THE UNIVERSITY OF MELBOURNE

COMP30023 – Computer Systems

2022 – Semester 1 - Week 4 – Lecture 2

Secure communication

Olya Ohrimenko

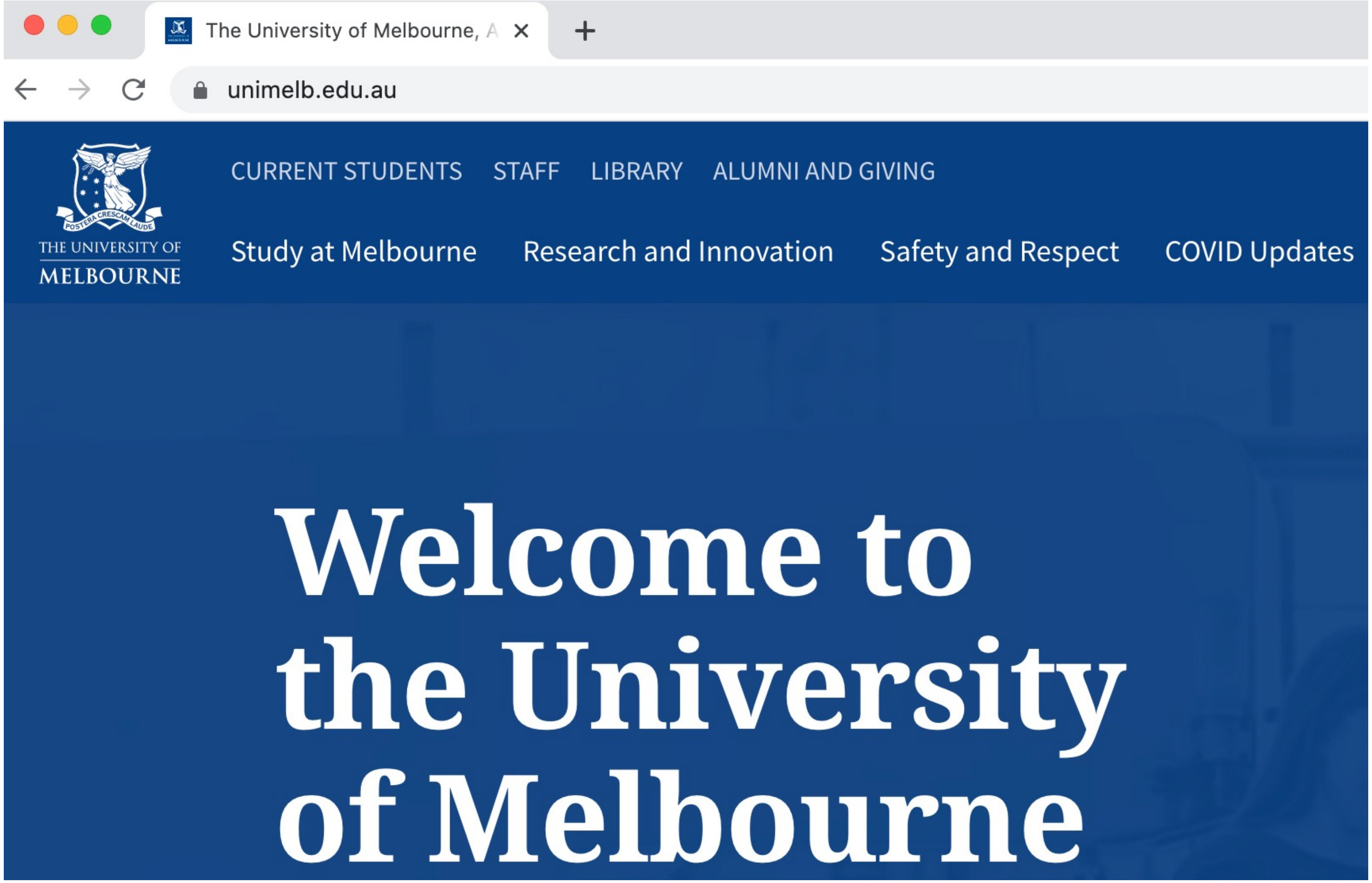
Project 1

- Announcement on LMS
- Spec is available via LMS
- Extra consultation hours
- Participation in Ed discussions

