

Cryptography Basics – Key exchange

Assignment Project Exam Help

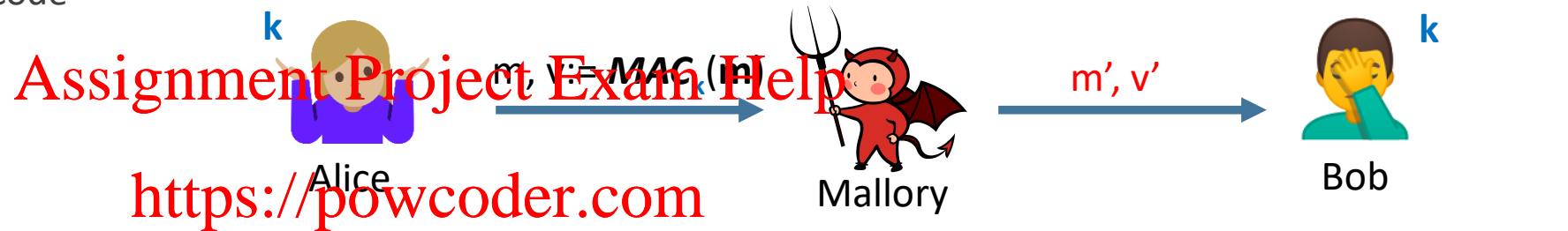
<https://powcoder.com>

Add WeChat powcoder

Review

Integrity: prevent Mallory from tampering

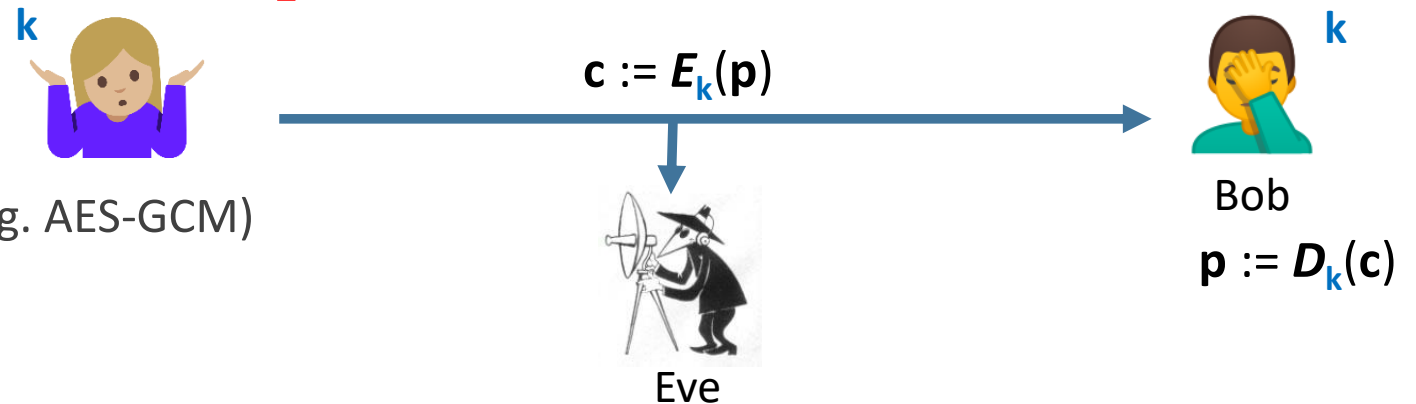
- Message Authentication Code
- Hashes -> HMAC
 - Use: SHA2, SHA3



Confidentiality: prevent eavesdropper (Eve) from learning the (plaintext) message

- Stream ciphers
 - AES-CTR, ChaCha20
- Block ciphers
 - AES-CBC (caution: padding oracles!)

Add WeChat powcoder



Best practice: Authenticated ciphers (e.g. AES-GCM)

- Encrypt, then MAC

Sharing k

Amazing fact:

Alice and Bob can have a public conversation to derive a shared key!

Diffie-Hellman (D-H) key exchange **Assignment Project Exam Help**

1976: Whit Diffie, Marty Hellman

with ideas from Ralph Merkle

(earlier, in secret, by Malcolm Williamson of British intelligence agency)

Relies on a mathematical hardness assumption called *discrete log problem*
(a problem believed to be hard)

<https://powcoder.com>



Add WeChat powcoder

Diffie-Hellman protocol

D-H protocol

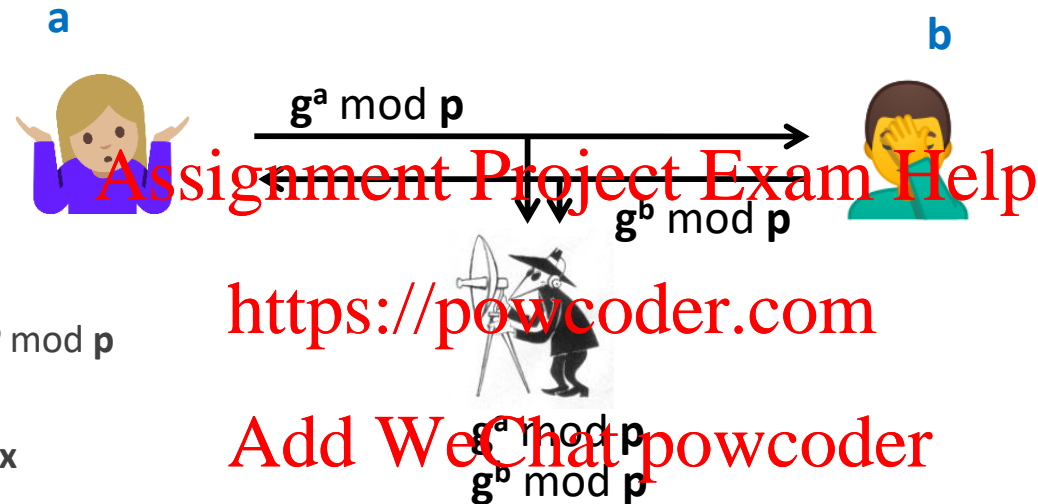
1. Alice and Bob agree on public parameters (maybe in standards doc*, or pick them)
 p : a large “safe prime” s.t. $(p-1)/2$ is also prime
 g : a square mod p (but not 1)

