CMSC/Math 456: Cryptography (Fall 2022)

Lecture 14

Daniel Gottesman

Administrative

Midterm: Thursday, Oct. 20 (1 week from today)

- In class
- Open book (including textbook), no electronic devices
- Will cover material through Diffie-Hellman and El Gamal, but not RSA.

Please fill out the poll on Piazza on which topics to review on Tuesday, Oct. 18. Vote by tonight for the biggest impact.

This class is being recorded

Diffie-Hellman and Encryption

Diffie-Hellman generates a key, a random element of a prime-order group G (a subgroup of $\mathbb{Z}_p^*$ or an elliptic curve).

How do we use it to encrypt?

We can use it as the key for a pseudo one-time pad.

But … it is not a bit string.

Use some key derivation function H(k) to convert it into a bit string. H(k) needs to be carefully chosen to make sure the key still appears uniform.

Example: Key k is random number in $\{0,\ldots,96\}$. Write k in binary (7 bits) and let H(k) be the least significant 6 bits of k.

Problem: About 1/3 of the time, $64 \leq k < 96$, which means the most significant bit of H(k) is 0. When $k < 64$, the first bit of H(k) is random. Overall, first bit is more likely to be 0!

This class is being recorded

Another Approach to Encryption

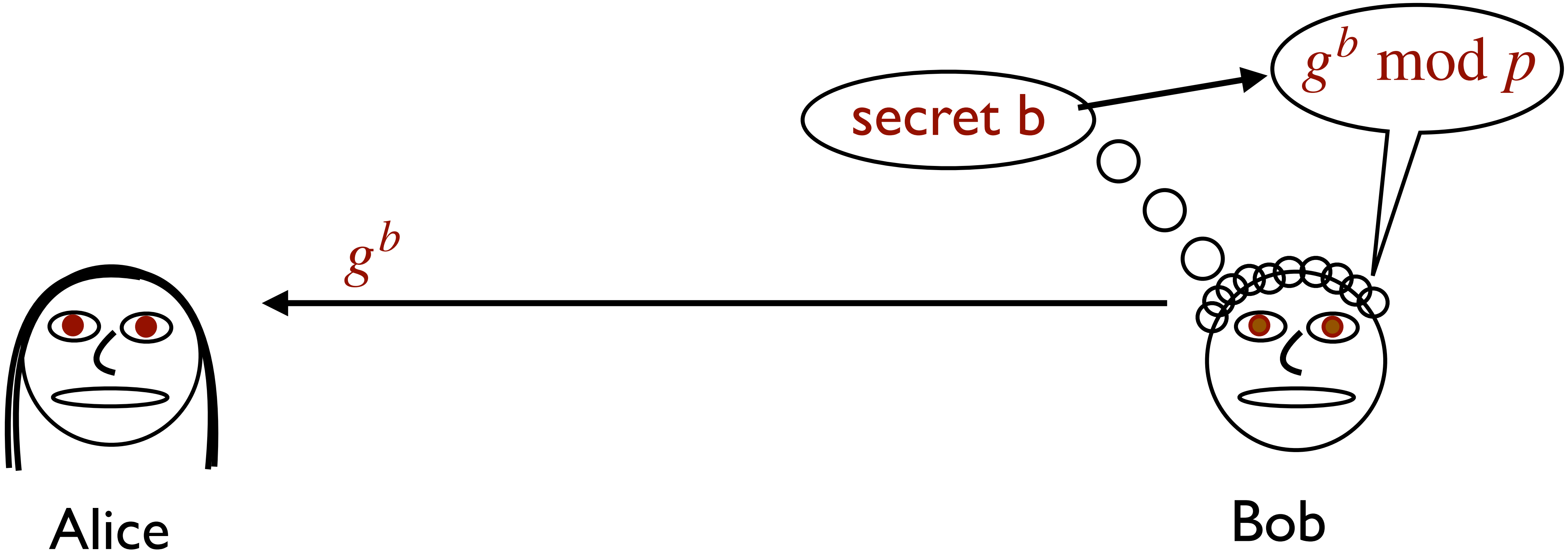
Another approach to encryption is to integrate encryption into the key exchange process.

To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

Another Approach to Encryption

Another approach to encryption is to integrate encryption into the key exchange process.

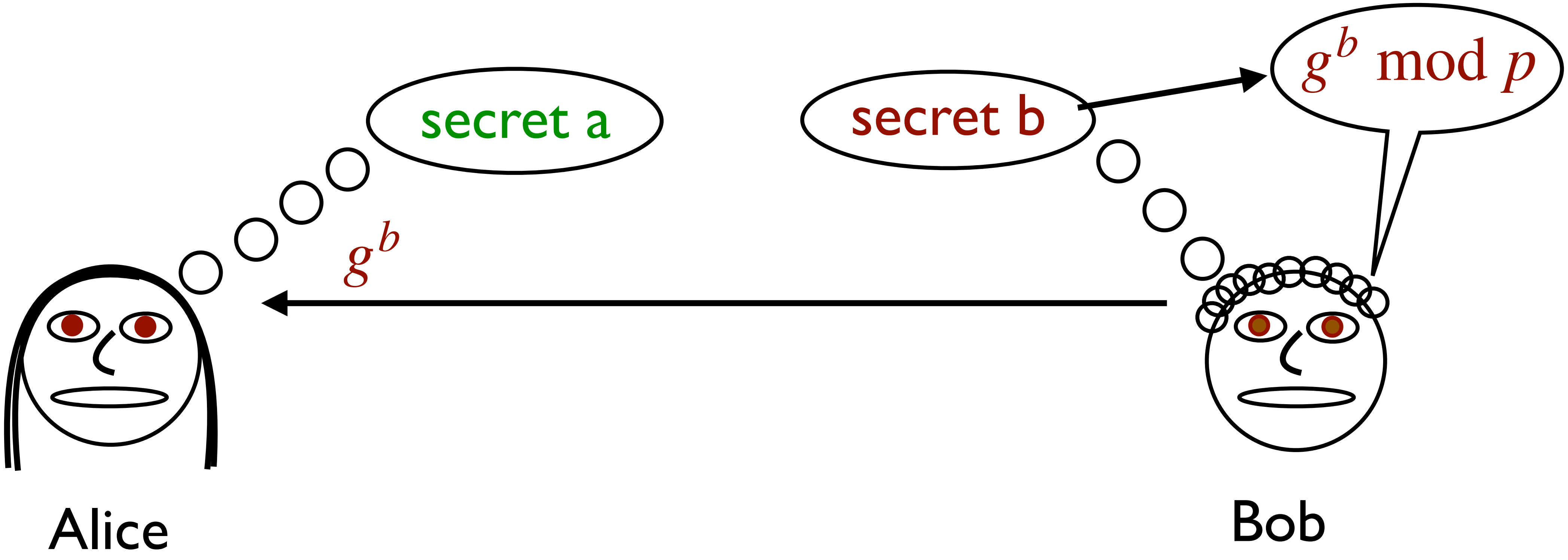


To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

Another Approach to Encryption

Another approach to encryption is to integrate encryption into the key exchange process.

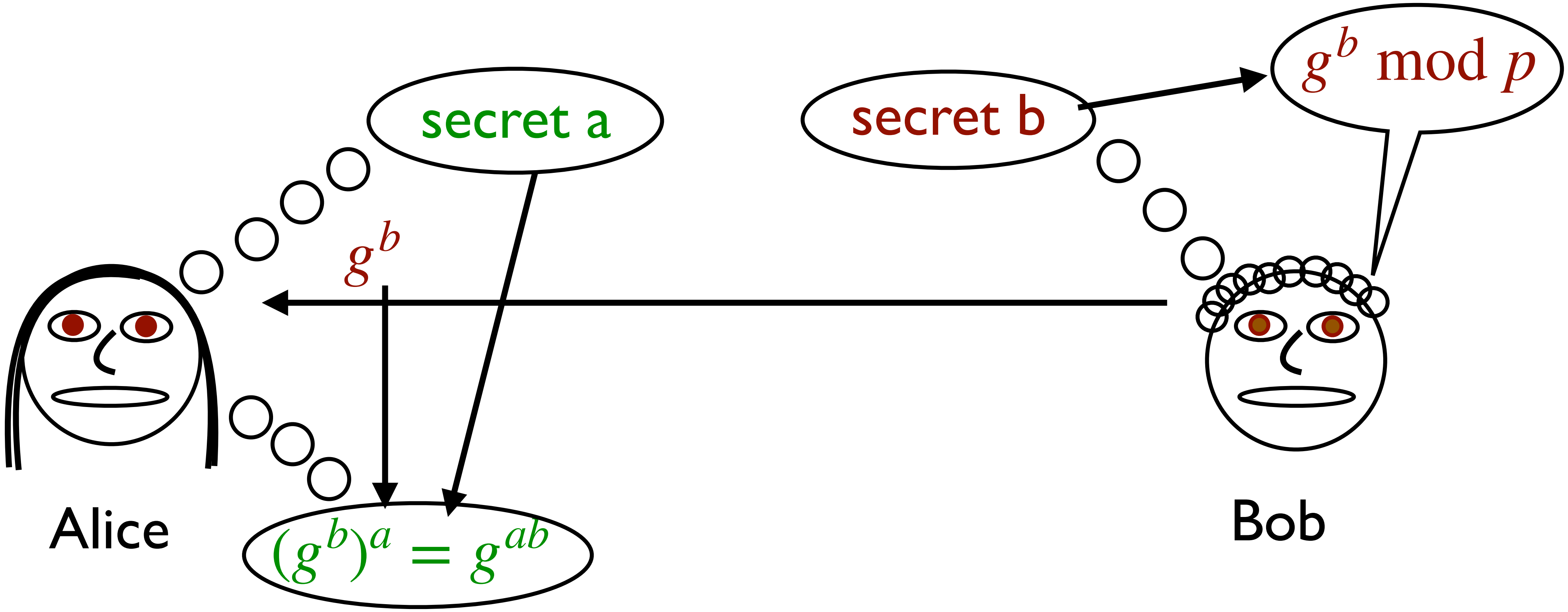


To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

Another Approach to Encryption

Another approach to encryption is to integrate encryption into the key exchange process.

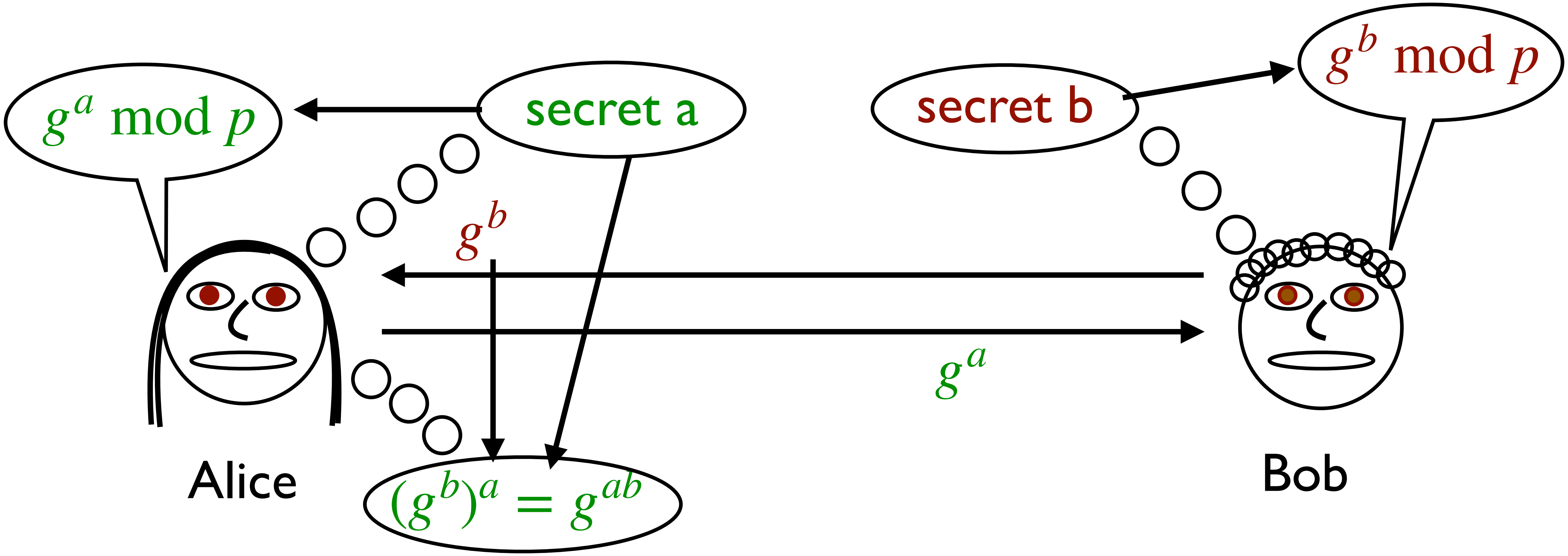


To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

Another Approach to Encryption

Another approach to encryption is to integrate encryption into the key exchange process.

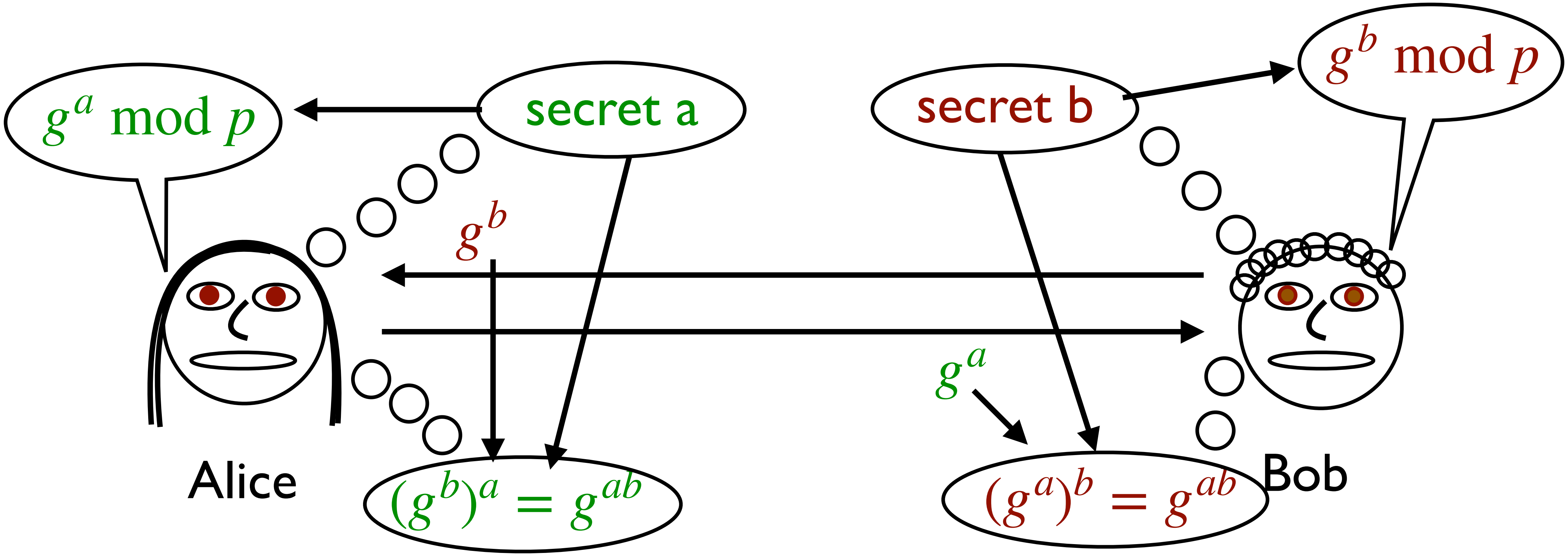


To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

Another Approach to Encryption

Another approach to encryption is to integrate encryption into the key exchange process.