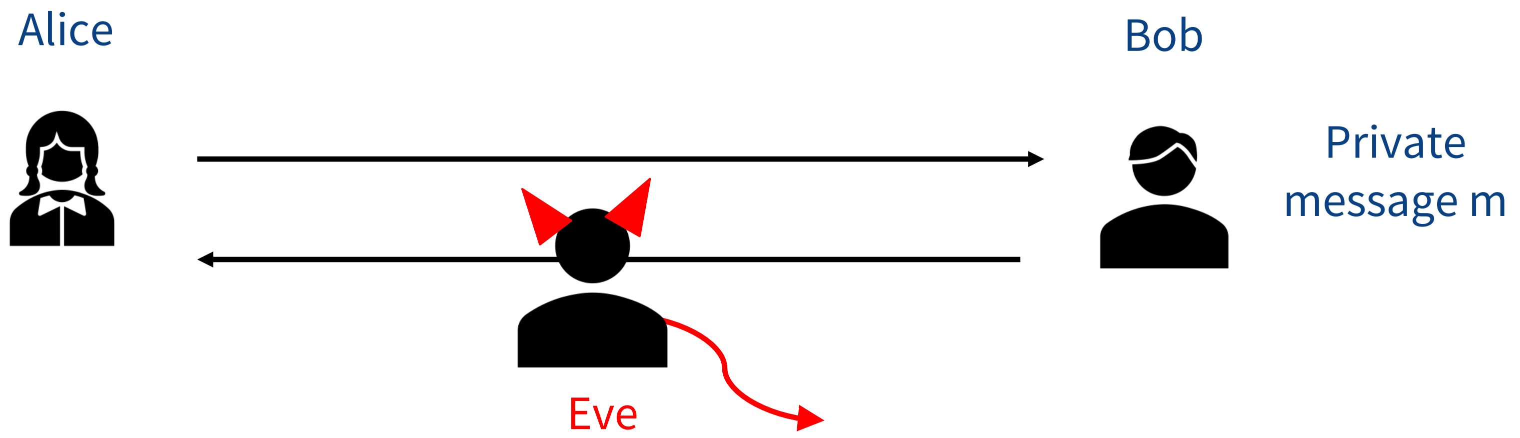
Recap

- Symmetric vs. asymmetric cryptography
- Encryption
- Signatures
- Hashing

What does the lock mean?