Assignment Project Exam Help

2. **Alice** a
Generates random secret value a .
($0 < a < p$)

Bob b
Generates random secret value b .
($0 < b < p$)

Communication:
Alice sends $g^a \bmod p$ to Bob.
Bob sends $g^b \bmod p$ to Alice.
<https://powcoder.com>
Add WeChat: powcoder
3. **Alice**
Computes x
 $= (g^b \bmod p)^a \bmod p$
 $= g^{ba} \bmod p$
Bob
Computes x'
 $= (g^a \bmod p)^b \bmod p$
 $= g^{ab} \bmod p$

(Notice that $x == x'$)
Can use $k := \text{HMAC}_0(x)$ as a shared key.

DH passive eavesdropping attack



Eve wants to compute $x = g^{ab} \bmod p$

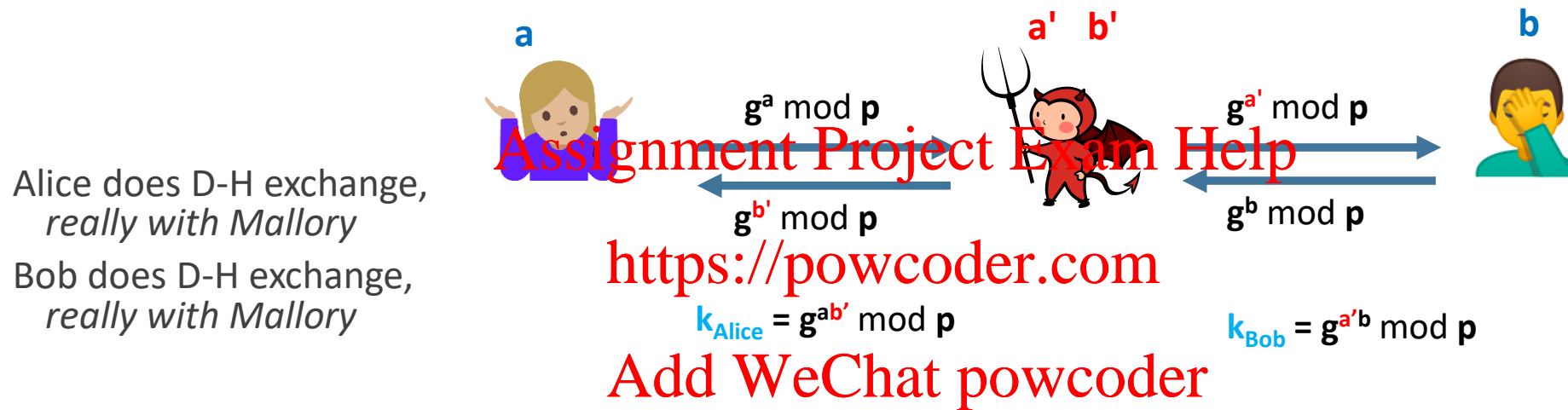
Best known approach:
Find a or b , then compute x

Finding y given $g^y \bmod p$ is an instance of the **discrete log problem**:
No known efficient algorithm*

Best practice: Use large DH group size (e.g. 2048-bit primes)
or a more secure group (Elliptic curve cryptography)

[Breakout exercise: what about Mallory (active attacks)?]

Man-in-the-middle (MITM) attack



Alice and Bob each think they are talking with the other,
but really Mallory is between them and knows both secrets

Bottom line:

D-H gives you secure connection, but you don't know who's on the other end!

Defending D-H against MITM attacks

- Cross your fingers and hope there isn't an active adversary.
- Rely on out-of-band communication between users. [Examples?]
- Rely on physical contact to make sure there's no MITM. [Examples?]
- Integrate D-H with user authentication.

If Alice is using a password to login to Bob, leverage the password:

Instead of a fixed g , derive g from the password – Mallory can't participate w/o knowing password.

- Use digital signatures. [More next week.]

Public key encryption

Can Alice share a “public key” ($g^a \bmod p$) and have anyone encrypt a message only she can read?

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Public key encryption

Can Alice share a “public key” ($g^a \bmod p$) and have anyone encrypt a message only she can read?

Diffie-Hellman doesn't allow this directly, but with some math:

Assignment Project Exam Help

Alice's **public key** is $A = g^a \bmod p$ and her **private key** is a

<https://powcoder.com>

Bob has Alice's public key, and a message m he wants to send her:

- Pick a random value r $[0, p-2]$
 - Compute $R = g^r \bmod p$
 - Compute $S = m * A^r \bmod p$
 - Send Alice (R, S)
- Add WeChat powcoder**

To decrypt:

- Alice computes $S * R^{-a} \bmod p = m * A^r * g^{r(-a)} \bmod p = m * g^{ar} g^{r(-a)} \bmod p = m * g^{ar-ar} \bmod p = m * g^0 \bmod p = m$