To see how to do this, the first step is notice that Alice's and Bob's announcements in Diffie-Hellman don't have to be simultaneous.

This class is being recorded

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.

secret b

Alice

Bob

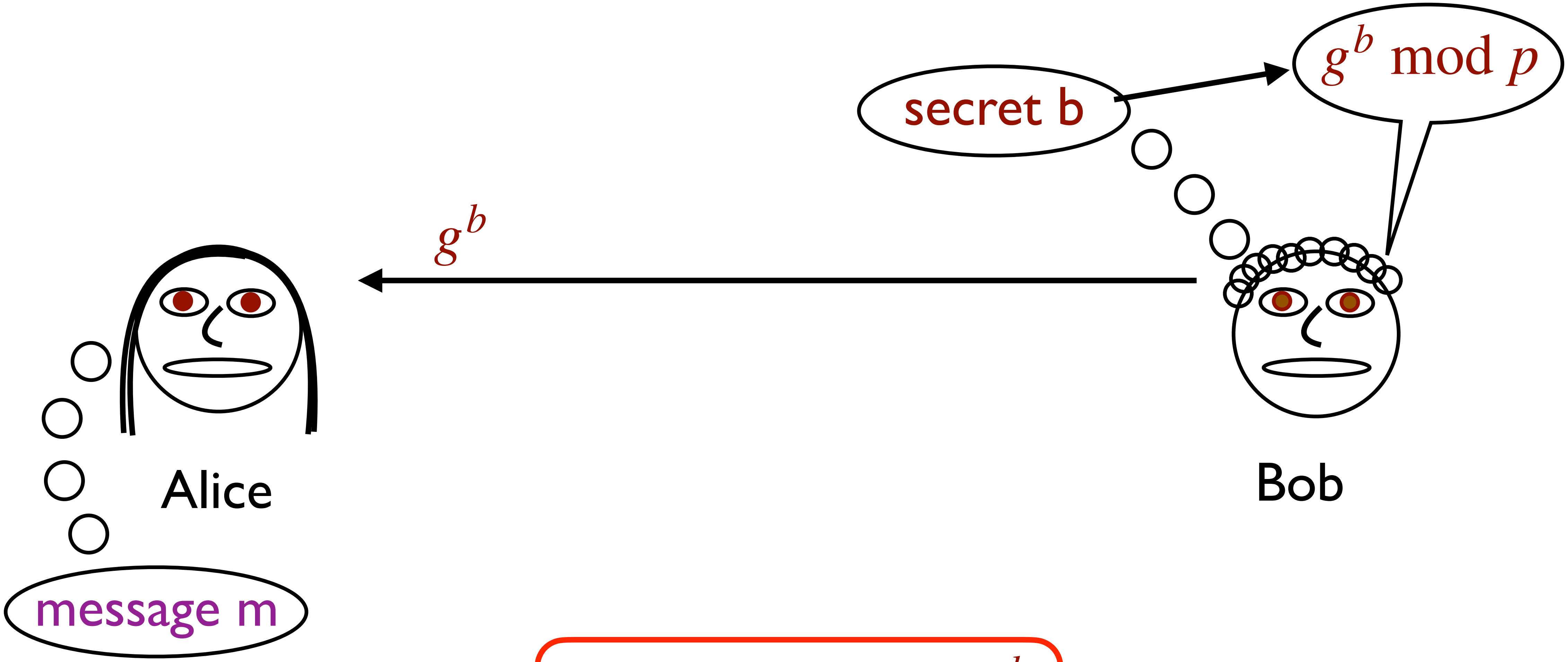
message m

Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.



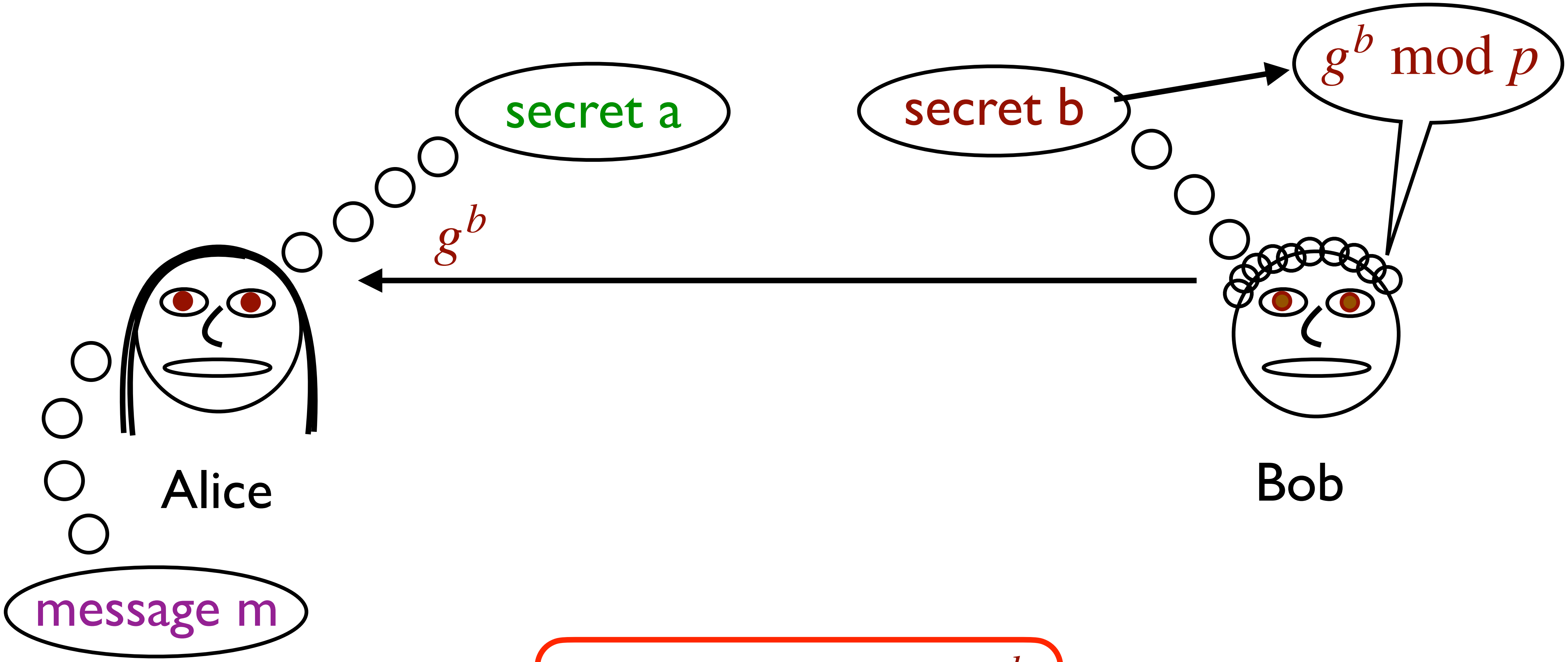
Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

This class is being recorded

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.



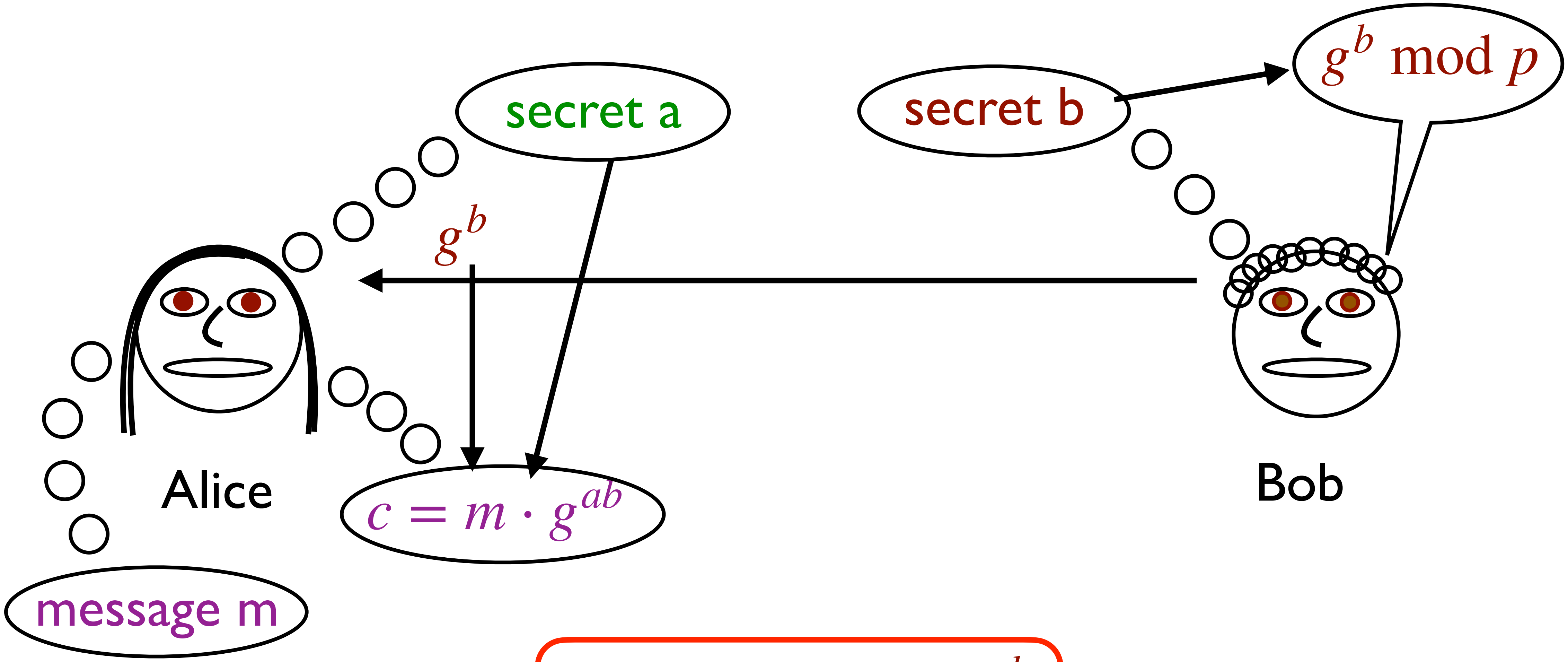
Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

This class is being recorded

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.



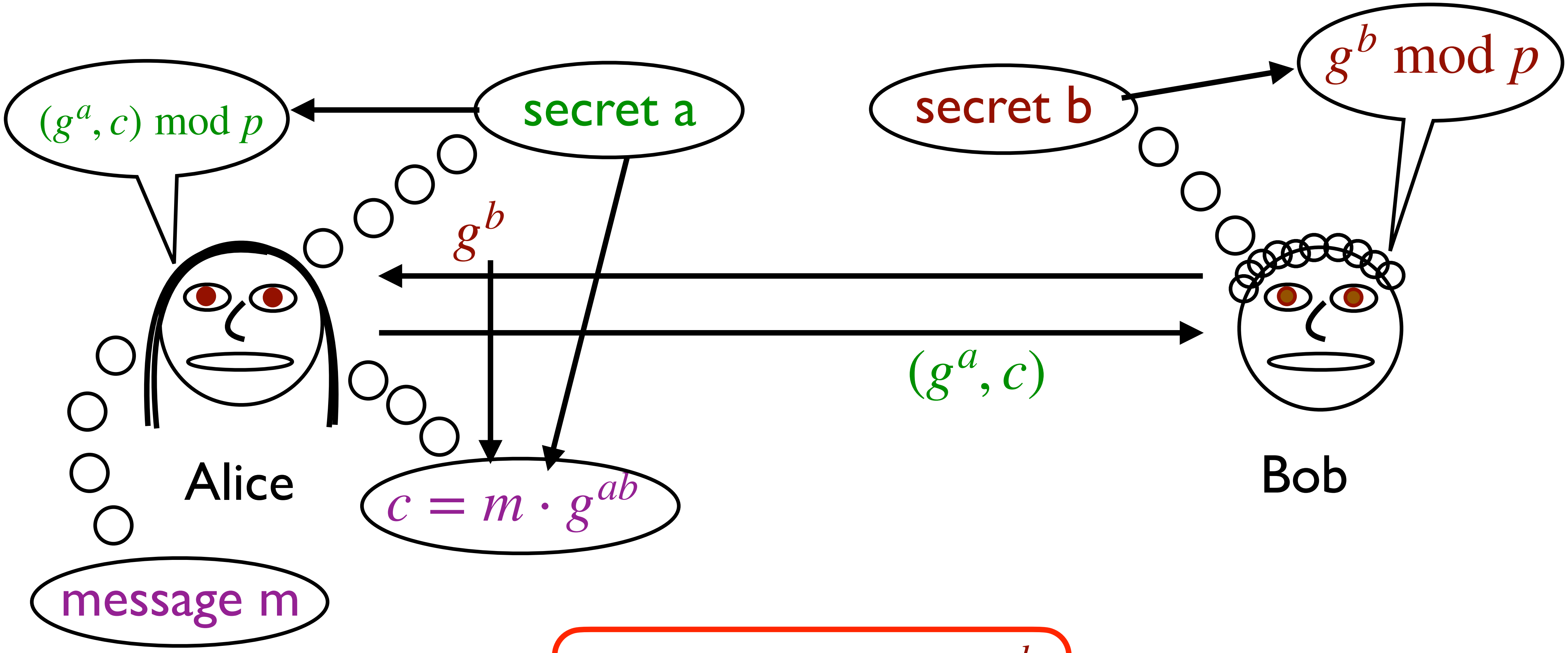
Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

This class is being recorded

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.