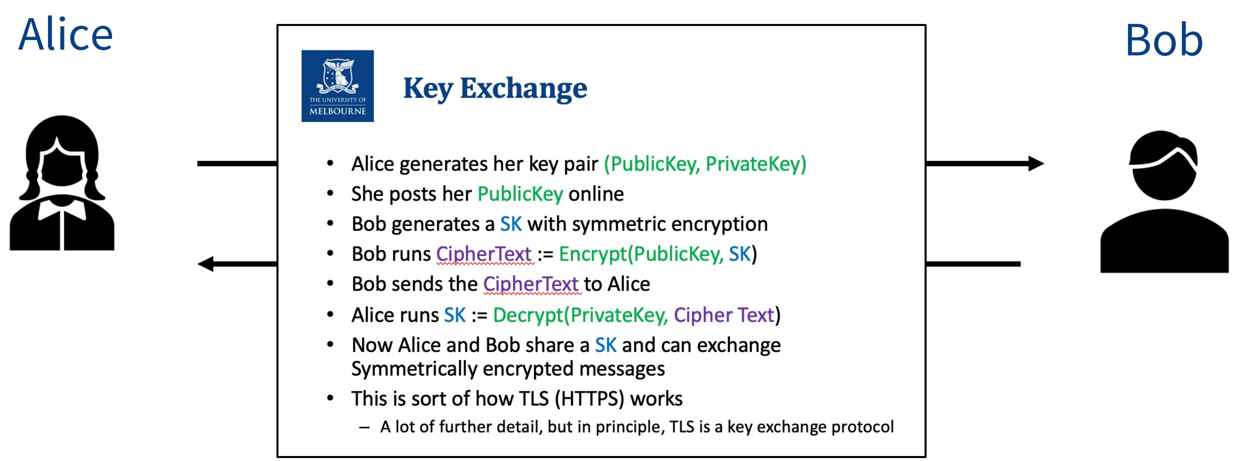


Goal: Secure Communication



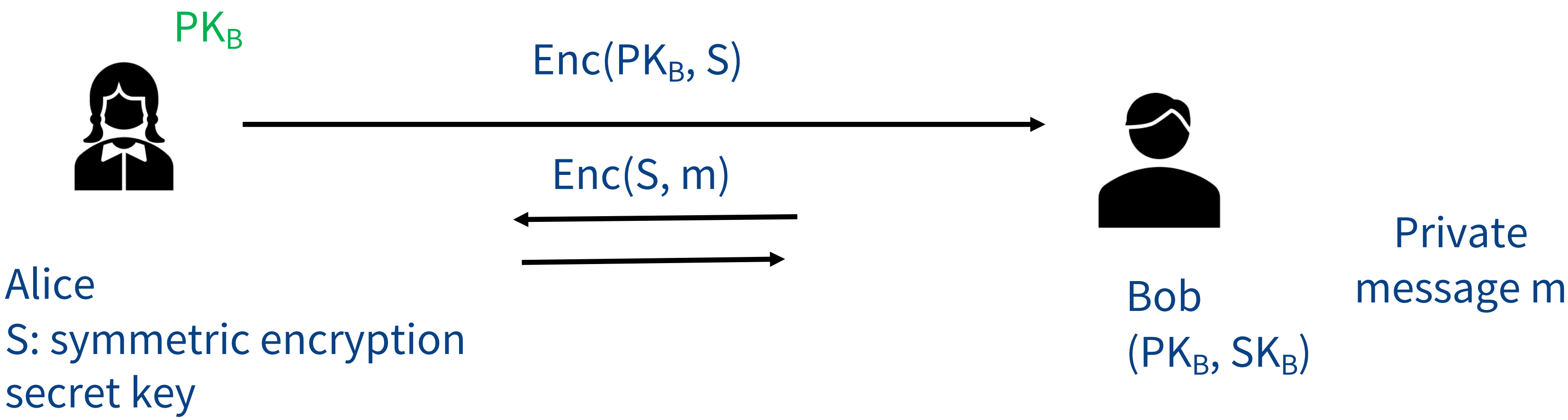
Adversary controls Wi-Fi, DNS, routers, can create its own websites, can listen to any packet, modify packets in transit, inject its own packets into the network

Goal: Secure Communication



Goal: Secure Communication

Why



PK_B
$Enc(PK_B, S)$
$Enc(S, m)$
Alice
S: symmetric encryption secret key
Bob
(PK_B, SK_B)
Private message m

Secure Communication

WHEN I SEND A MESSAGE TO SOMEONE, HOW DO I KNOW THEY ARE PPL I TRUST.

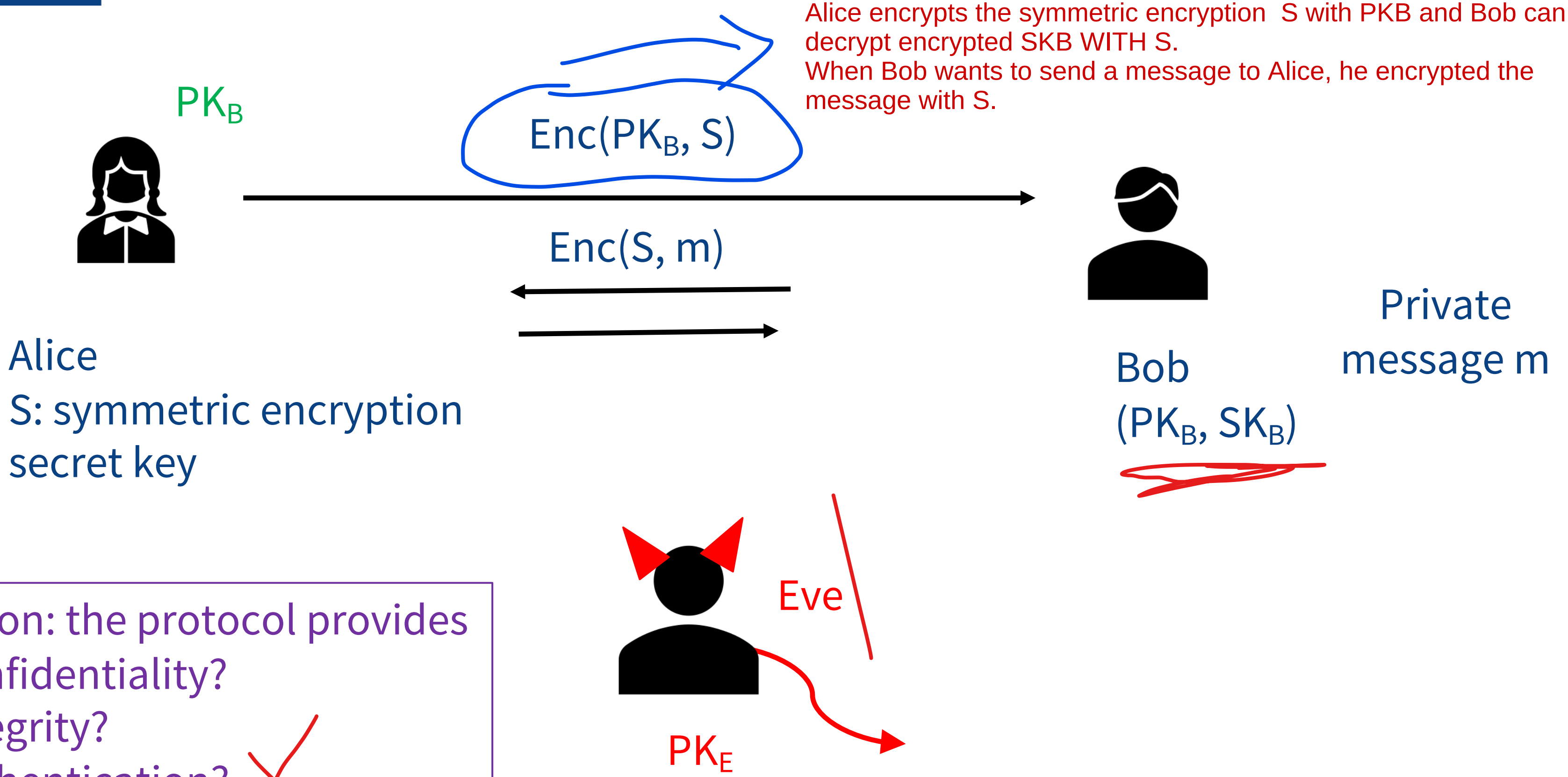
how do i know my message is not modified

