COSC 301
Network Management and Security

Lecture 16: SSL/TLS and HTTPS

Haibo Zhang

Computer Science, University of Otago

Today's Focus

How to **secure** web applications?

-- Secure Socket Layer (SSL)

-- Transport Layer Security (TLS)

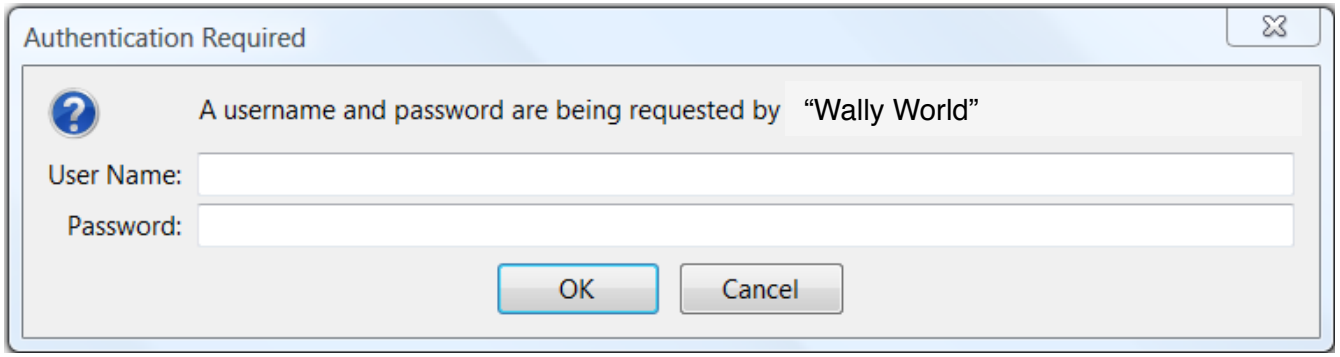
-- HTTPS

HTTP Basic Authentication (1)

- The simplest method for enforcing access control to web resources using a username and password.
- Uses static standard HTTP headers with no handshake
- **Server side:** uses a WWW-authenticate HTTP header

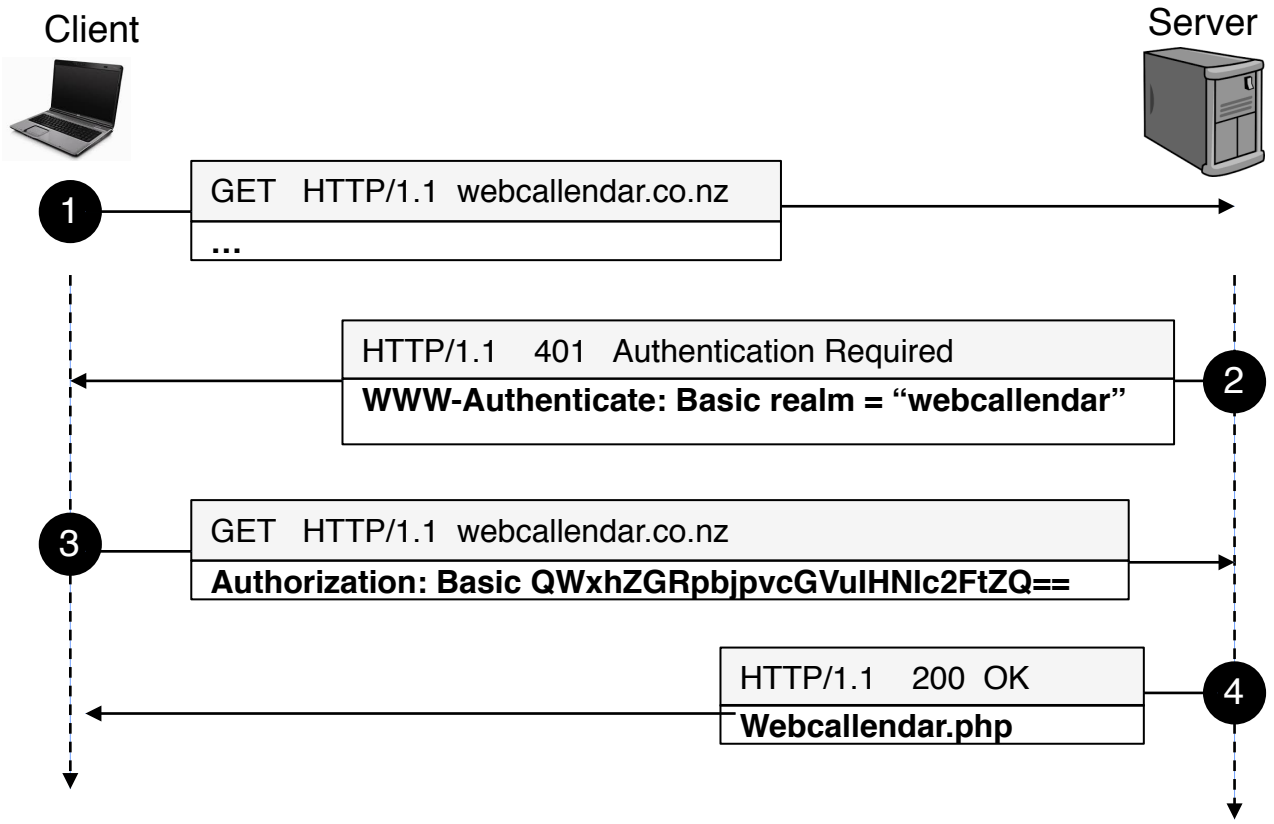
  WWW-Authenticate: Basic realm=”Wally World”
- **Client side:** uses an Authorization header