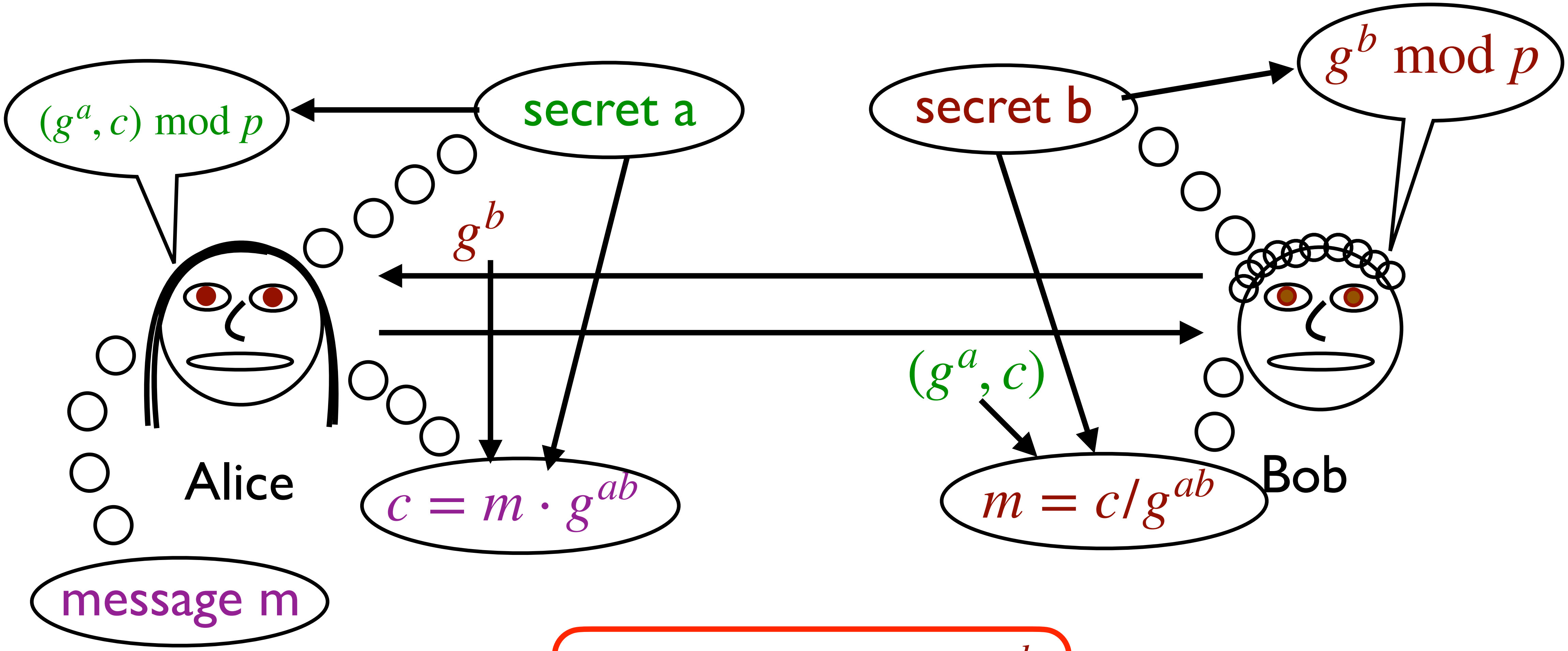
Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

This class is being recorded

El Gamal Encryption

When Alice sends g^a, she can also send an encrypted message m, for instance with ciphertext $c = m \cdot g^{ab}$.



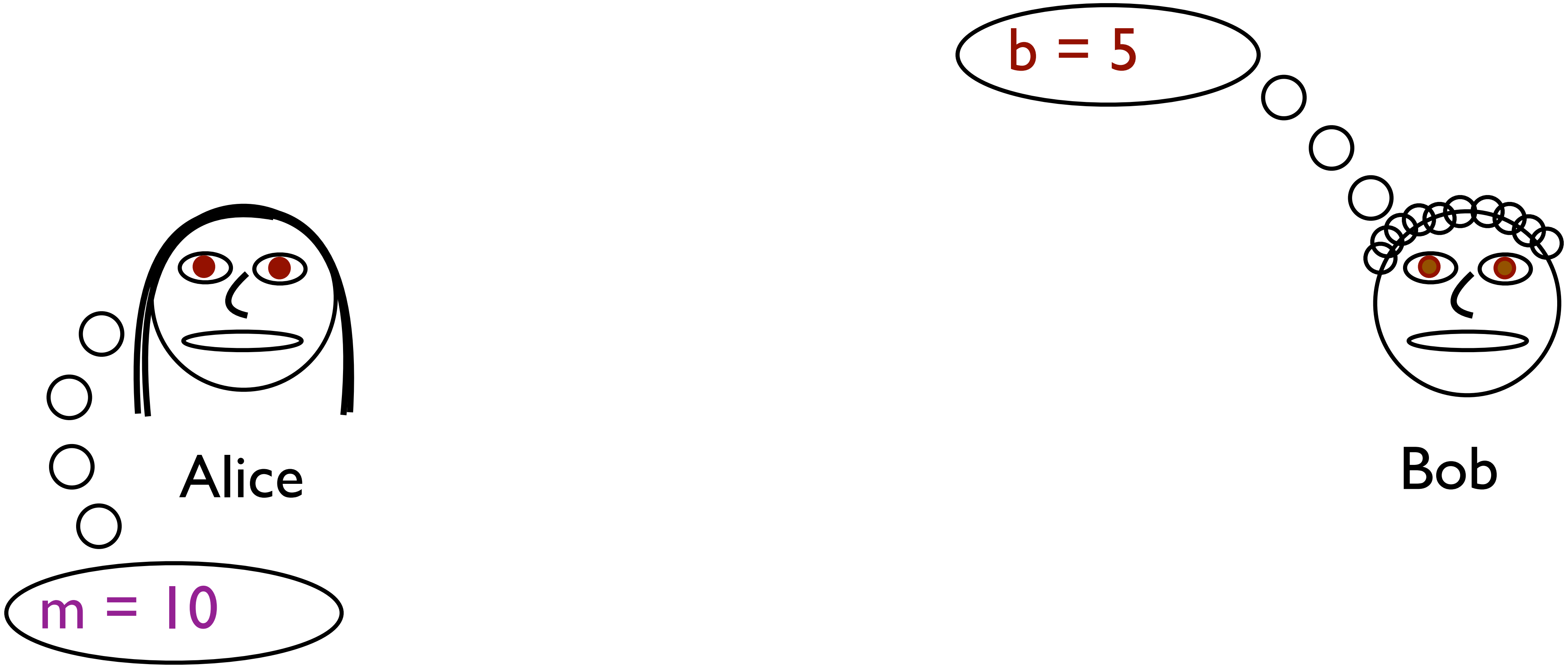
Bob's public key is g^b.

Advantage over Diffie-Hellman: non-interactive

This class is being recorded

El Gamal Example

Example using $p = 23$, $g = 2$. (g has order 11 in $\mathbb{Z}_{23}^{*}$.)



b = 5

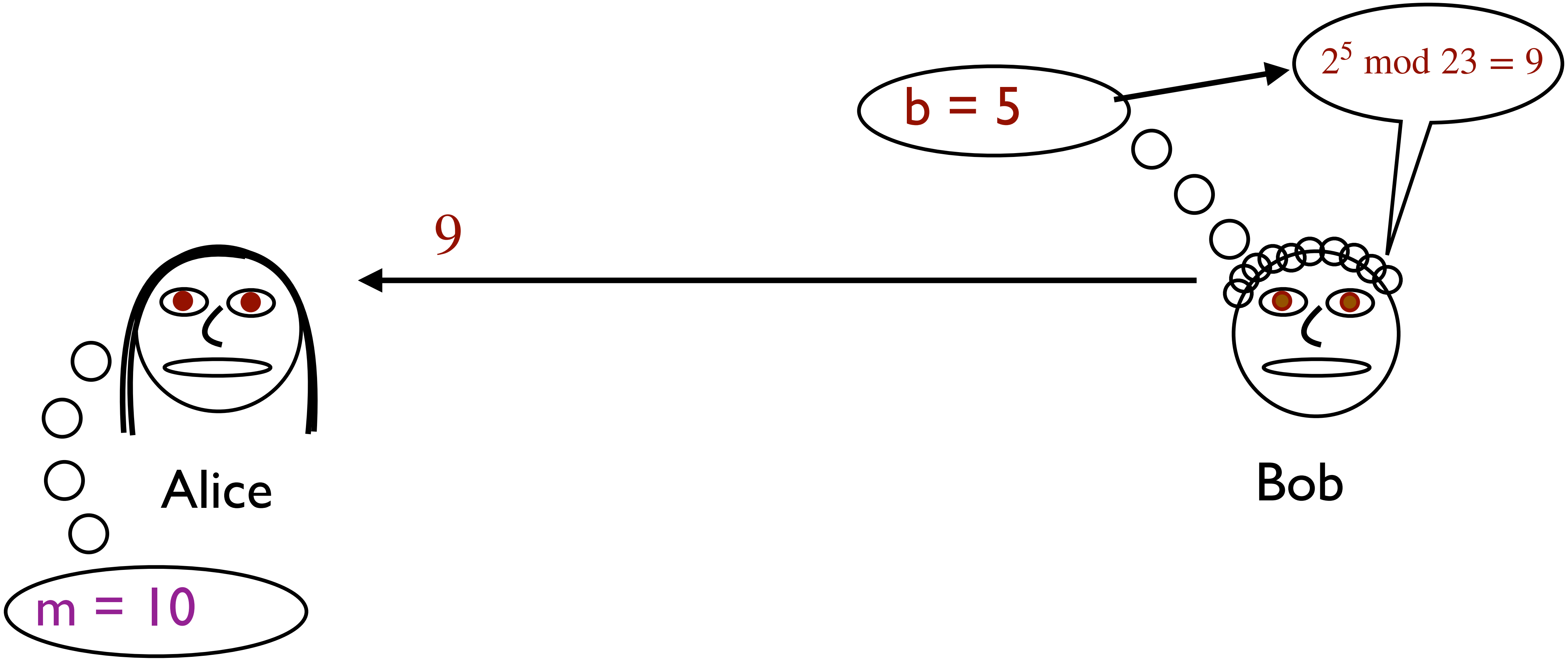
Alice

Bob

m = 10

El Gamal Example

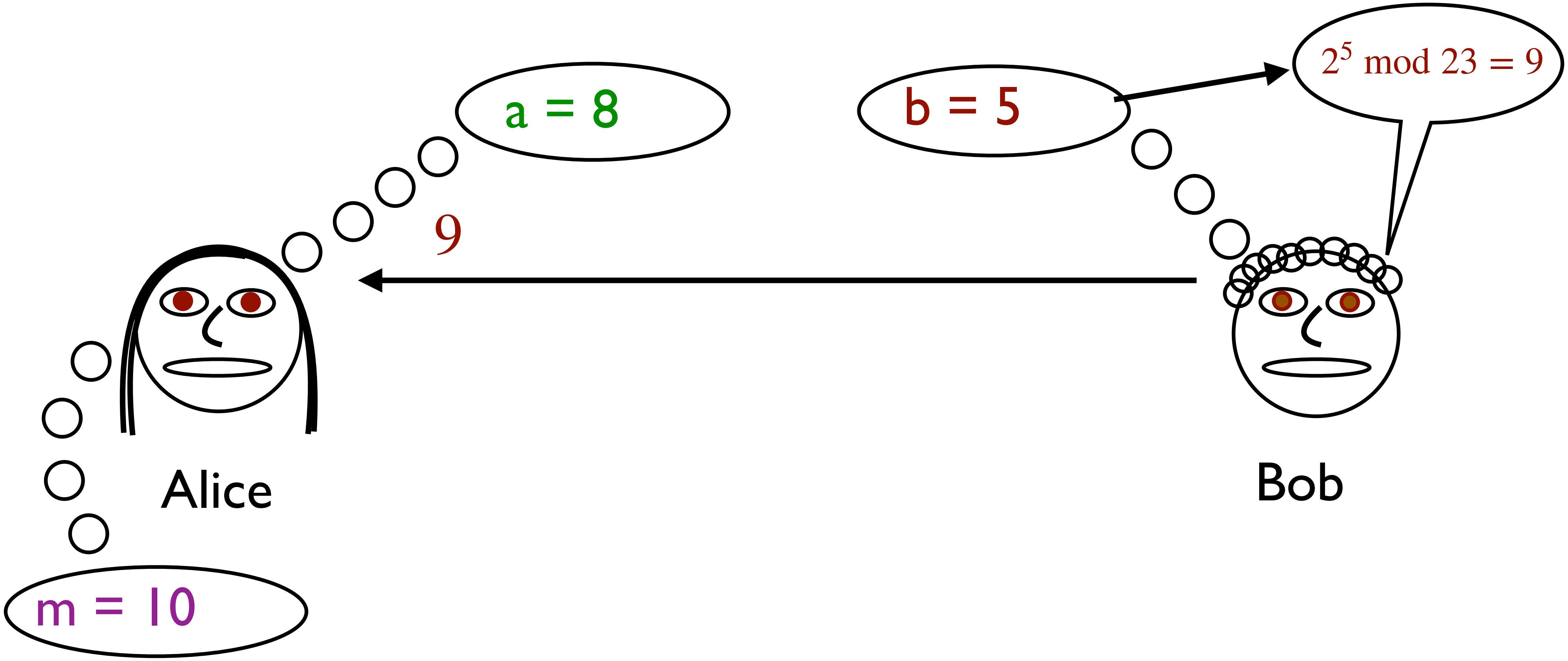
Example using p = 23, g = 2. (g has order 11 in $\mathbb{Z}_{23}^*$.)



This class is being recorded

El Gamal Example

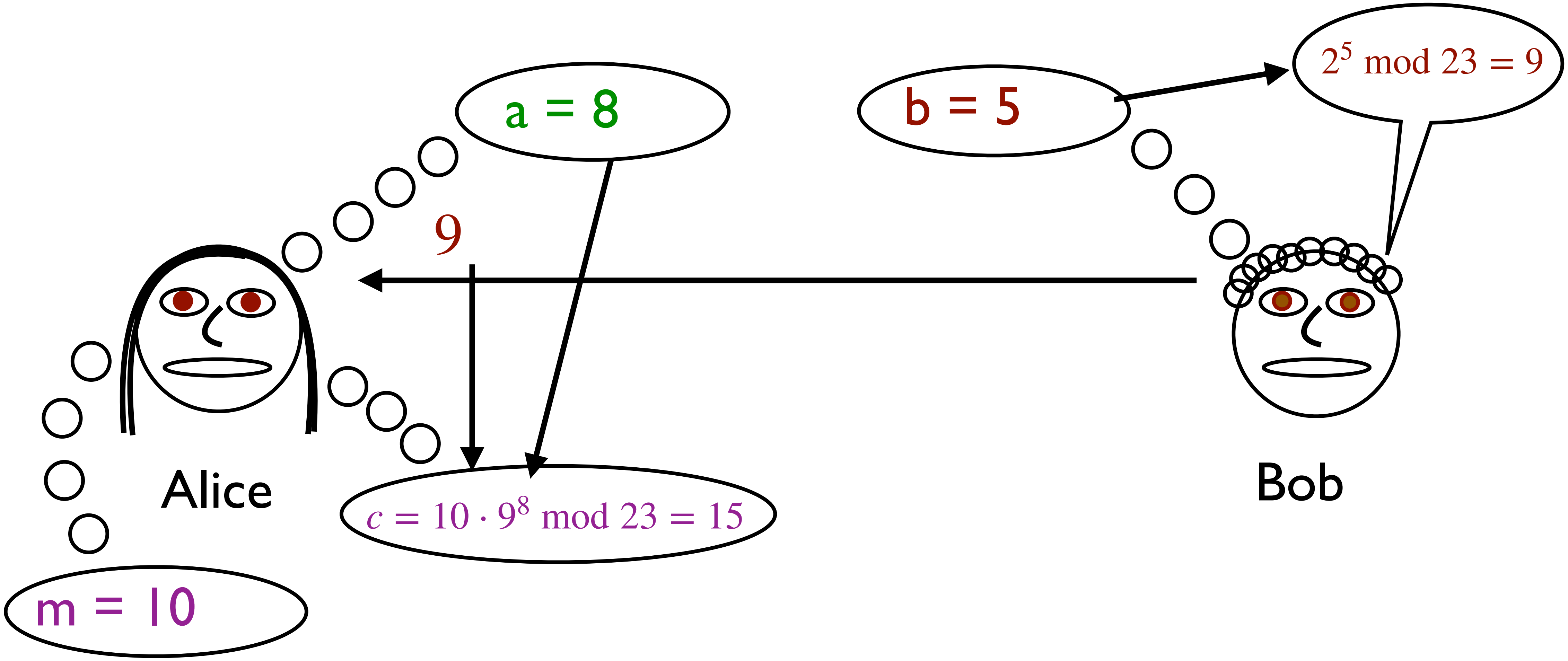
Example using p = 23, g = 2. (g has order 11 in $\mathbb{Z}_{23}^*$.)