Confidentiality – Authentication – Integrity

Objective is to provide **secure private** communication between two end-points, with **integrity checks** to ensure data does not change in transit, and **authentication** to establish identities of one or both of the end-points.

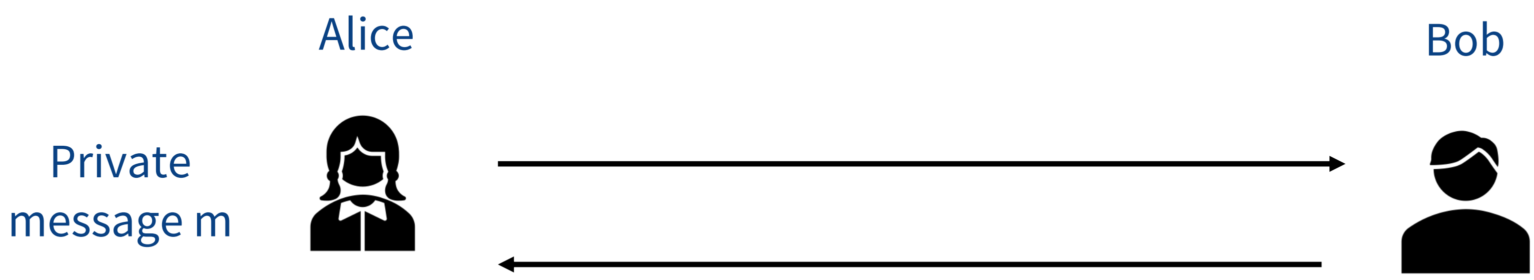
Why

Alice encrypts the symmetric encryption S with PKB and Bob can decrypt encrypted SKB WITH S.
When Bob wants to send a message to Alice, he encrypted the message with S.



Question: the protocol provides
- Confidentiality?
- Integrity?
- Authentication?

Goal: Secure Communication



2 problems:

1. How does Alice know ciphertext has not been modified?
2. How does Alice know PK_B is Bob's public key?

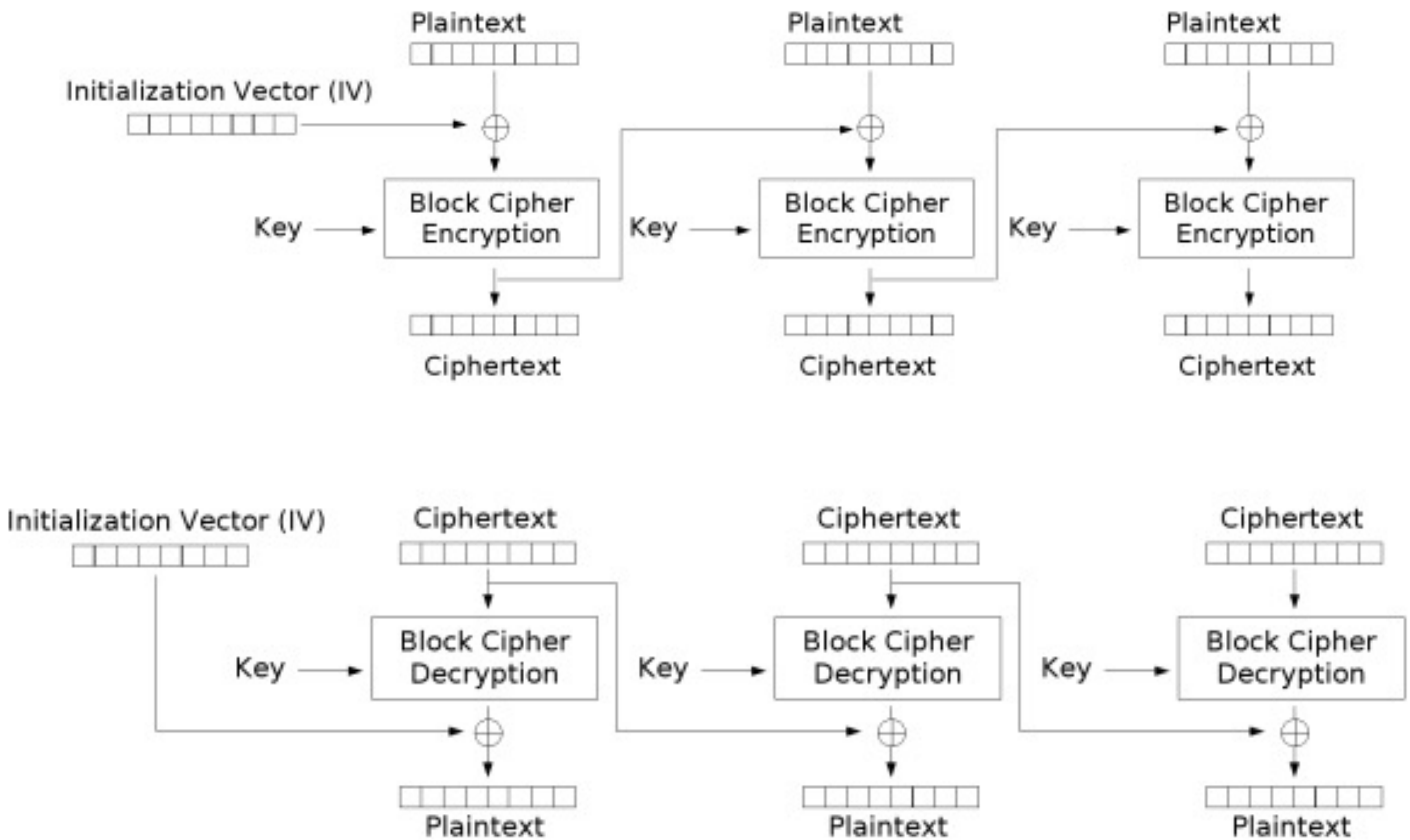
Today:
Towards Secure Communication

- Message authentication code
- Authenticated encryption

Properties we want: confidentiality and integrity

- Diffie Hellman Key Exchange
- Public Key Infrastructure (Certificates)

CBC – Cipher Block Chaining Tampering I (previous lecture)