  Authorization: base64(username + “:” + password)



HTTP Basic Authentication (2)

challenge-response paradigm



The basic mechanism **does not provide enough confidentiality protection** for the transmitted credentials.

Message Digest

- Verifies that a message has not be altered
- Uses a hash function
  - MD5
  - SHA-1

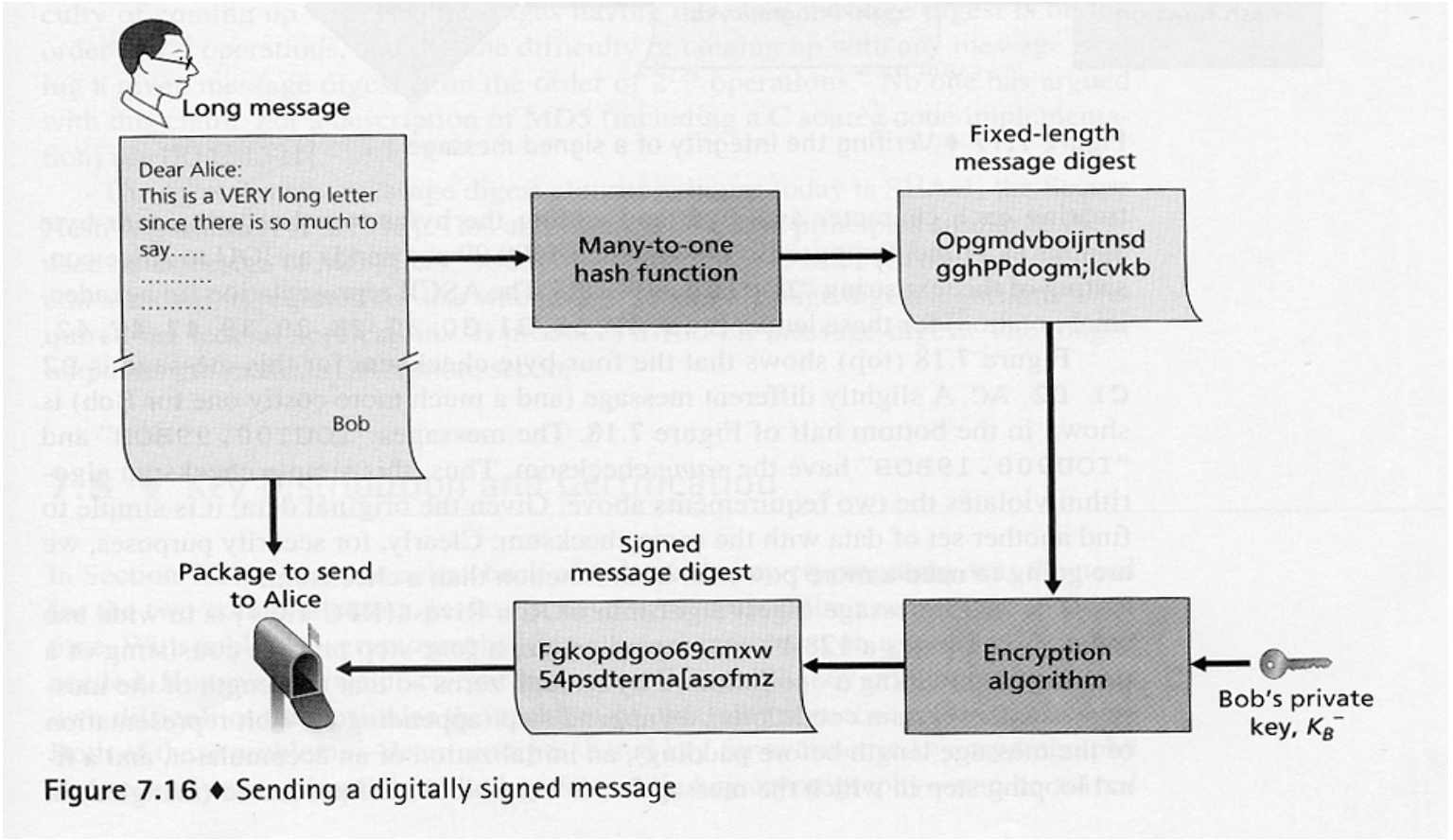


Figure 7.16 ♦ Sending a digitally signed message

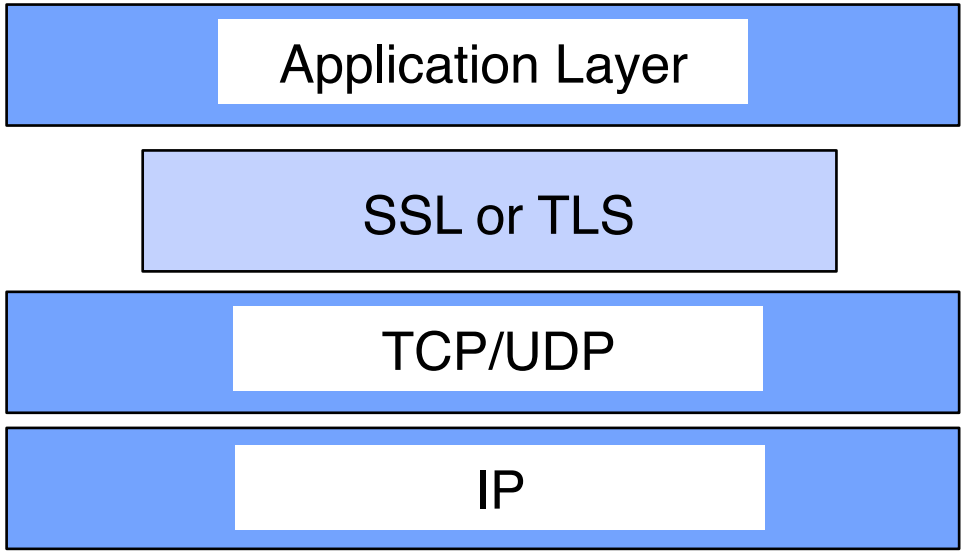
HTTP Digest Authentication (1)

- Avoids the most serious flaws of basic authentication, but not intents to be a complete solution to web security.
  - Sever challenges using a **nonce** value
  - Client responses with a digest (by default, the MD5 checksum) of the username, password, nonce value, HTTP method, and the requested URI

HA1=MD5(username:realm:password)
HA2=MD5(method:digestURI)
response=MD5(HA1:nonce:HA2)