This class is being recorded

El Gamal Example

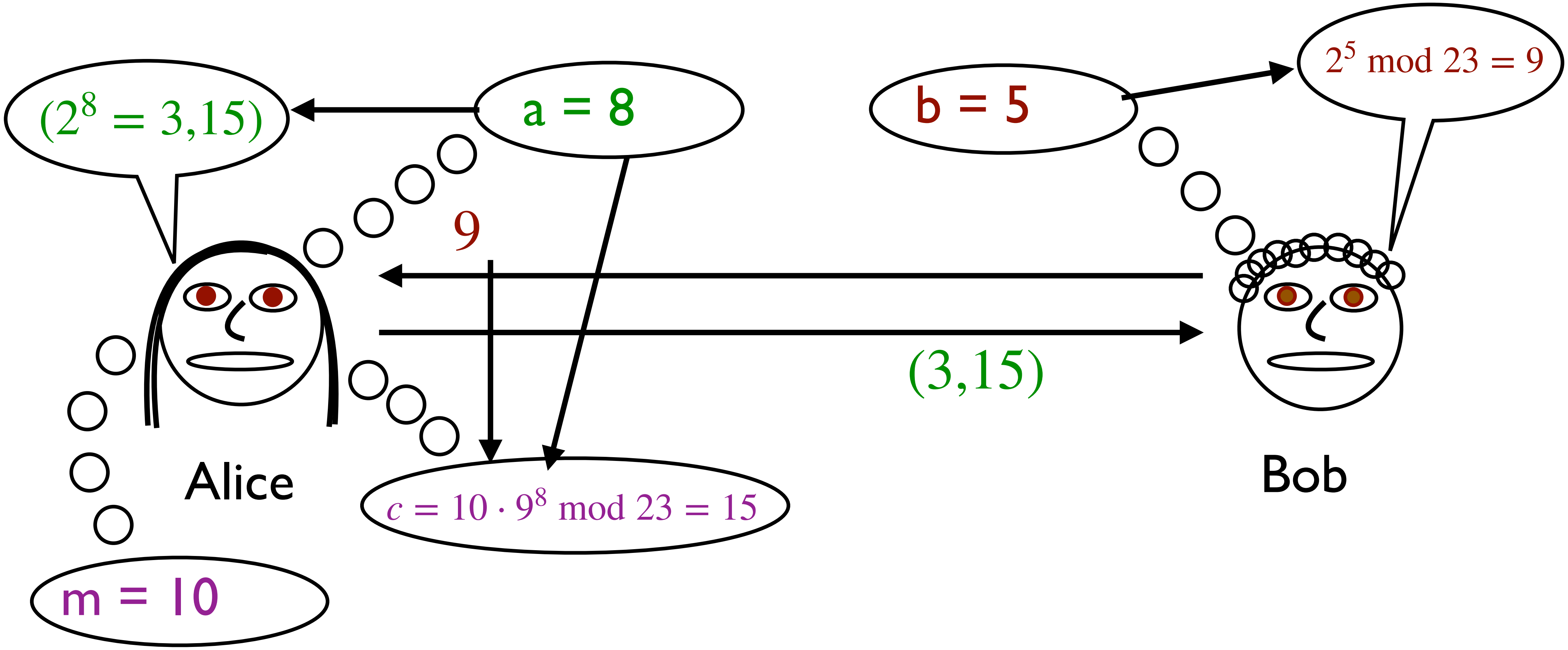
Example using p = 23, g = 2. (g has order 11 in $\mathbb{Z}_{23}^*$.)



This class is being recorded

El Gamal Example

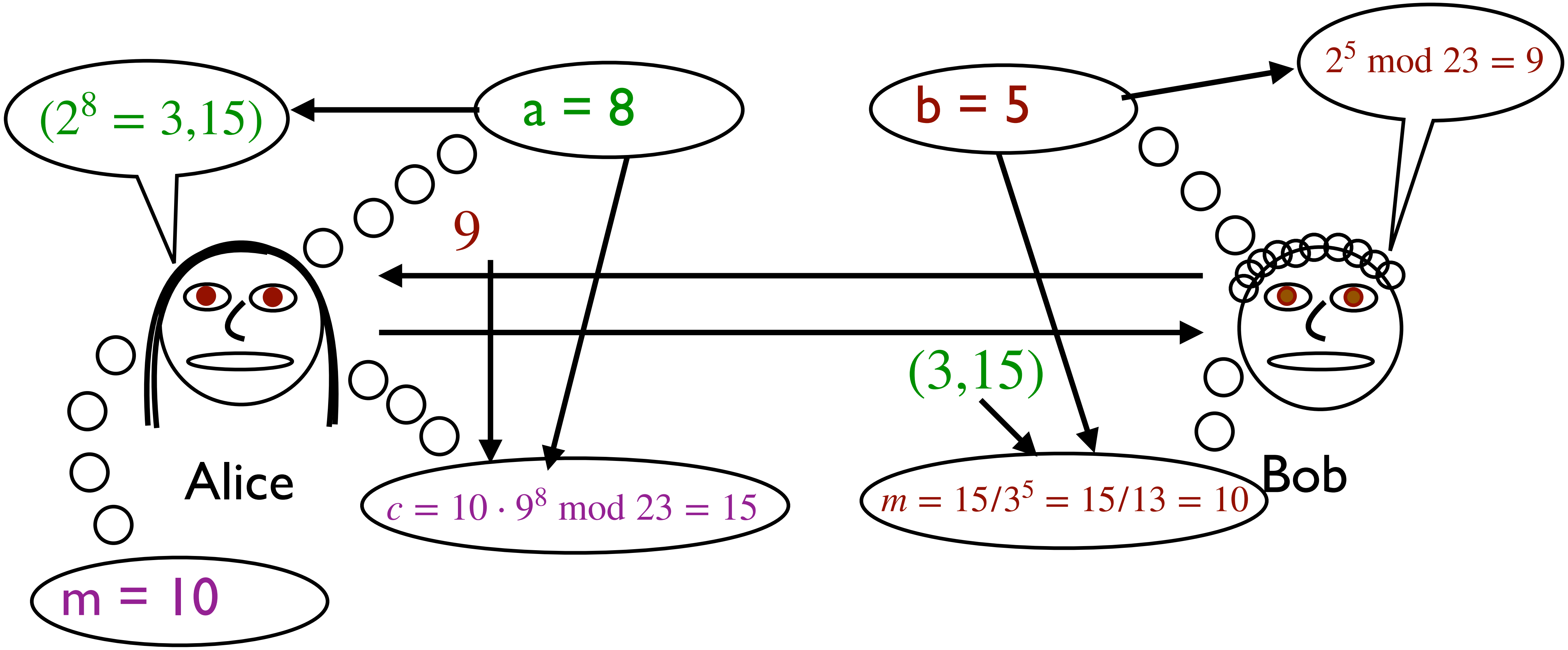
Example using p = 23, g = 2. (g has order 11 in $\mathbb{Z}_{23}^*$.)



This class is being recorded

El Gamal Example

Example using $p = 23$, $g = 2$. (g has order 11 in $\mathbb{Z}_{23}^*$.)



$(2^8 = 3,15)$

a = 8

b = 5

$2^5 \bmod 23 = 9$

9

$(3,15)$

Alice

Bob

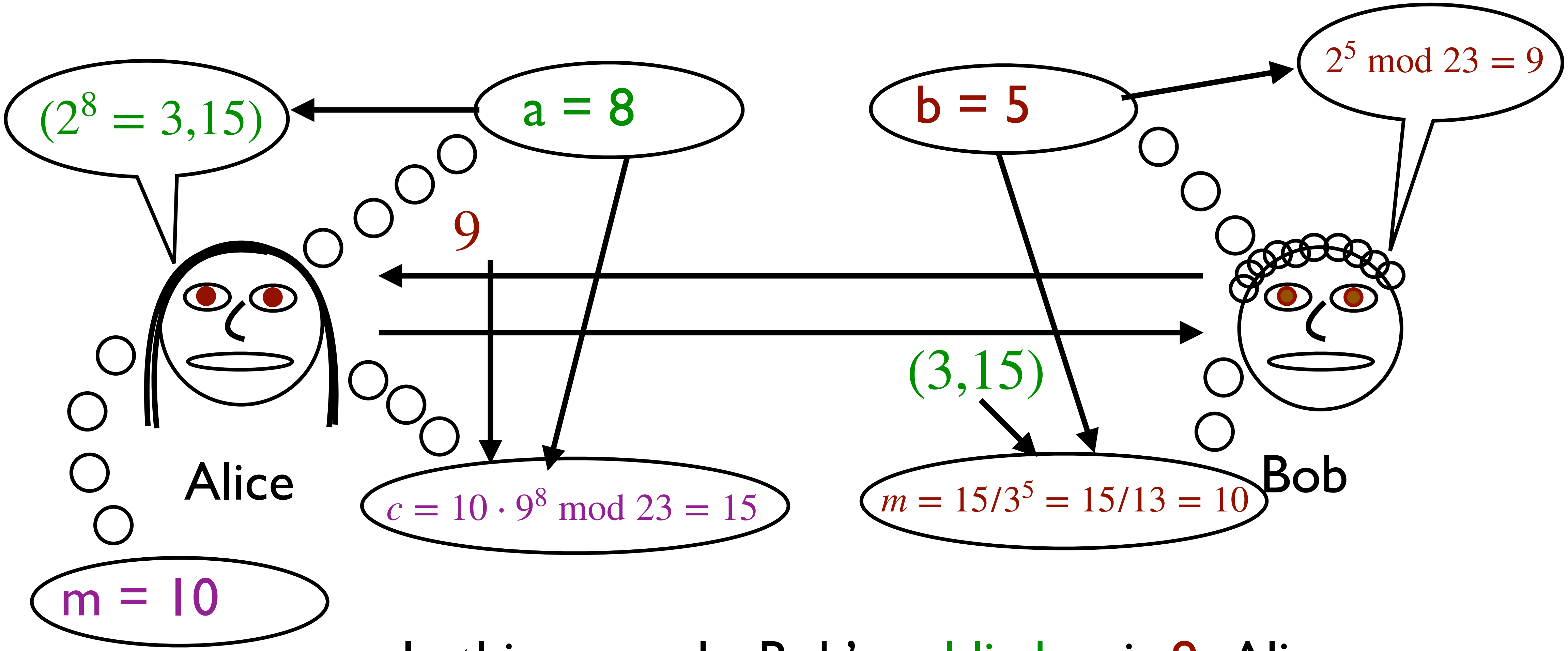
$c = 10 \cdot 9^8 \bmod 23 = 15$

$m = 15/3^5 = 15/13 = 10$

m = 10

El Gamal Example

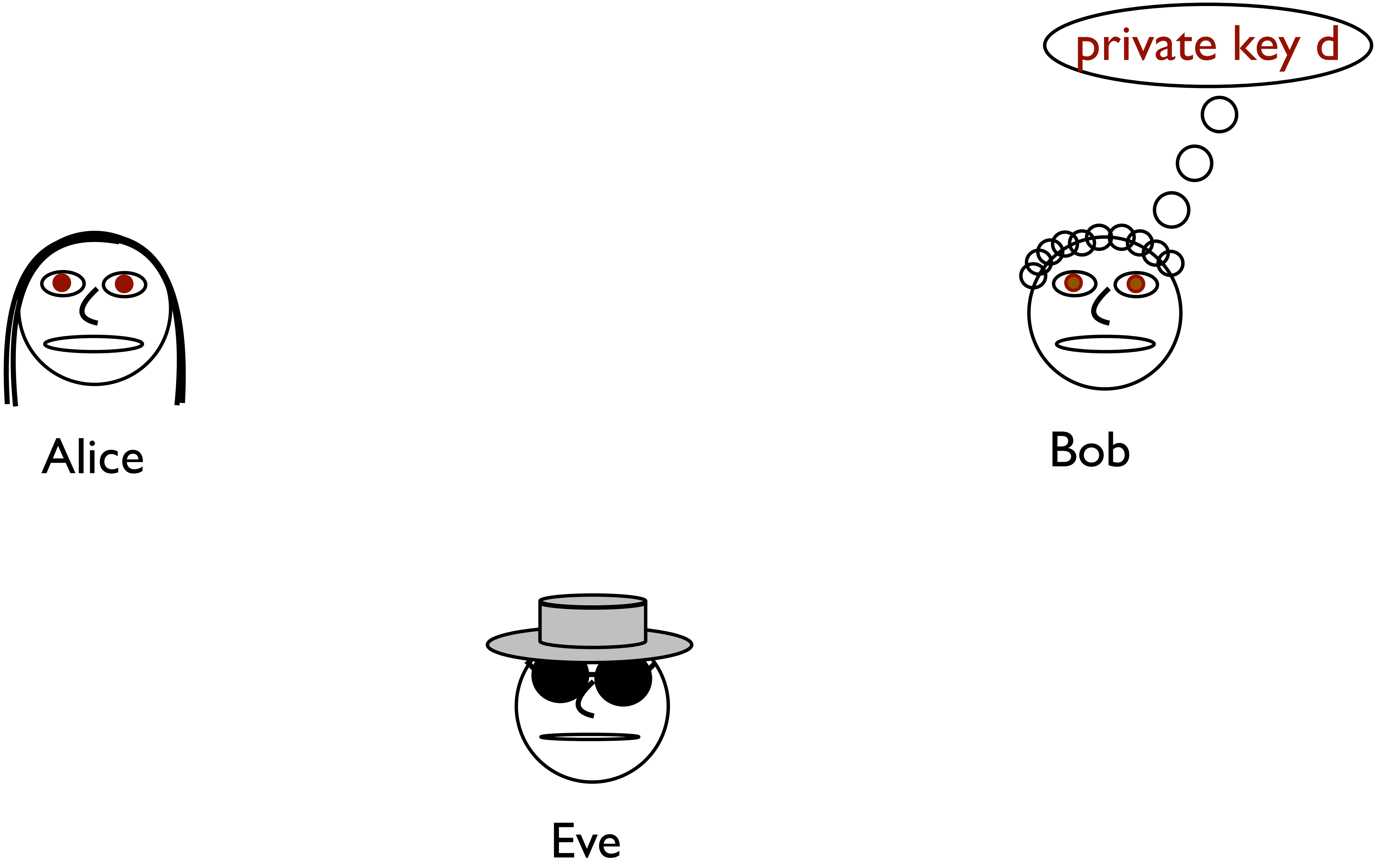
Example using p = 23, g = 2. (g has order 11 in $\mathbb{Z}_{23}^*$.)



In this example, Bob's public key is 9. Alice uses it to encrypt. Bob's private key is 5. He uses it to decrypt.