Encryption

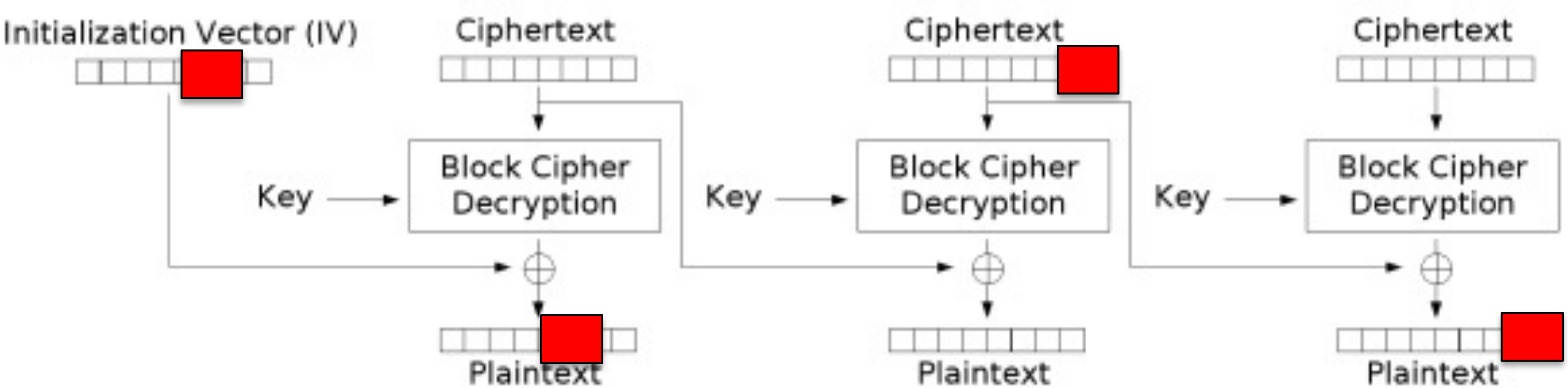
Decryption

Cipher Block Chaining (CBC) mode decryption

CBC – Cipher Block Chaining Tampering I (previous lecture)

Attacker can:
- reorder ciphertext
- flip bits

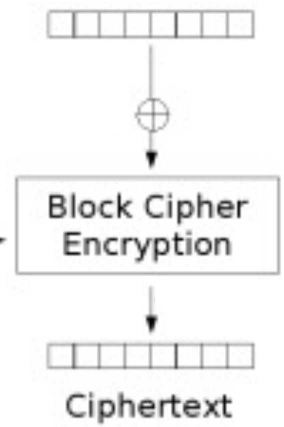
Every possible ciphertext corresponds to some valid plaintext



Cipher Block Chaining (CBC) mode decryption

Towards Authenticated Encryption

m: message



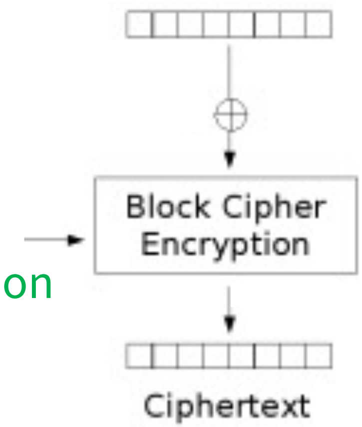
Secret key SK of symmetric encryption

c: ciphertext

Properties we want: confidentiality and integrity

Potential solutions for message authentication

m: message



Secret key SK of symmetric encryption

c: ciphertext

t := Authenticate (, m/c,)

Verify(…, ~~m/c, t,…)~~ ??

Hash: collision resistant hash function

- Hashing? (Is Hash(m) a good authentication method? Is Hash(c)?)
- Digital signatures? (Is Sign(SigningKey, m)? Is Sign(SigningKey, c)?)

we can only sign one of them (leave as a homework to figure out which one we should sign)

Left blank

Message Authentication Code (MAC)

Like we mac the original message to create a tag (this secret key (s1) used for the Mac function and and when receiver has the message + tag, they can put them as inputs for the Verify function. b = 0 => message was tampered. b = 1 => message was not tampered.

Message Authentication Code ensures integrity but not confidentiality because we dont encrypt the message.

- Detect if message has been tampered with
- s: MAC's secret key; m: message
- t := Mac(s,m); b := Verify(s,m,t)
  - b is 0/1 indicating successful verification
- Verifies integrity of a message using a secret key
- Security: Adversary cannot create (m', t') such that Verify(s,m',t') returns b = 1 for m' it has not seen

CBC-MAC and HMAC

CBC-MAC based on encryption (careful with variable length messages)

HMAC: Industry standard and used widely in practice

HMAC: Generate a MAC tag t:

$$t := \text{Hash}\ (\ (s \oplus \text{opad})\ \|\ \text{Hash}\ ((s \oplus \text{ipad})\ \|\ m))$$

ipad and opad are fixed constants used for padding

Authenticated Encryption

- Confidentiality and integrity of messages exchanged between Alice and Bob

- General construction: Encrypt-then-Mac:
  - c :=Encrypt(SK, m)
  - t := Mac(s, c)

- Encrypt -> Mac is more secure than Mac and Encrypt.
- If we use Digital Signature instead of Mac , it is not secure (figure out)