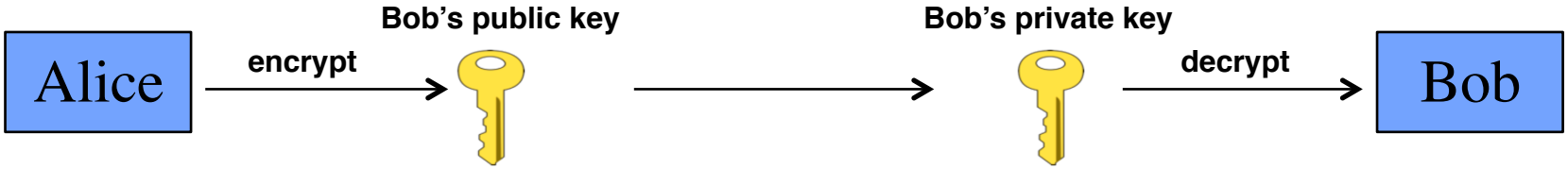
SSL and TLS

- SSL – Secure Sockets Layer protocol
- TLS – Transport Layer Security protocol

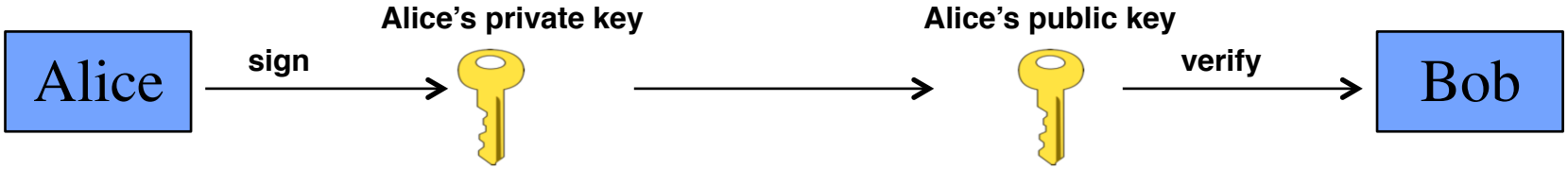


SSL and TLS

- SSL/TSL provides three essential services:
  - Encryption: ensure only authorized access



  - Authentication: verify the provided identification



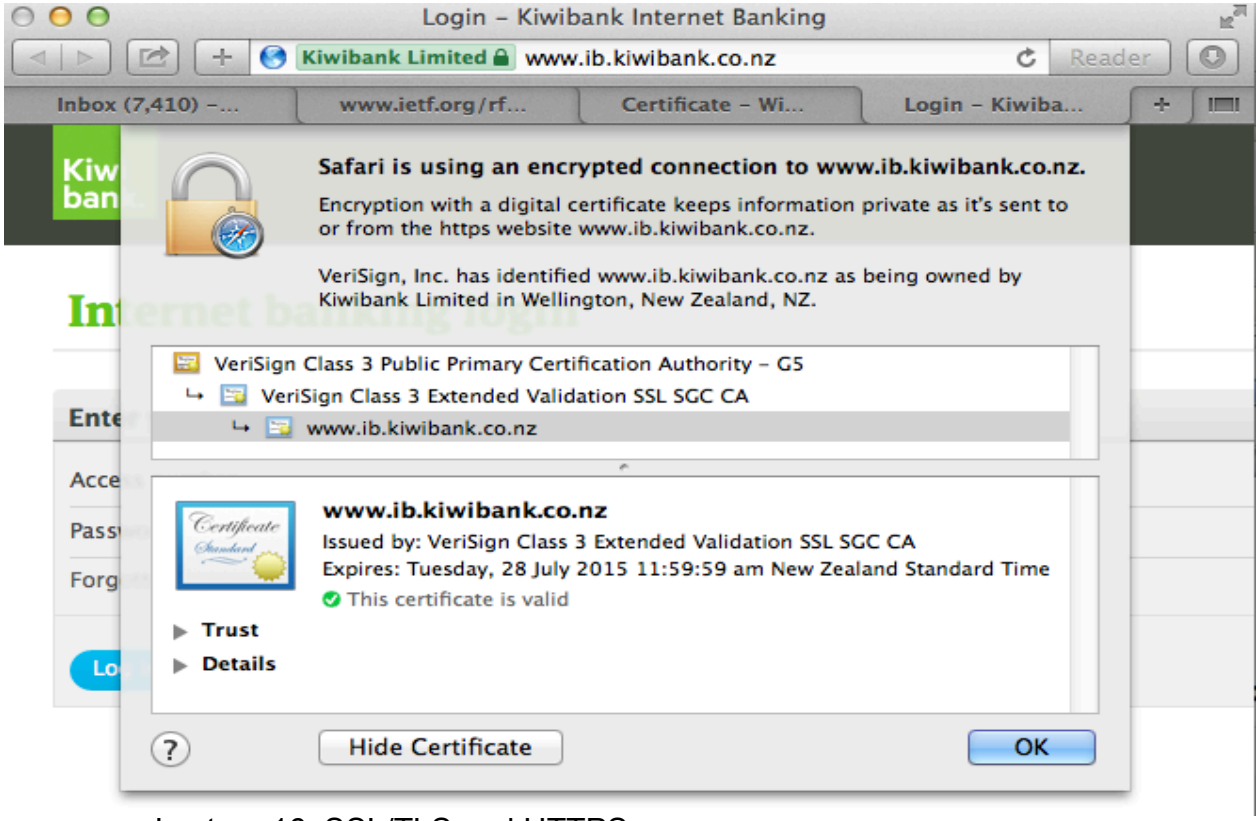
  - Data Integrity: detect message tampering and forging
    - Message digest

How to get the public key?