This class is being recorded

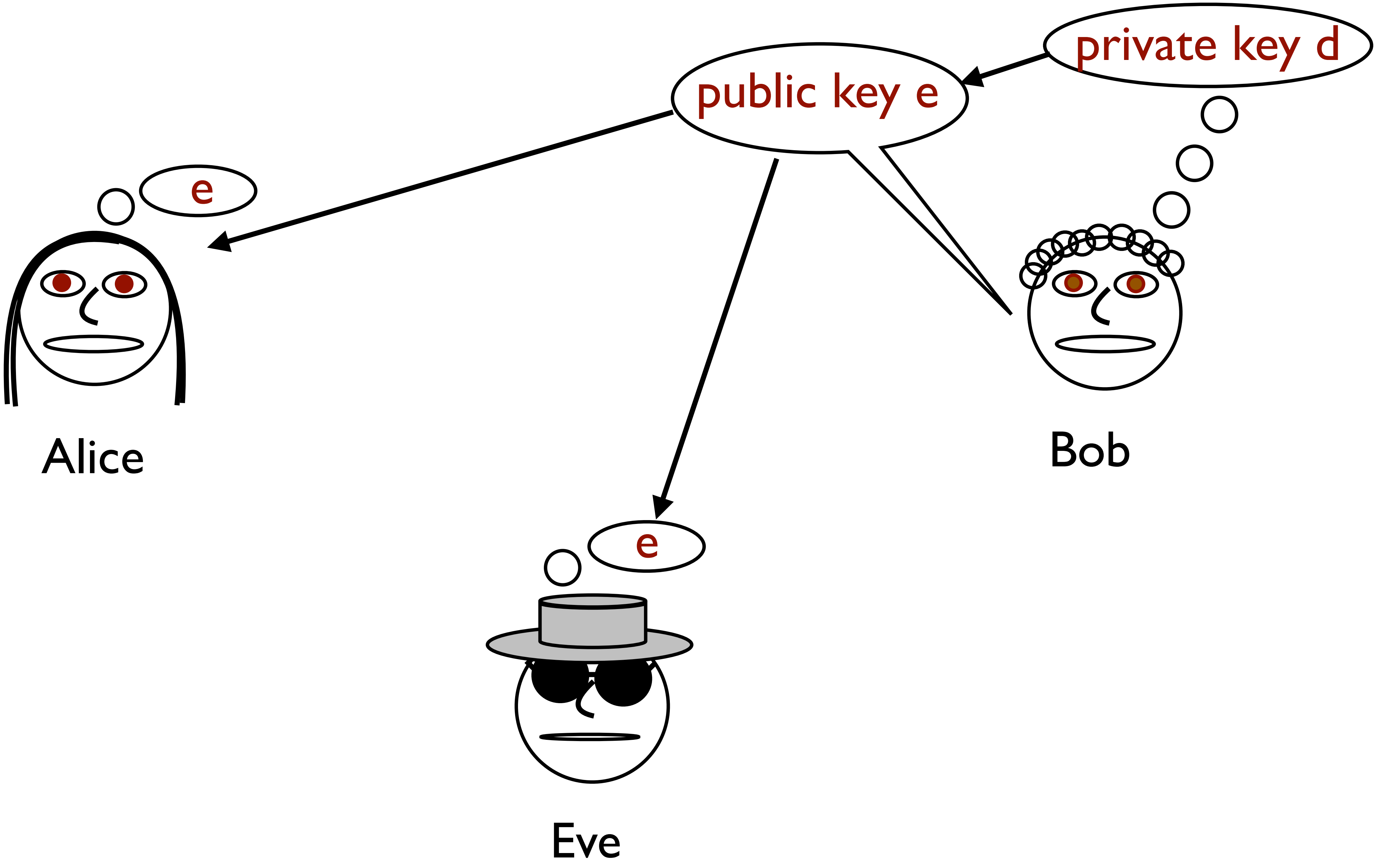
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

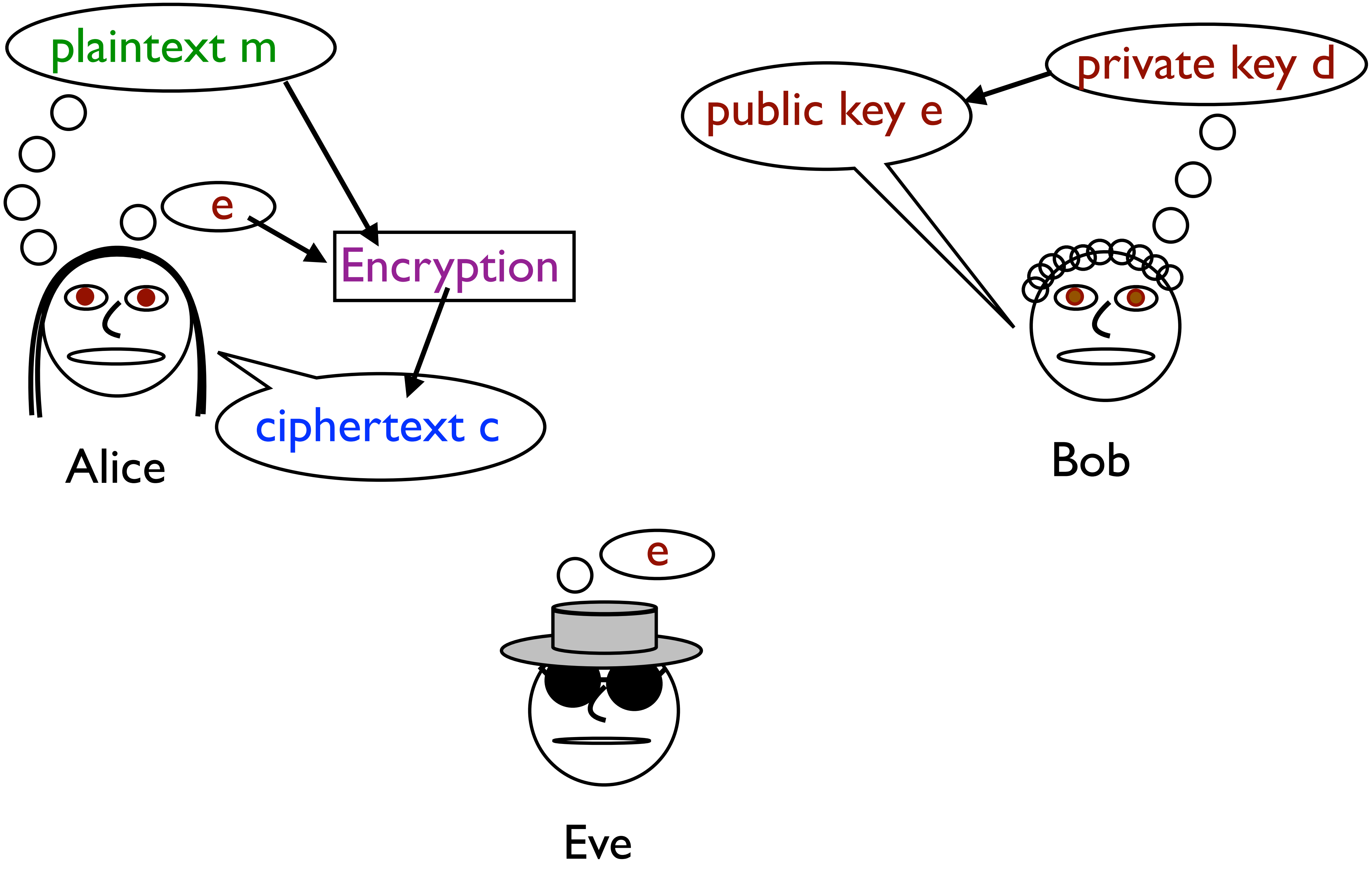
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

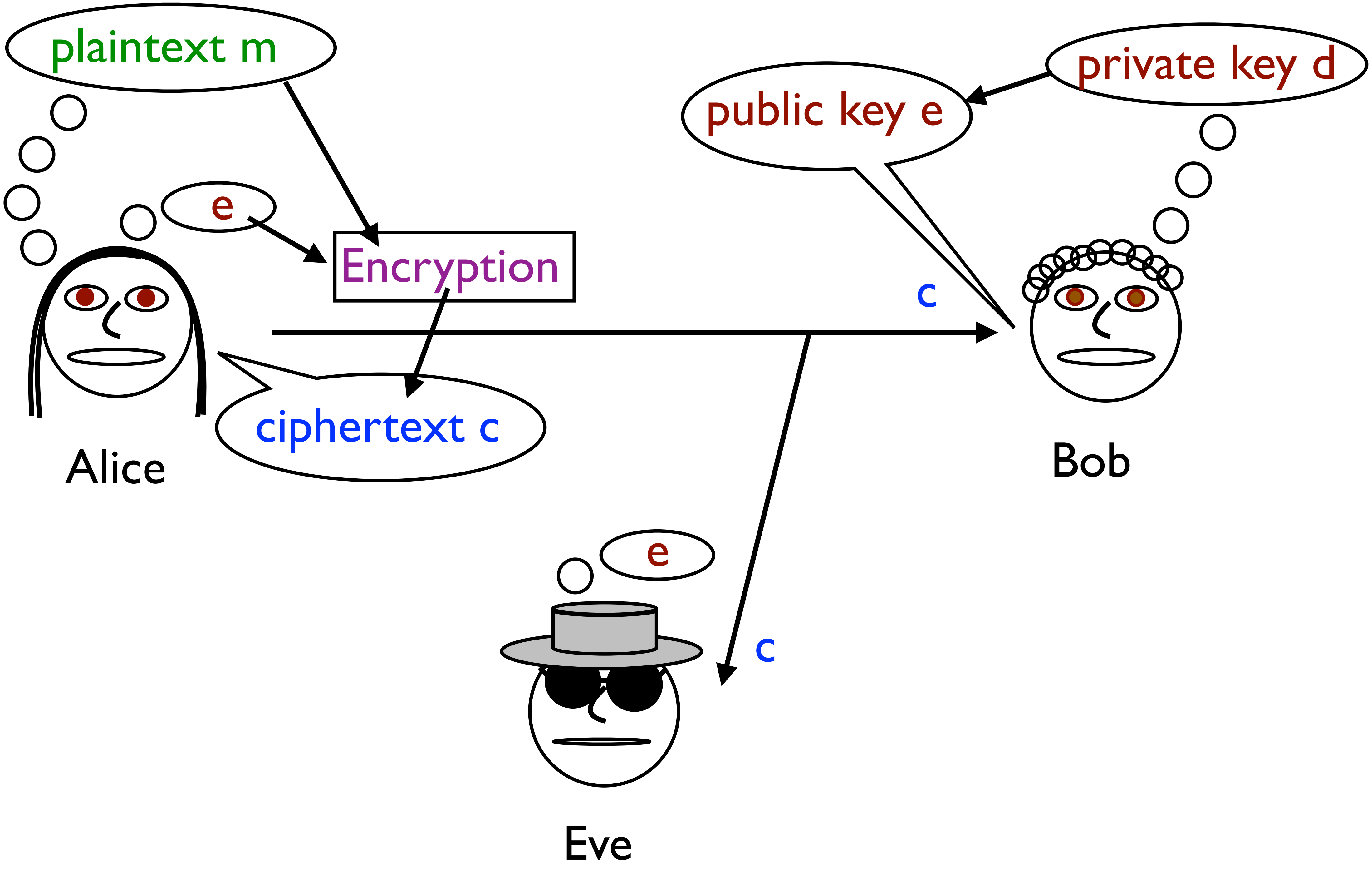
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

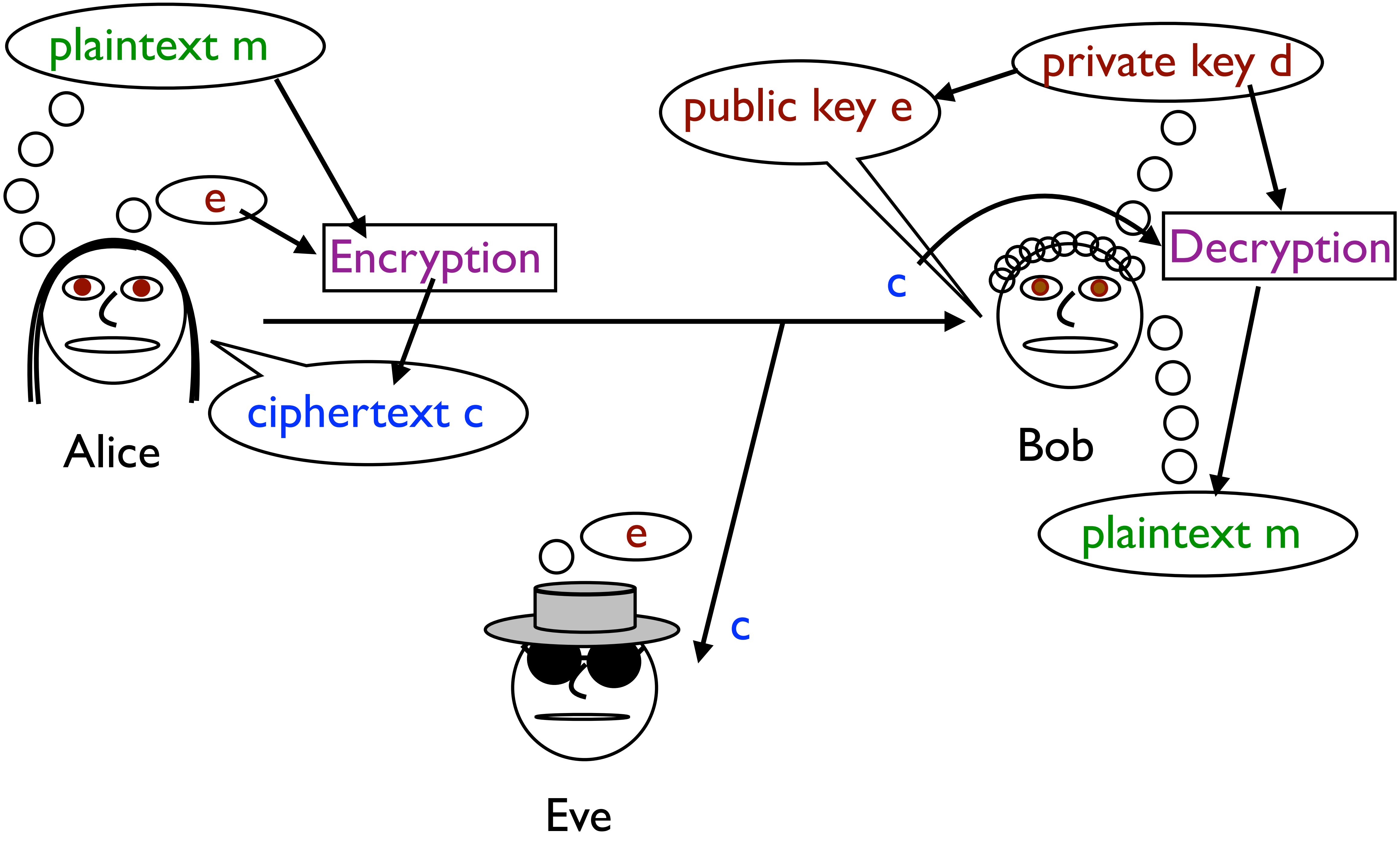
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