Message m
Secret key SK of symmetric encryption

- Verify: if Verify(s,t,c) returns 0, ~~do not decrypt~~

- Examples: AES-GCM, AES-OCB, AES-CCM

Today:
Towards Secure Communication

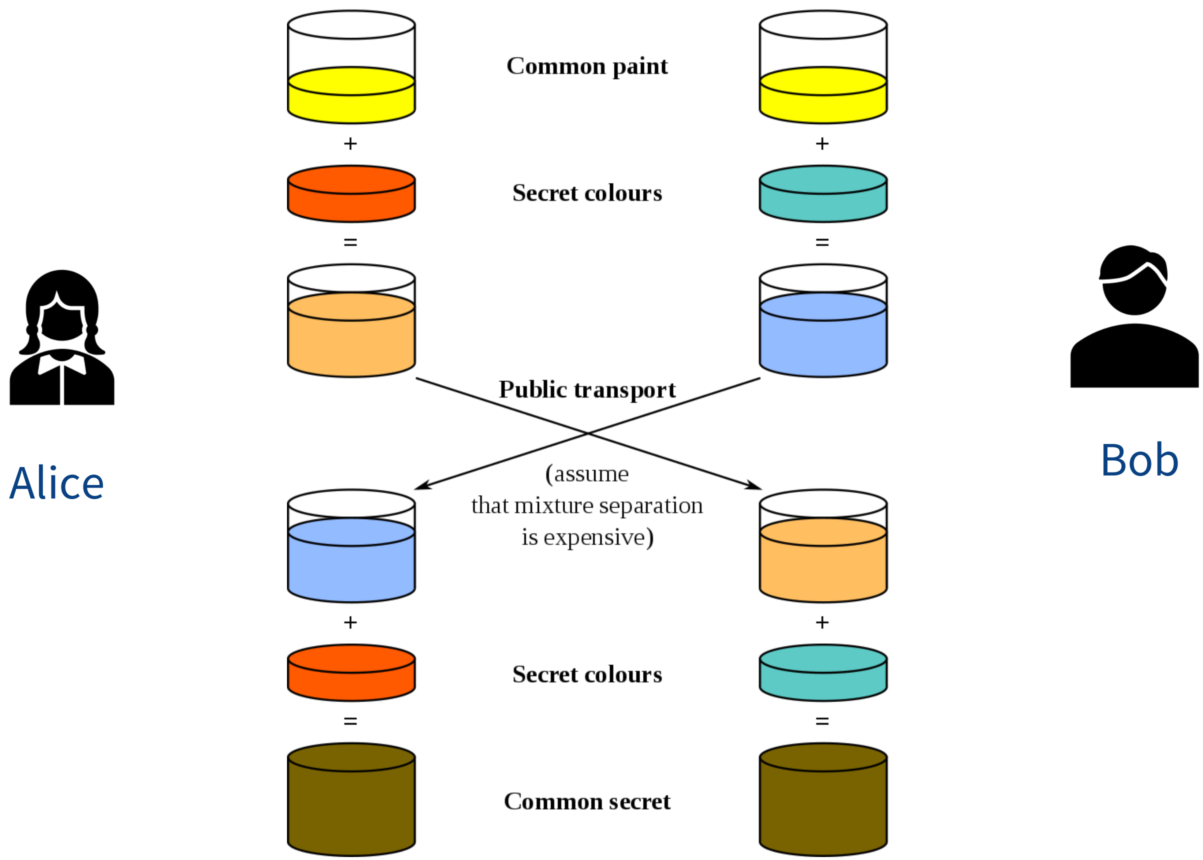
- Message authentication code
- Authenticated encryption
- Diffie Hellman Key Exchange
- Public Key Infrastructure

Diffie-Hellman Key Exchange

Turing award 2015

- Fundamental to protocols such as HTTPS, Secure Shell (SSH), Internet Protocol Security (IPsec), Simple Mail Transfer Protocol Secure (SMTPS), and other protocols that rely on Transport Layer Security (TLS).
- Agree on a shared key
- Provides perfect forward secrecy: exposure of long term keys does not compromise security of past sessions
- Sends information in a way that allows both parties to calculate a shared key without having to ever explicitly communicate the shared key

Diffie-Hellman Key Exchange



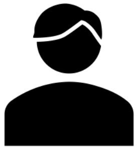
Diffie-Hellman Key Exchange

- Generate some public information:
  - A large prime p
  - A generator g (primitive root modulo p)
- Alice picks a random value x and computes $X=g^x \bmod p$
  - Sends X to Bob
- Bob picks a random value y and computes $Y=g^y \bmod p$
  - Send Y to Alice
- Alice calculates the secret $s=Y^x \bmod p = g^{yx} \bmod p$
- Bob calculates the secret $s=X^y \bmod p = g^{xy} \bmod p$
- $g^{yx} \bmod p = g^{xy} \bmod p$