- Through certificate issued by Certificate Authority

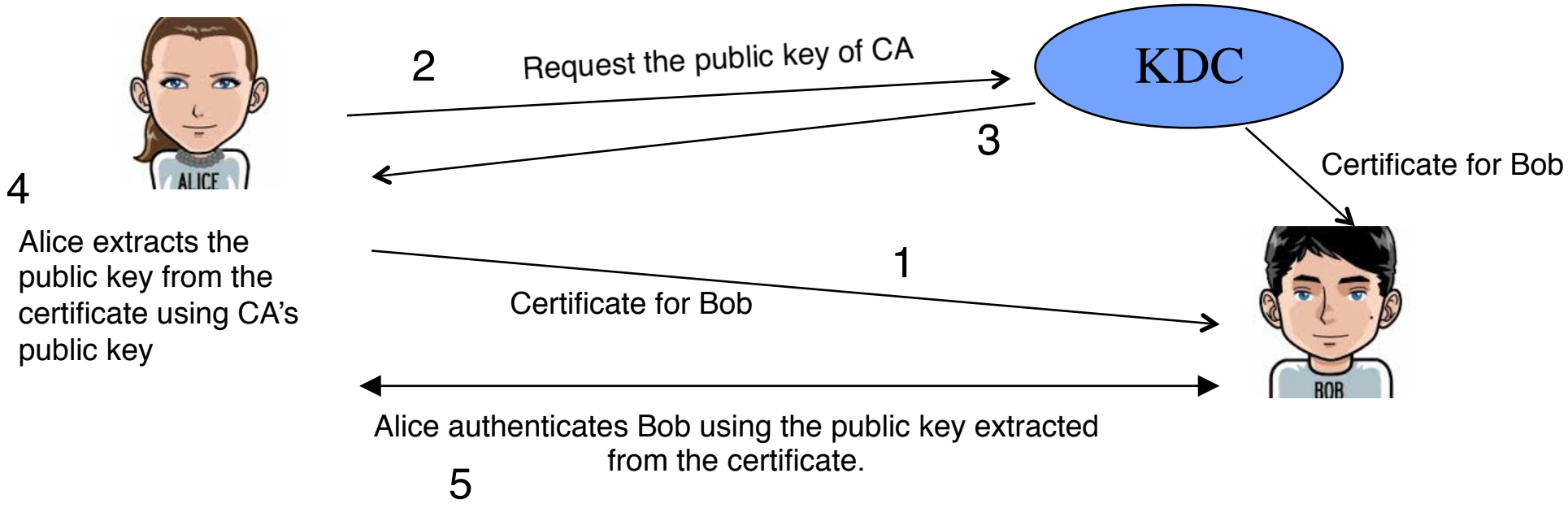
Digital Certificate (1)

- What is digital certificate?
  - Used to guarantee one is talking to the partner with whom one want to talk.
  - Generated, signed, and managed by the certificate authorities (CA)



Digital Certificate (2)

- Certificate Authority (CA)
  - Generate a certificate for Bob with its private key
  - Step 2 is usually not necessary as the public key of the CA is stored locally by the browser as trusted CA



Digital Certificate (3)