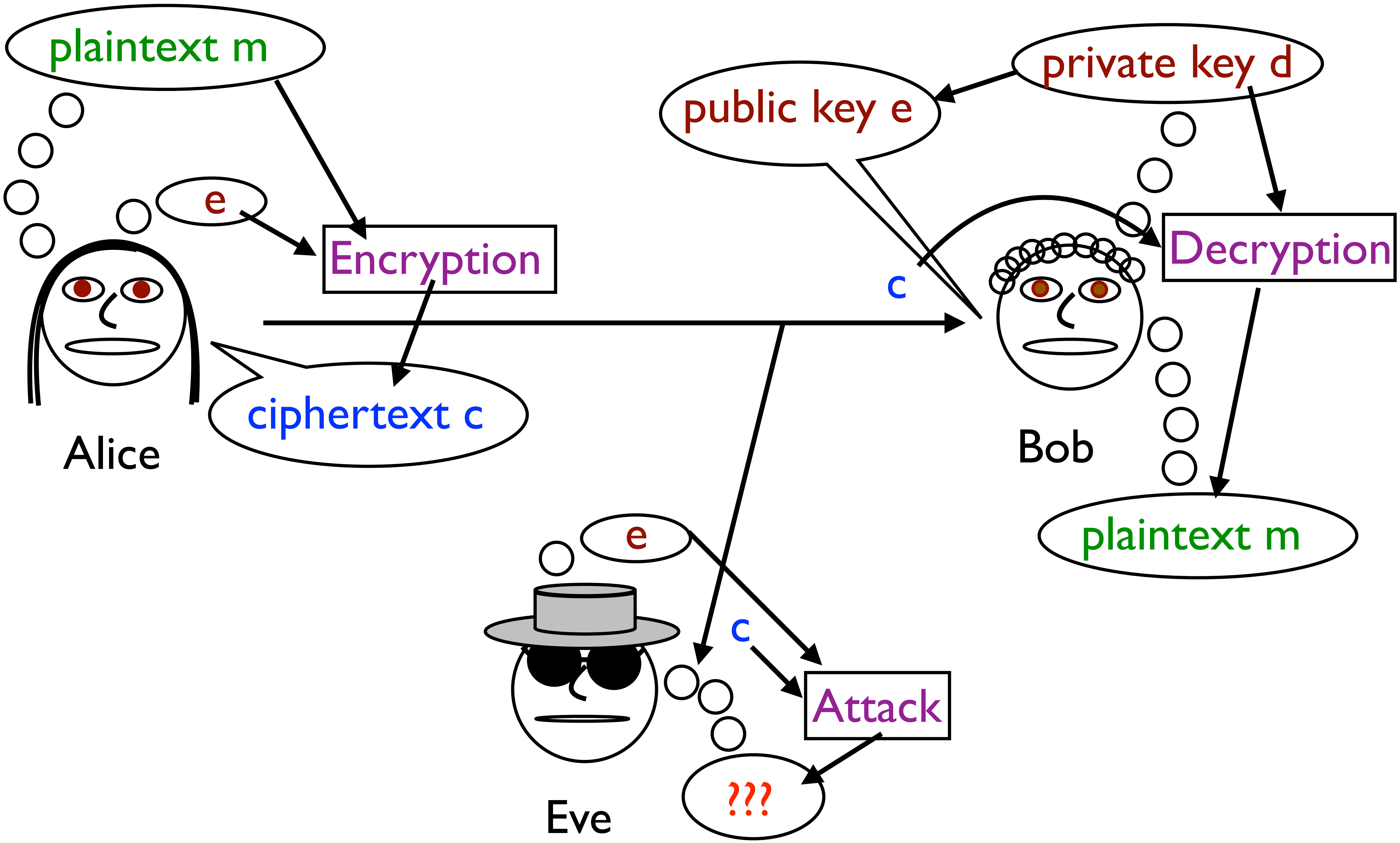
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

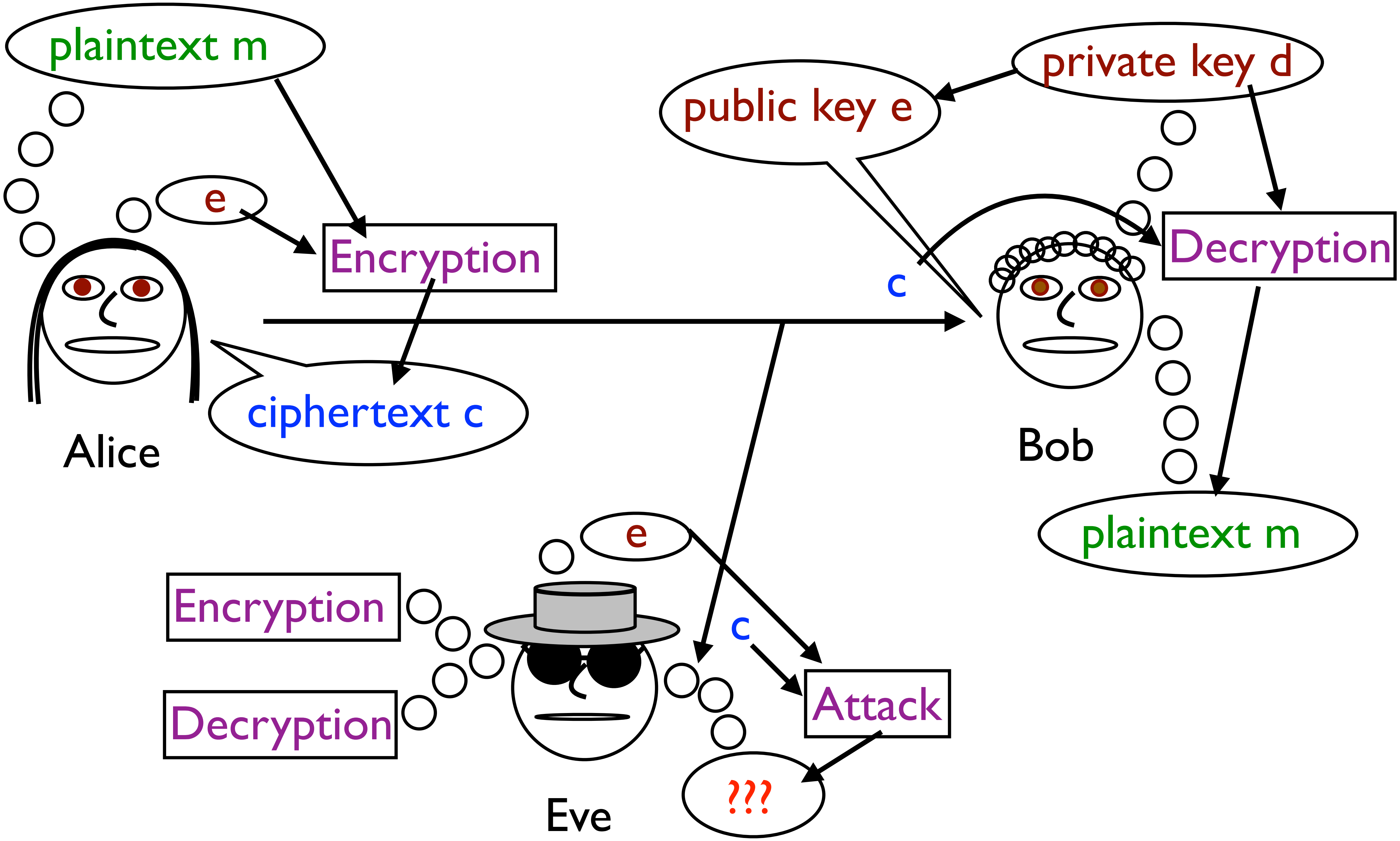
Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

Public Key Encryption



Public-key encryption is an asymmetric protocol.

This class is being recorded

Definition of Public Key Encryption

Definition: A public-key encryption protocol is a set of three probabilistic polynomial-time algorithms (Gen, Enc, Dec).

Gen is the key generation algorithm. It takes as input s, the security parameter, and outputs a public key, private key pair $(e,d) \in \{0,1\}^* \times \{0,1\}^*$.

Enc is the encryption algorithm. It takes as input e and a plaintext or message $m \in \{0,1\}^*$ and outputs a ciphertext $c \in \{0,1\}^*$.

Dec is the decryption algorithm. It takes as input d and c and outputs some $m' \in \{0,1\}^*$.

The encryption protocol is correct if

$$Dec(d, Enc(e,m)) = m$$

Note: Gen here is much more complex than for private-key encryption and doesn't just generate random bit strings.

This class is being recorded

El Gamal as a Public Key Scheme

Gen: The group G and base g have been pre-determined. Gen chooses a random $b \in \{0,\ldots,\mathrm{ord}(g)-1\}$, which becomes the private key. I.e., d = b. The public key is $e = g^b$.

Enc: Given message m and public key e. Choose random $a \in \{0,\ldots,\mathrm{ord}(g)-1\}$. The ciphertext is $c = (g^a, m \cdot e^a)$.

Dec: Given $c = (A, x)$ and d. The decrypted message is $m' = x/A^d$.

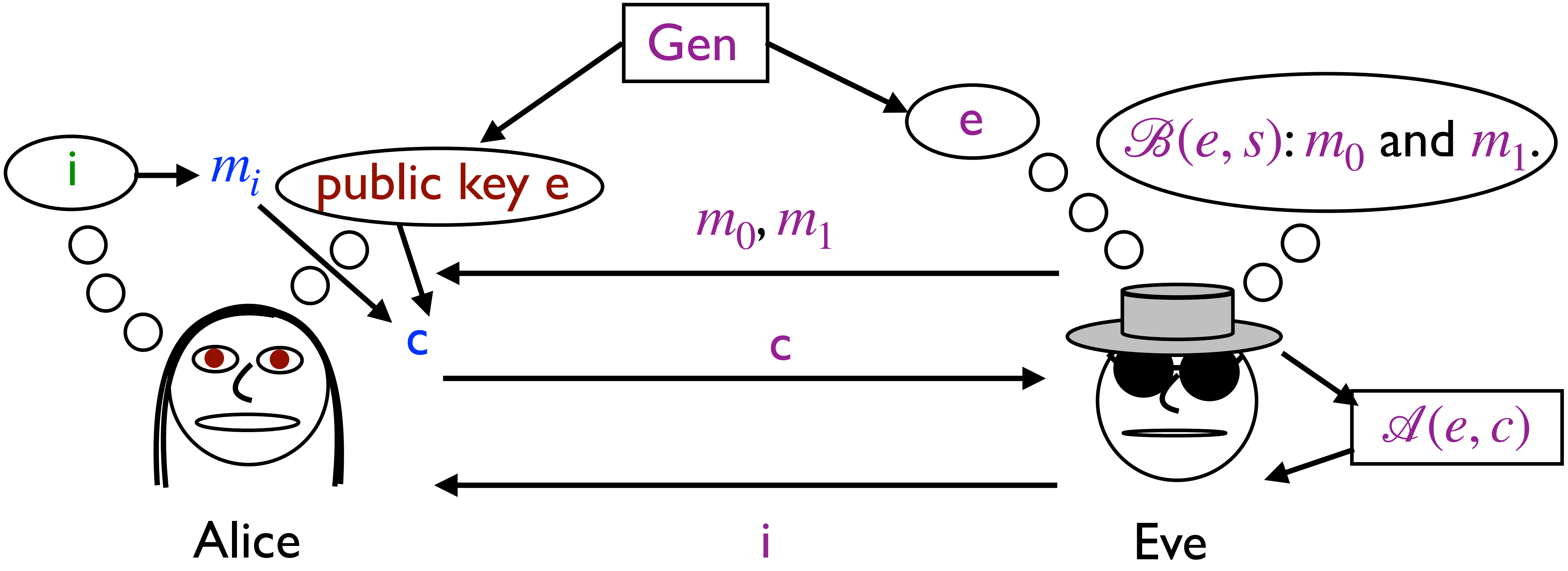
G and g can instead be generated using Gen and made part of the public key.

(All calculations here are done within the group G.)

This class is being recorded

Security of Public Key Encryption

The definition of EAV security for public key encryption is very similar to EAV security for private key encryption.



The main difference is that now Eve is given the public key e. She can use e to choose the messages she wants to use as challenges as well as in her attack to decipher the message.

This class is being recorded

Public Key EAV Security

Definition: A public-key encryption protocol (Gen, Enc, Dec) with security parameter s has indistinguishable encryptions in the presence of an eavesdropper (is EAV-secure) if, for any pair of messages m_0 and m_1 chosen by the adversary (using efficient algorithm $\mathscr{B}(e,s)$) and for any efficient attack $\mathscr{A}(e,c)$,

$$|\Pr_{(e,d)}(\mathscr{A}(e, Enc(e, m_0)) = 1) - \Pr_{(e,d)}(\mathscr{A}(e, Enc(e, m_1)) = 1)| \leq \epsilon(s)$$

for negligible $\epsilon(s)$ and probability taken over valid public key, private key pairs (e,d) and randomness of Enc, $\mathscr{A}$, and $\mathscr{B}$.

Note that d and Dec are not needed in this definition. But they are still important because the protocol cannot be *correct* if they are not chosen right.

This class is being recorded

CPA Security with Public Keys

Suggestions: Does anyone have an idea how to modify this definition to get CPA security?

CPA Security with Public Keys

Suggestions: Does anyone have an idea how to modify this definition to get CPA security?

How about we don't change anything?

CPA Security with Public Keys

Suggestions: Does anyone have an idea how to modify this definition to get CPA security?

How about we don't change anything?

> For private key cryptography, to get CPA security, we modified the definition of EAV security by giving Eve access to an oracle that performed Enc(k,x). This enables Eve to create plaintext - ciphertext pairs as she wishes.

CPA Security with Public Keys

Suggestions: Does anyone have an idea how to modify this definition to get CPA security?

How about we don't change anything?

For private key cryptography, to get CPA security, we modified the definition of EAV security by giving Eve access to an oracle that performed Enc(k,x). This enables Eve to create plaintext - ciphertext pairs as she wishes.

But Eve has the public key. She doesn't need an oracle — she can already create plaintext - ciphertext pairs using the public key.

This class is being recorded

CPA Security with Public Keys

Suggestions: Does anyone have an idea how to modify this definition to get CPA security?

How about we don't change anything?

For private key cryptography, to get CPA security, we modified the definition of EAV security by giving Eve access to an oracle that performed Enc(k,x). This enables Eve to create plaintext - ciphertext pairs as she wishes.

But Eve has the public key. She doesn't need an oracle — she can already create plaintext - ciphertext pairs using the public key.

For public key encryption, EAV security is the same as CPA security!

This class is being recorded

Key Encapsulation

Both of the techniques I mentioned so far for encryption (El Gamal and using the key directly in the pseudo one-time pad) suffer from being very slow and inefficient.

> But what if we use the key generated by Diffie-Hellman for a symmetric cryptosystem? Then we only have to use Diffie-Hellman for the initial set-up and the rest of the message can be sent with the more efficient symmetric encryption protocol.

We want to keep the non-interactive part of El Gamal, but we don't need (or want) to encrypt a message with it. We do, however, need to convert the key from a group element to a bit string using a key derivation function.

> We will get a procedure to create a shared random encrypted key using a public key. This is known as a key encapsulation mechanism (KEM).

