browsers can support 1.2 TLS
  - Note SSL 2.0 is very insecure

- Sketch of TLS
  1. Alice gets the server public key: PKS
  2. Alice and server do key exchange (DHE - 2a, RSA "Keytransport - 2b) \( k = \text{key} \times PKI \) * need two keys
  3. \( K_{enc}, K_{mac} = \text{PRG} (\text{EXT}(k)) \)
  4. \( C = \text{Enc}_{K_{enc}} (m \ldots maeK_{mac}(m) \ldots \text{pad}) \)

  Mac - always use HMAC - hash based
  - MD5, SHA1, SHA256
  - ENC - Block ciphered based- 3des (cbc mode , GCM) , AES Or stream cipher based RC4

- Cipher Suites
  - Specify what algorithms are used
  - Browsers will support many suites
  - The server will have a prioritize list of cipher suites
  - Negotiated to use highest priority of suites on the servers list which the client supports

Syntax of cypher suit
- TLS.xxx_with_yyy
  - With xxx being that key exchange(Step 2)
  - With yyy being encryption and mac from step 3

- xxx: DHE_DSS
DHE_RSA

2a
DSS = DSA = schaar
RSA = signature
RSA – RSA encryption (Key transport) 2b

yyy: AES256_cbc_sha256 from step 4
First 256 = key length
CBC = mode of operation
SHA256 - MD5, SHA1, SHA374
yyy: RC4_128_Hashfunction
TLS 1.0 There is vulnerability with the key negotiation
RC4 with in the last two weeks there is an attack, Exploits that fact that the implementation does not throw away the first X bits.

• Certificates: How do we know what the servers key is?
  Browser comes with a list of ”certificates (Root certificates)” for companies called certificate authorities (CA)
  - CA’s are trusted by the browser to correctly bind the public keys to a domain name

Ca Root (CA0) Sigk, k(Ca0, pkca0, ca: true) – self signed certificate
CaI: Intermediate (CAI) SigCAi−1, (CaI, pkcaI, ca: true)
Server (s) SigCAi, (Server.com, pk, ca: false)
- Sig (Hash(m)) where the hash is a Sha family hash
  – Signature algorithm – RSA, DSA, DSS
Other things in a certificate
  1. expiry date
  2. Revocation information

* Link info (where they can get the revocation information from)
* Two protocols for getting revocation information

  1. CRL (Certificate Revocation List )
  2. OLSP

Path length - denotes if the Cert holder 1. can it issue other certificates
  2. if so can it authorize other CA’s