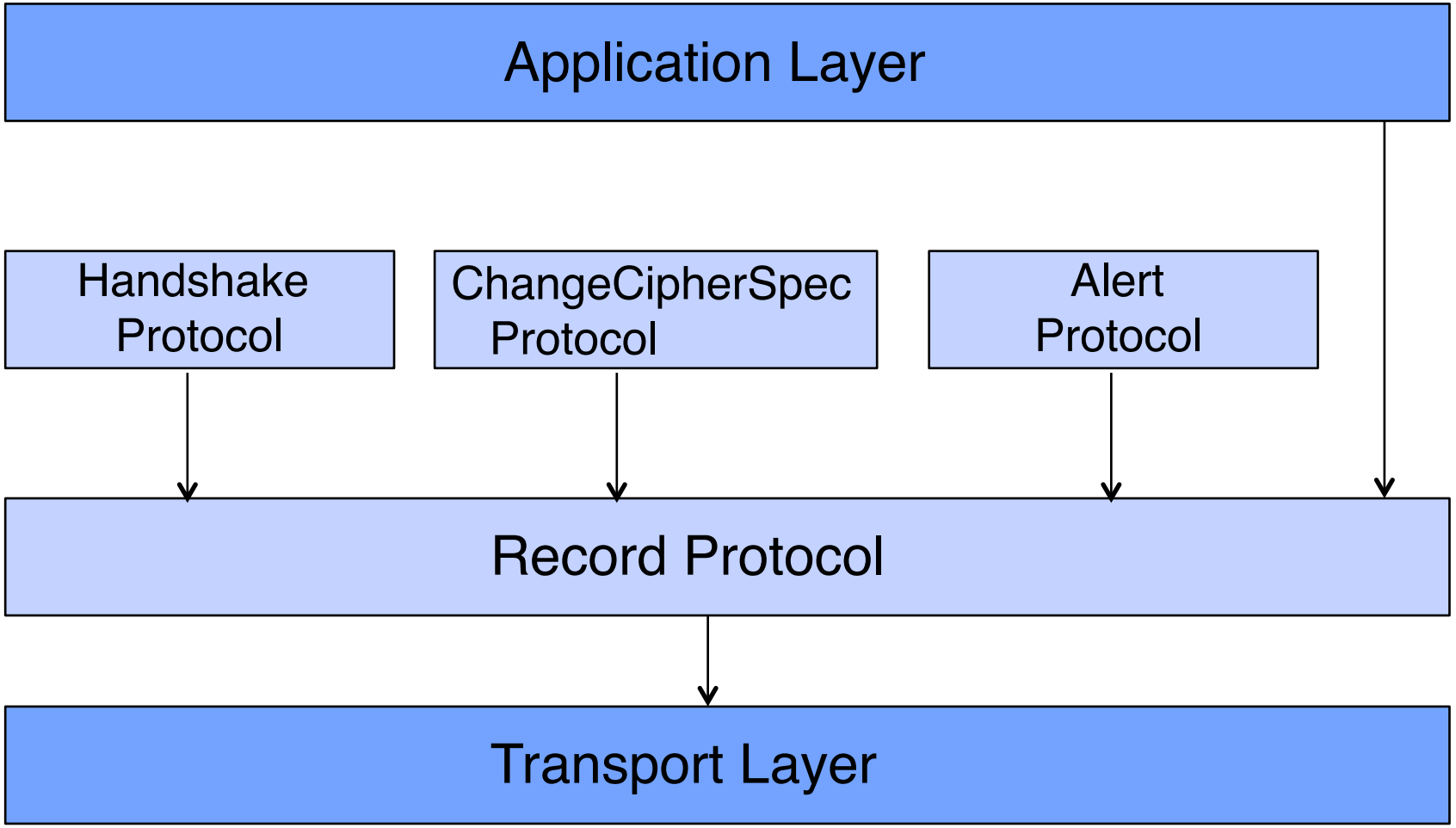
- Can you trust a certificate?
  - Check if the web address matches the address on the certificate
  - Check if the certificate is signed by a trusted certificate authority, and the date is valid
  - Check the key length, the extensions, the encryption algorithms

SSL & TLS History

- SSL v2.0 by Netscape (disable this!)
- SSL v3.0 more scrutiny, fixes attack vectors
- TLS v1.0 (= SSL v3.1) by IETF

  Few changes, but incompatible with v3.0
- TLS v1.1 clarifies, adds recommendations
- TLS v1.2 cipher updates, extensions
- TLS v1.3 released in 2018