KEM with Diffie-Hellman/El Gamal

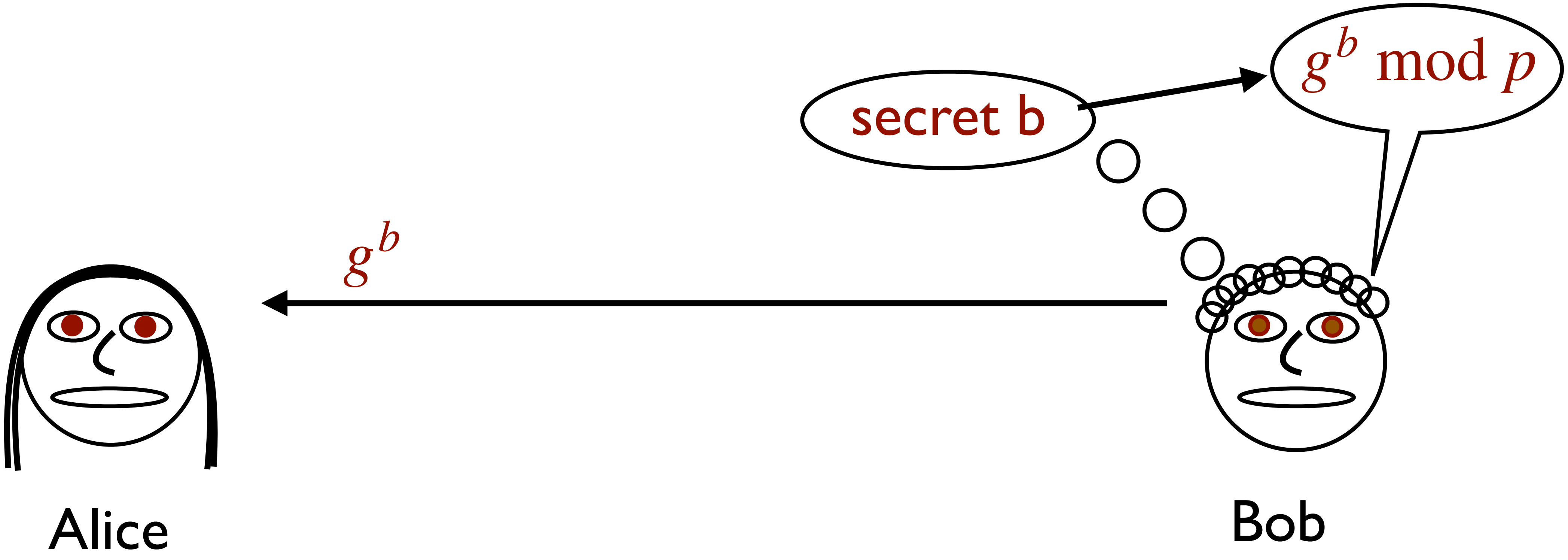
Fixed as part of the protocol: p, g, H. Bob's public key is g^b.

secret b
Alice
Bob

This class is being recorded

KEM with Diffie-Hellman/El Gamal

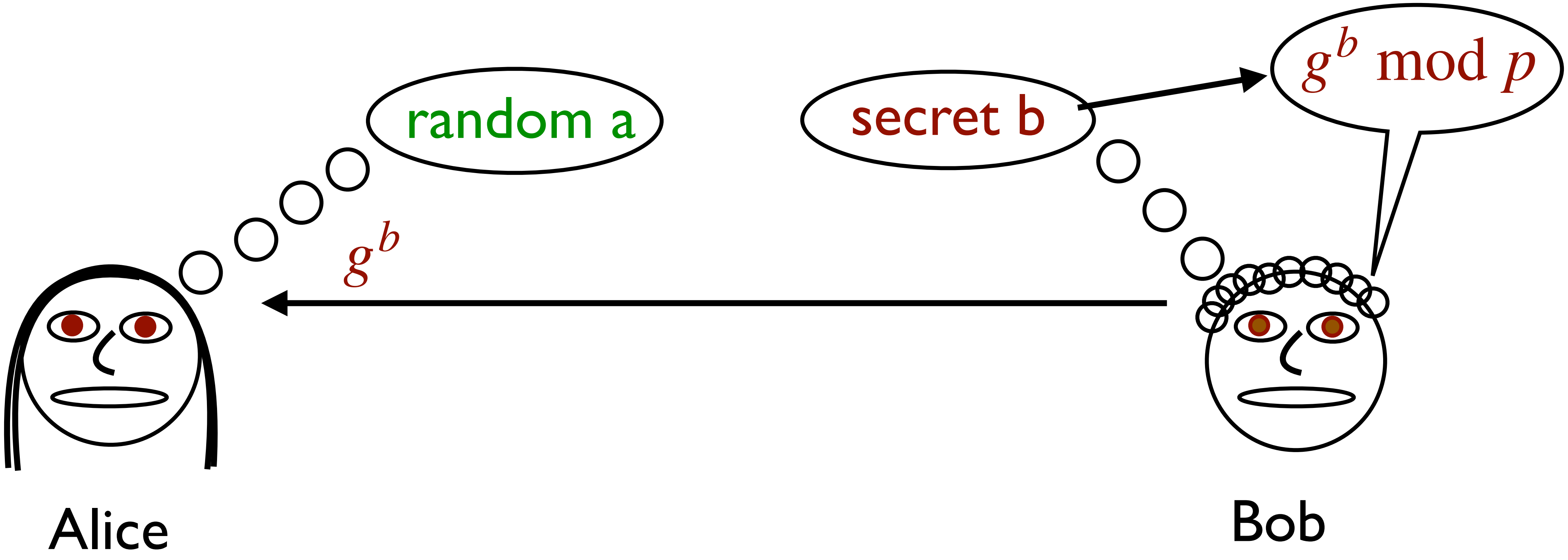
Fixed as part of the protocol: p, g, H. Bob's public key is g^b.



This class is being recorded

KEM with Diffie-Hellman/El Gamal

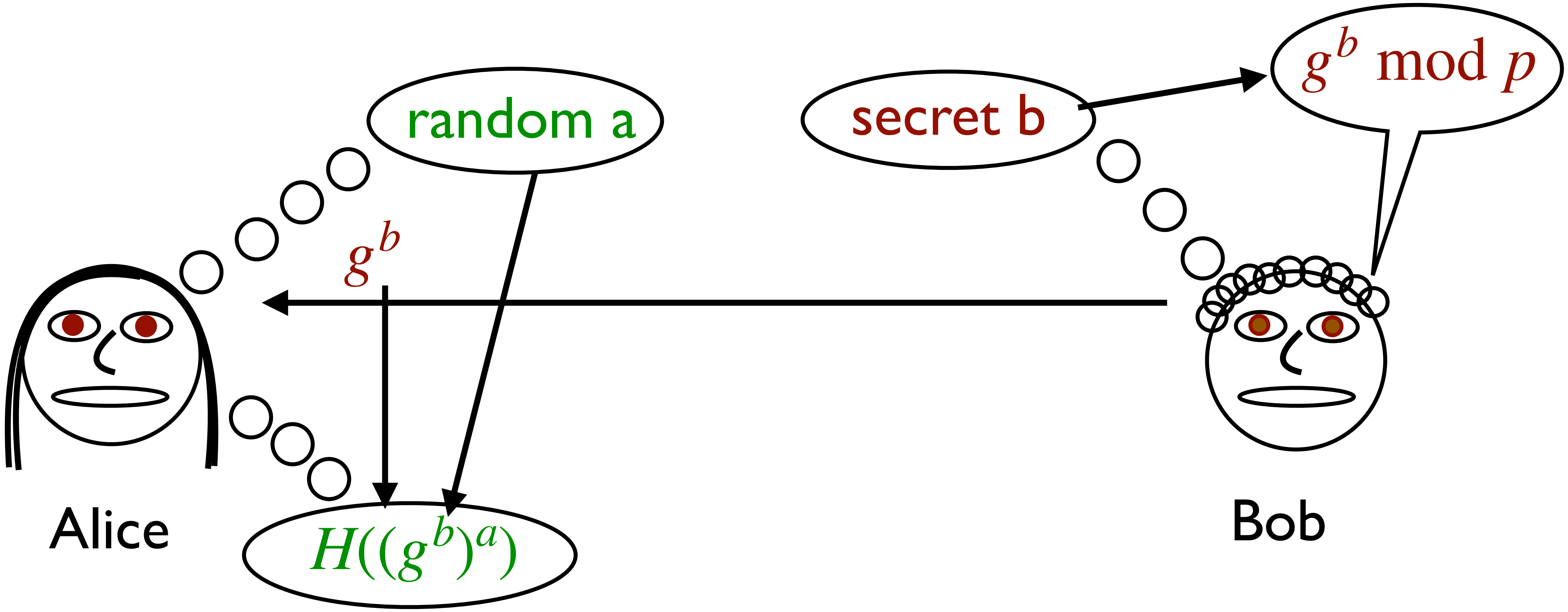
Fixed as part of the protocol: p, g, H. Bob's public key is g^b.



This class is being recorded

KEM with Diffie-Hellman/El Gamal

Fixed as part of the protocol: p, g, H. Bob's public key is g^b.

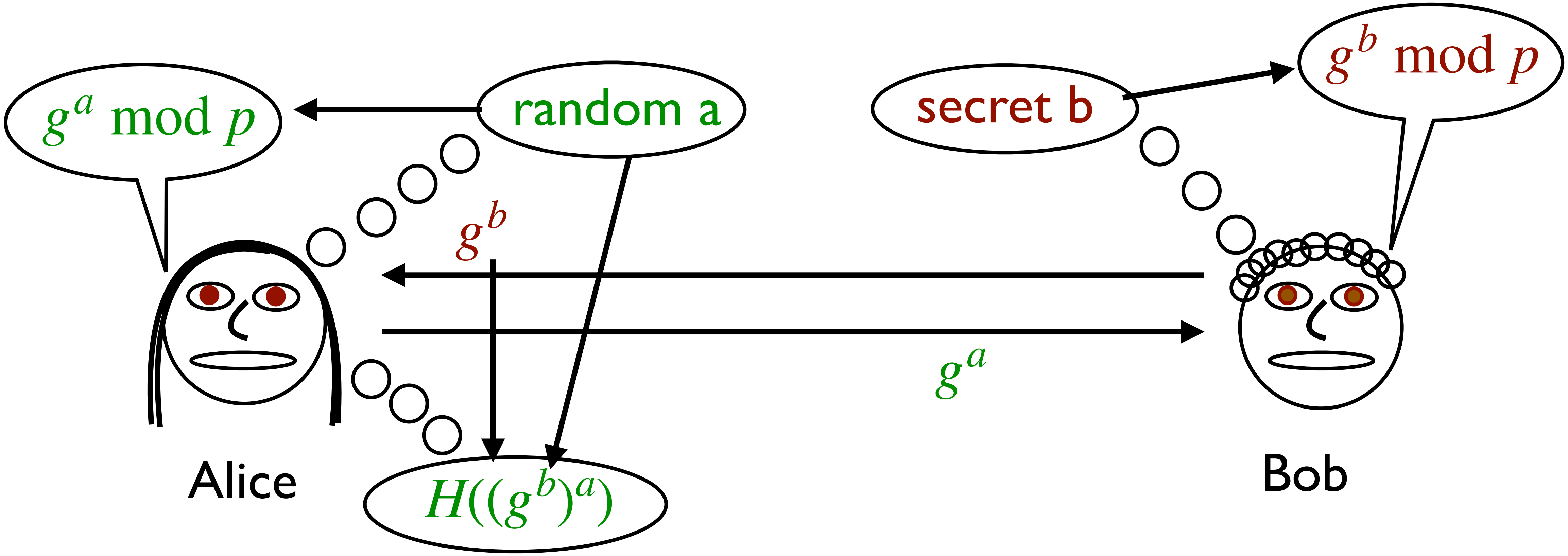


The generated key is $H(g^{ab})$.

This class is being recorded

KEM with Diffie-Hellman/El Gamal

Fixed as part of the protocol: p, g, H. Bob's public key is g^b.

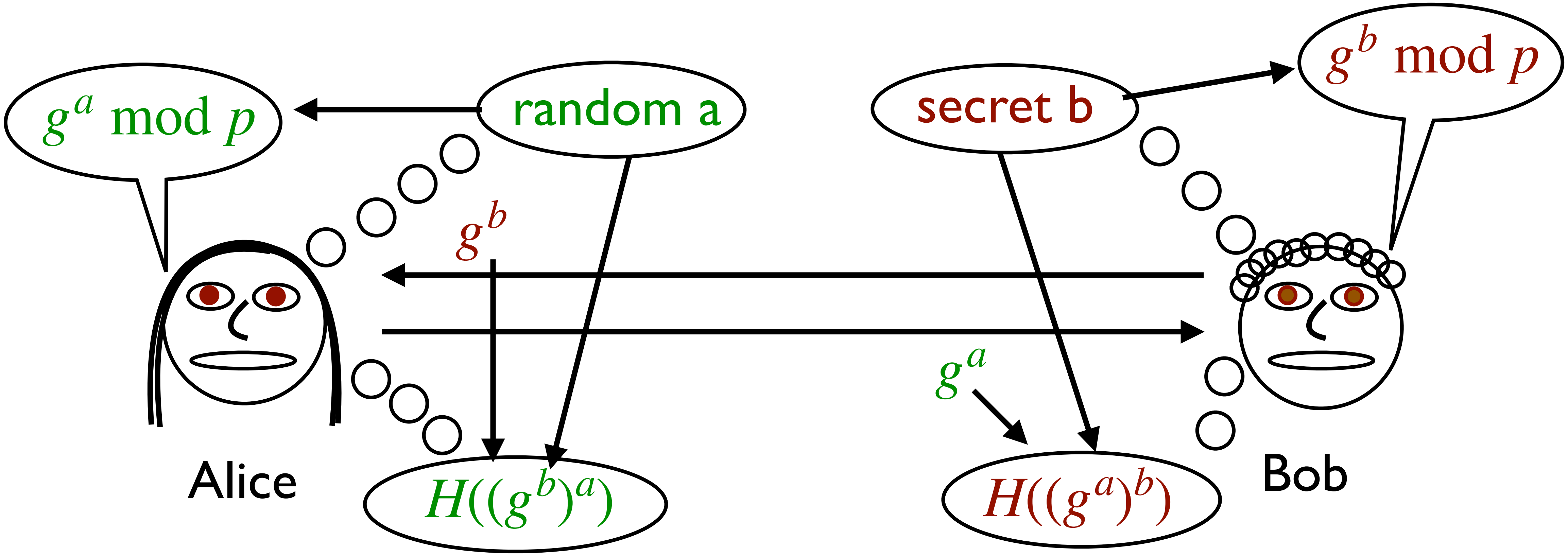


The generated key is $H(g^{ab})$.

This class is being recorded

KEM with Diffie-Hellman/El Gamal

Fixed as part of the protocol: p, g, H. Bob's public key is g^b.



The generated key is $H(g^{ab})$.

This class is being recorded

Definition of Key Encapsulation

Definition: A public-key encryption protocol is a set of three probabilistic polynomial-time algorithms (Gen, Enc, Dec).

Gen is the key generation algorithm. It takes as input s, the security parameter, and outputs a public key, private key pair $(e, d) \in \{0,1\}^* \times \{0,1\}^*$.