Left blank

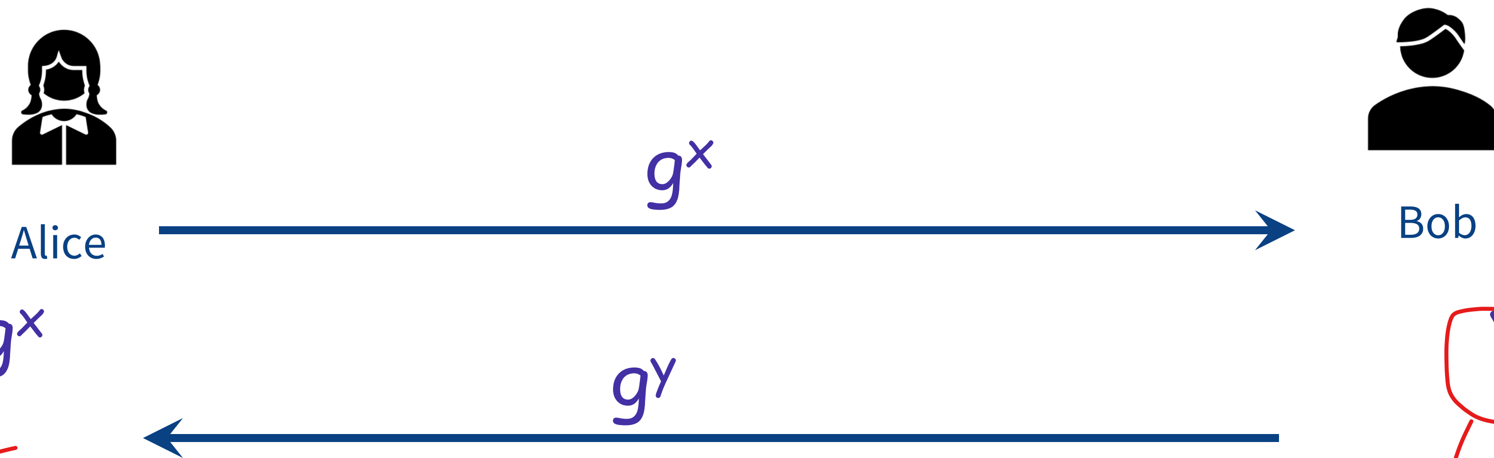
Diffie-Hellman Exchange [DH'76]

Alice



Bob

Diffie-Hellman Exchange [DH'76]



x, g^x

y, g^y

secret key of Alice

secret key of Bob

both parties compute the secret key $s=g^{xy}$

shared secret key.

Diffie-Hellman Key Exchange

- At the end of the process we have a shared secret, the component parts of which we have never openly communicated
- Solving the discrete log (in the particular group we operate) is considered a hard problem
- As such, it is considered infeasible to recover the x from g^x
- Provided the two parties discard their secrets, even if one of them loses their private key, it will not allow past communication to be decrypted

What does it mean secure

- Secret key should look indistinguishable from random
- DH key exchange relies on Decisional DH

Assumptions based on Hard Problems

Problem	Given	Figure out
Discrete logarithm (DL)	g^x	x
Computational Diffie-Hellman (CDH)	g^x, g^y	g^{xy}
Decisional Diffie-Hellman (DDH)	g^x, g^y, g^z	Is $z \equiv xy \pmod{\lvert G \rvert}$?

Figure 10.1: An informal description of three discrete logarithm related problems over a cyclic group G with generator g. For each problem we indicate the input to the attacker, and what the attacker must figure out to "win." The formal definitions are in the text.

https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf

Assumptions based on Hard Problems

Problem	**Given**	**Figure out**
Discrete logarithm (DL)	g^x	x
Computational Diffie-Hellman (CDH)	g^x, g^y	g^{xy}
Decisional Diffie-Hellman (DDH)	g^x, g^y, g^z	Is $z \equiv xy \pmod{\lvert G \rvert}$?

Figure 10.1: An informal description of three discrete logarithm related problems over a cyclic group G with generator g. For each problem we indicate the input to the attacker, and what the attacker must figure out to "win." The formal definitions are in the text.

Bonus Question:

- If you can solve DL, can you solve CDH?
- If you can solve CDH, can you solve DDH?

https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf

Summary

- Message authentication code
- Authenticated encryption
- Diffie Hellman Key Exchange

Acknowledgement

- The slides were prepared by Olya Ohrimenko based on some material developed previously by Chris Culnane
- Reference: KR 8.3, 8.3.1, 8.3.2 and references from Week 4 Lecture 1
- Some of the images included in the notes were supplied as part of the teaching resources accompanying the text books listed in lecture 1.
  - (And also) Wikimedia Commons