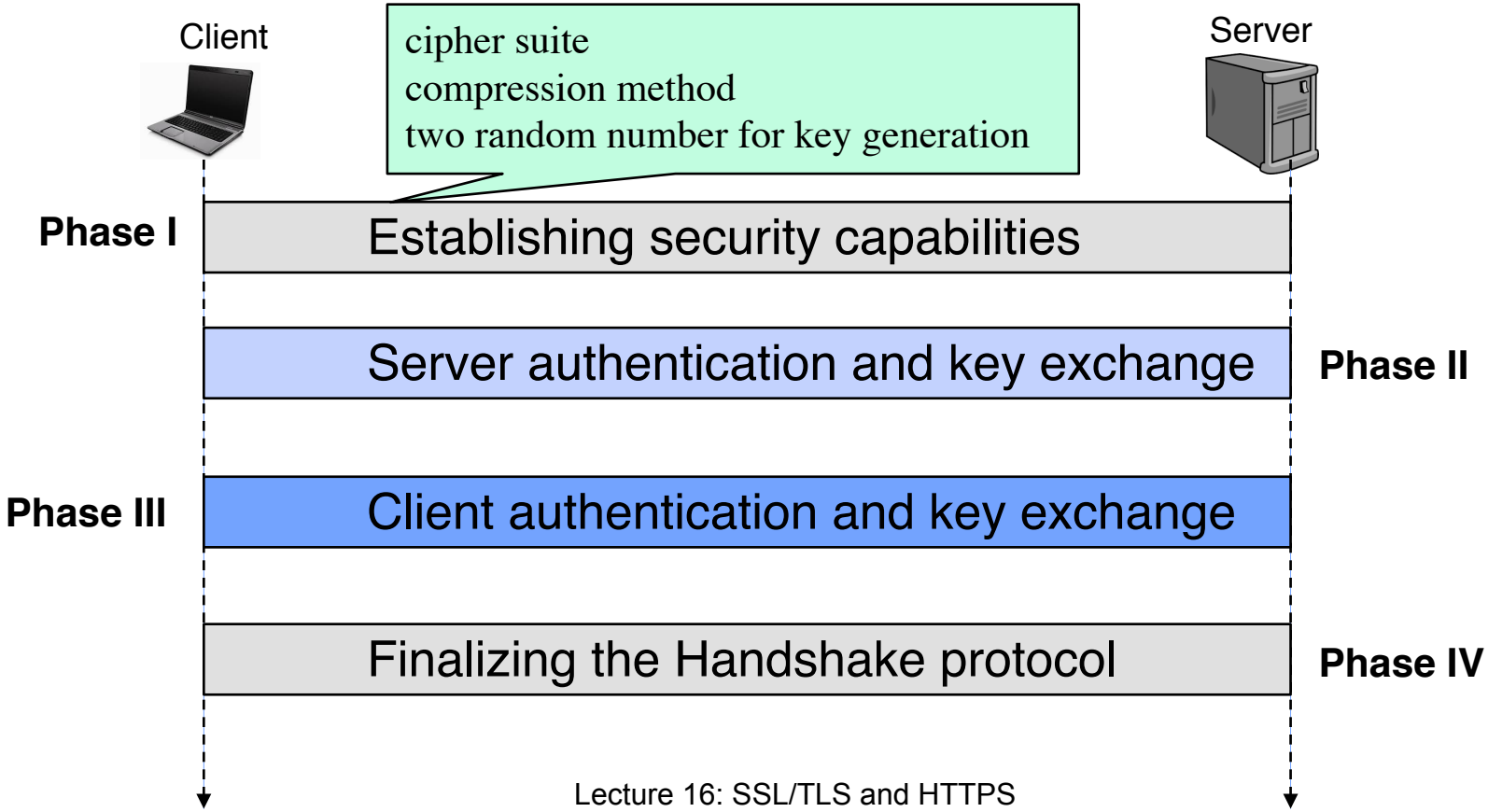
SSL/TLS Protocols



The Handshake Protocol

- Uses messages to
  - Negotiate the cipher suite
  - Authenticate sever and/or client
  - Exchange information for building cryptographic secrets

Refer to https://tls.ulfheim.net for detailed explanation.



A Closer Look at Handshake

- The client sends a "Client hello" message to the server, along with the **client's random value** and **supported cipher suites**.
- The server responds by sending a "Server hello" message to the client, along with the **server's random value**.
- The server sends its certificate to the client for authentication and may request a certificate from the client. The server sends the "Server hello done" message.
- If the server has requested a certificate from the client, the client sends it.
- The client creates a **random Pre-Master Secret** and encrypts it with the public key from the server's certificate, sending the encrypted Pre-Master Secret to the server.
- The server receives the Pre-Master Secret. The server and client each generate the **Master Secret and session keys** based on the Pre-Master Secret and the random numbers.
- The client sends "Change cipher spec" notification to server to indicate that the client will start using the new session keys for hashing and encrypting messages. Client also sends "Client finished" message.
- Server receives "Change cipher spec" and switches its record layer security state to symmetric encryption using the session keys. Server sends "Server finished" message to the client.
- Client and server can now exchange application data over the secured channel they have established. All messages sent from client to server and from server to client are encrypted using session key.

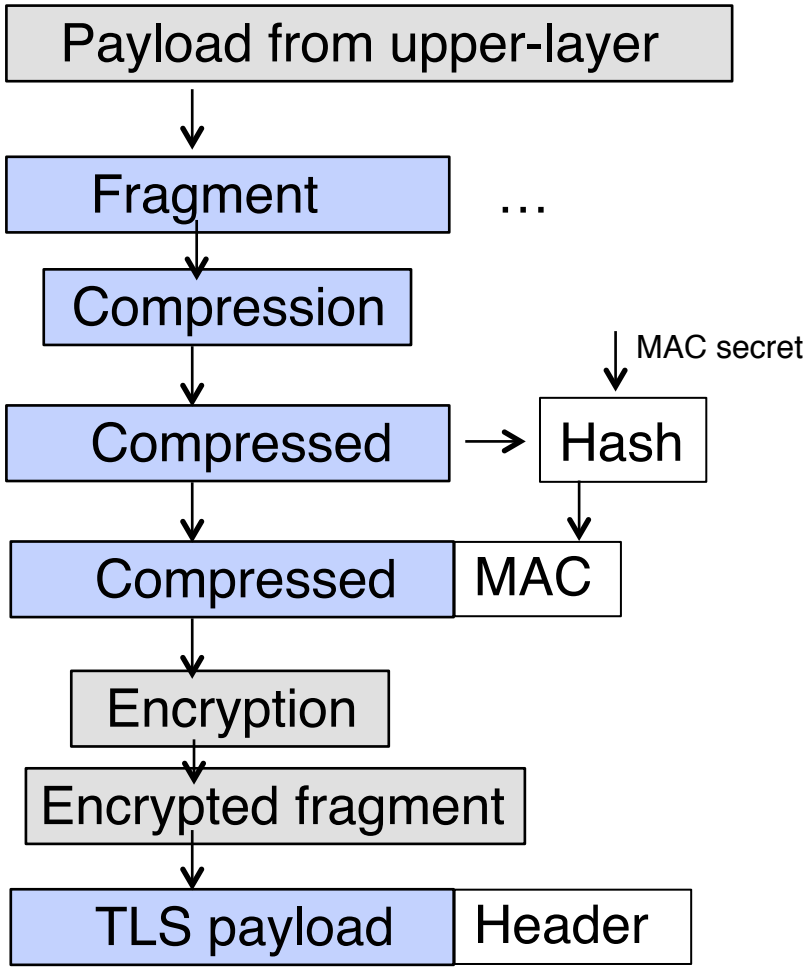
ChangeCipherSpec & Alert Protocols

- When can the two parties use these parameters or secrets?
  - Cannot use them until they have sent or received a special message -> the ChangeCipherSpec message
- How to deal with errors?
  - Uses the Alert protocol to report errors and abnormal conditions.

The Record Protocol

- Carries messages from the upper layers
  - Message fragmentation
  - Message compression (optional)
  - Message encryption

All encrypted excepted the header!



Examples to use SSL/TLS

- openssl s_client –starttls smtp –connect smtp.gmail.com:587 –crlf
- openssl s_client –connect smtp.gmail.com:465 –crlf

- See more details of STARTTLS at

https://www.fastmail.com/help/technical/ssltlsstarttls.html

HTTPS

- HTTP over TLS or HTTP over SSL
  - Layering HTTP on top of the SSL or TLS
  - Adding security capabilities of SSL/TLS to standard HTTP

- Difference from HTTP
  - HTTP URLs begin with "http://" and use port 80 by default
  - HTTPS URLs begin with "https://" and use port 443 by default

https://www.ib.kiwibank.co.nz

HTTPS

- How do we know a website uses encryption?
  - A closed padlock

  - A URL that begins with “https:” rather than “http:”

Summary

- HTTP authentication
  - Basic authentication
  - Digest authentication
- Digital certificate
- SSL/TLS
  - Protocols
  - HTTPS