Encaps is the encapsulation algorithm. It takes as input e (only) and outputs a ciphertext $c \in \{0,1\}^*$ and a key $k \in \{0,1\}^{\ell(s)}$, for some function $\ell(s)$ (the key length).

Decaps is the decapsulation algorithm. It takes as input d and c and outputs some $k' \in \{0,1\}^{\ell(s)}$.

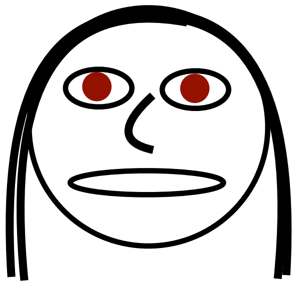
If Encaps(e) outputs ciphertext c and key k, then we require that Decaps(d,c) outputs k' = k.

This class is being recorded

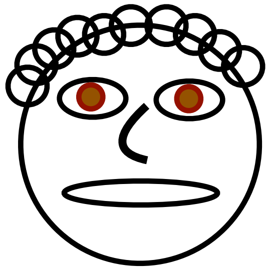
KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

> That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)



Alice

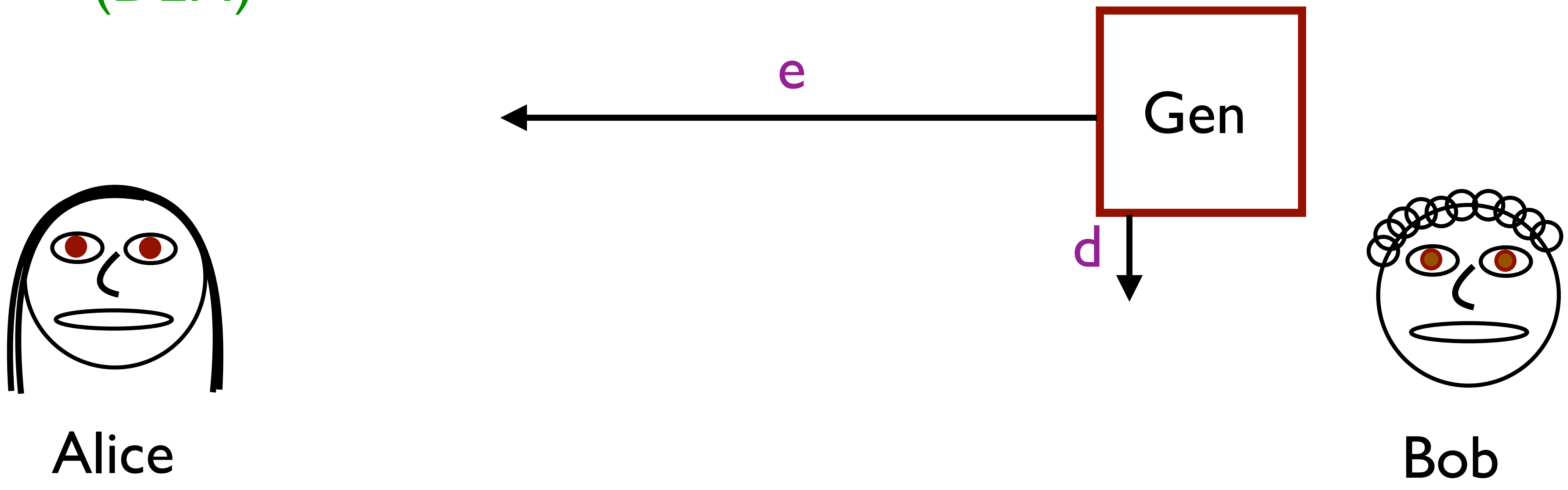


Bob

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)

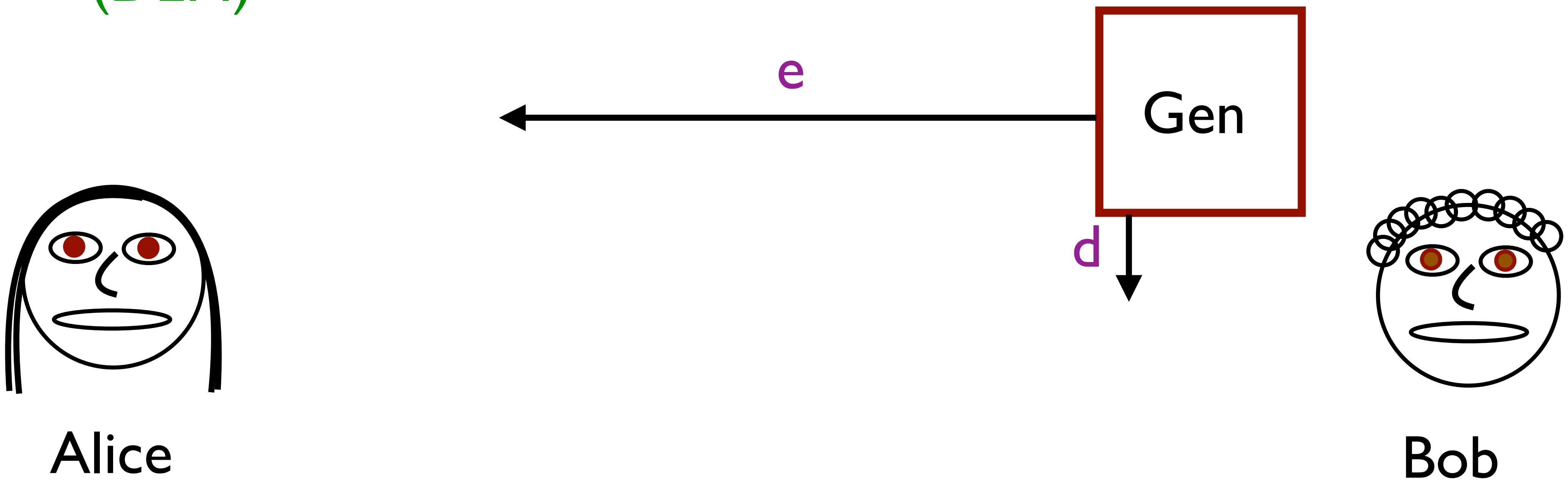


This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)



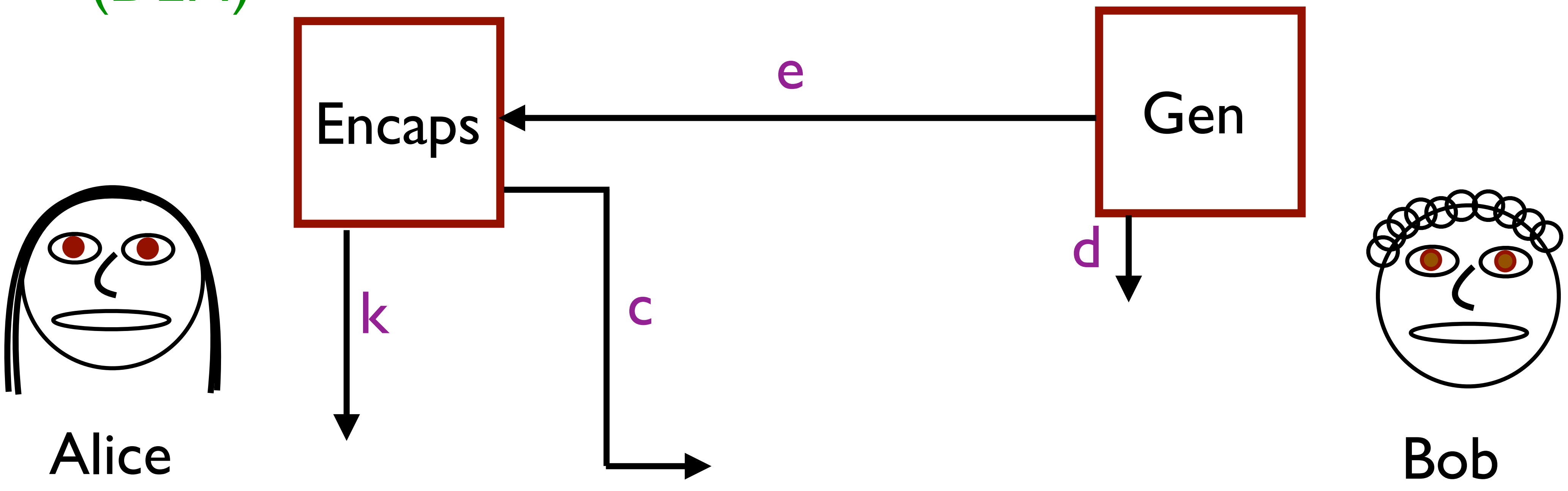
message m

This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)



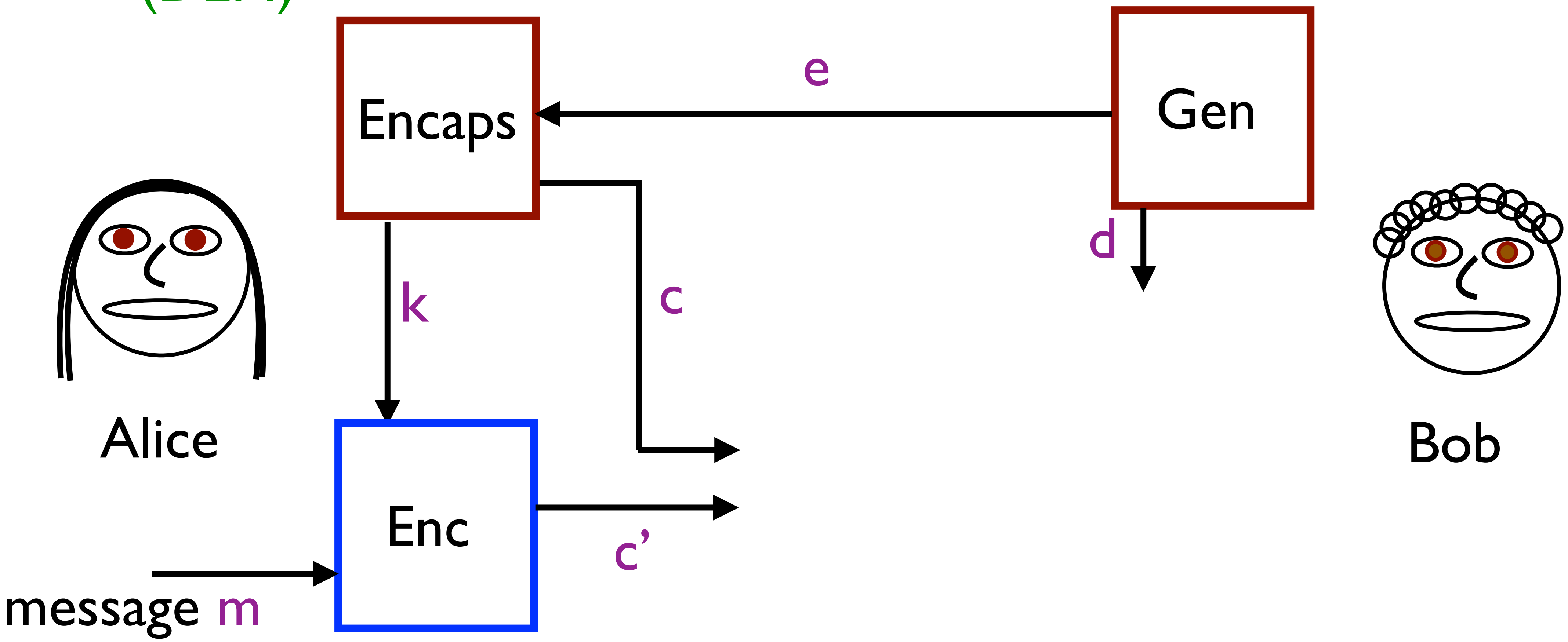
message m

This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)

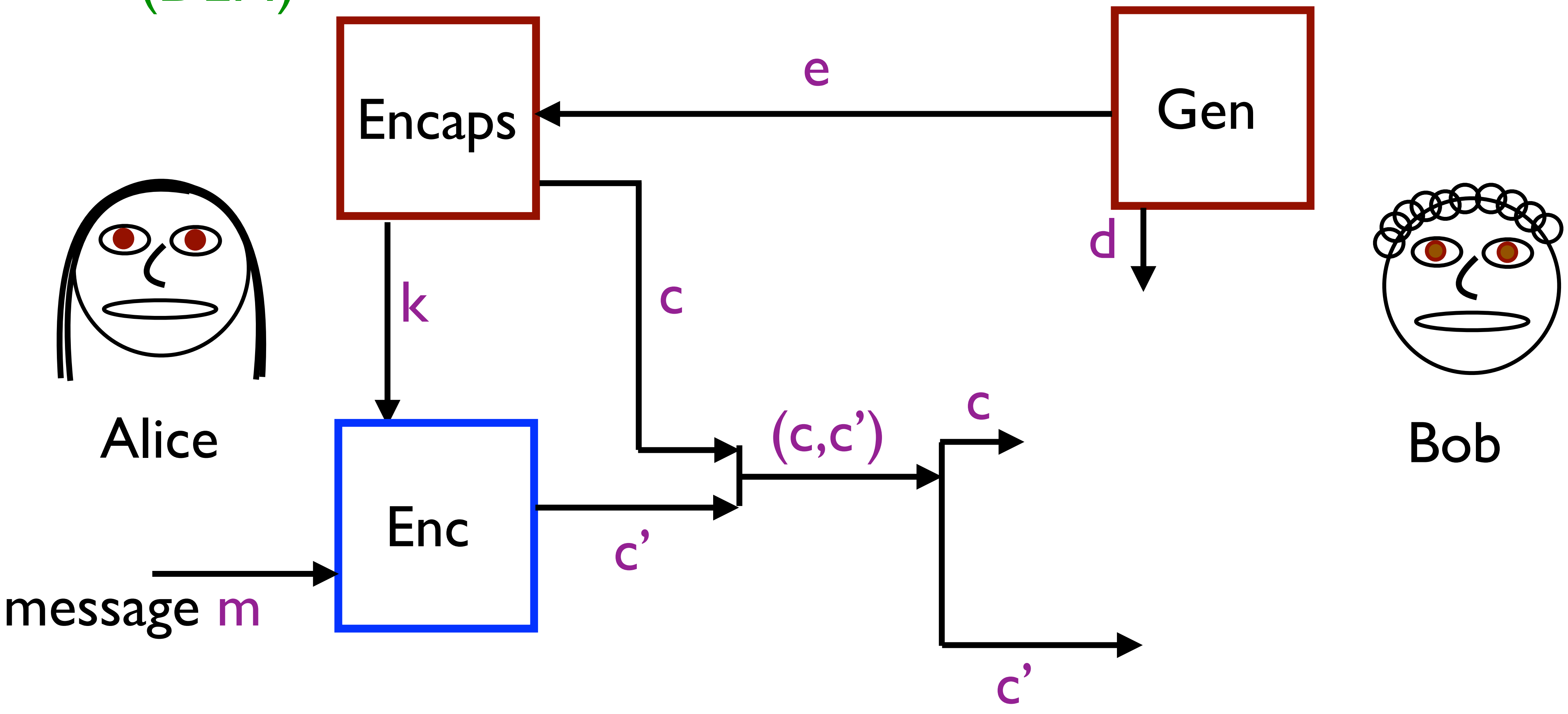


This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)

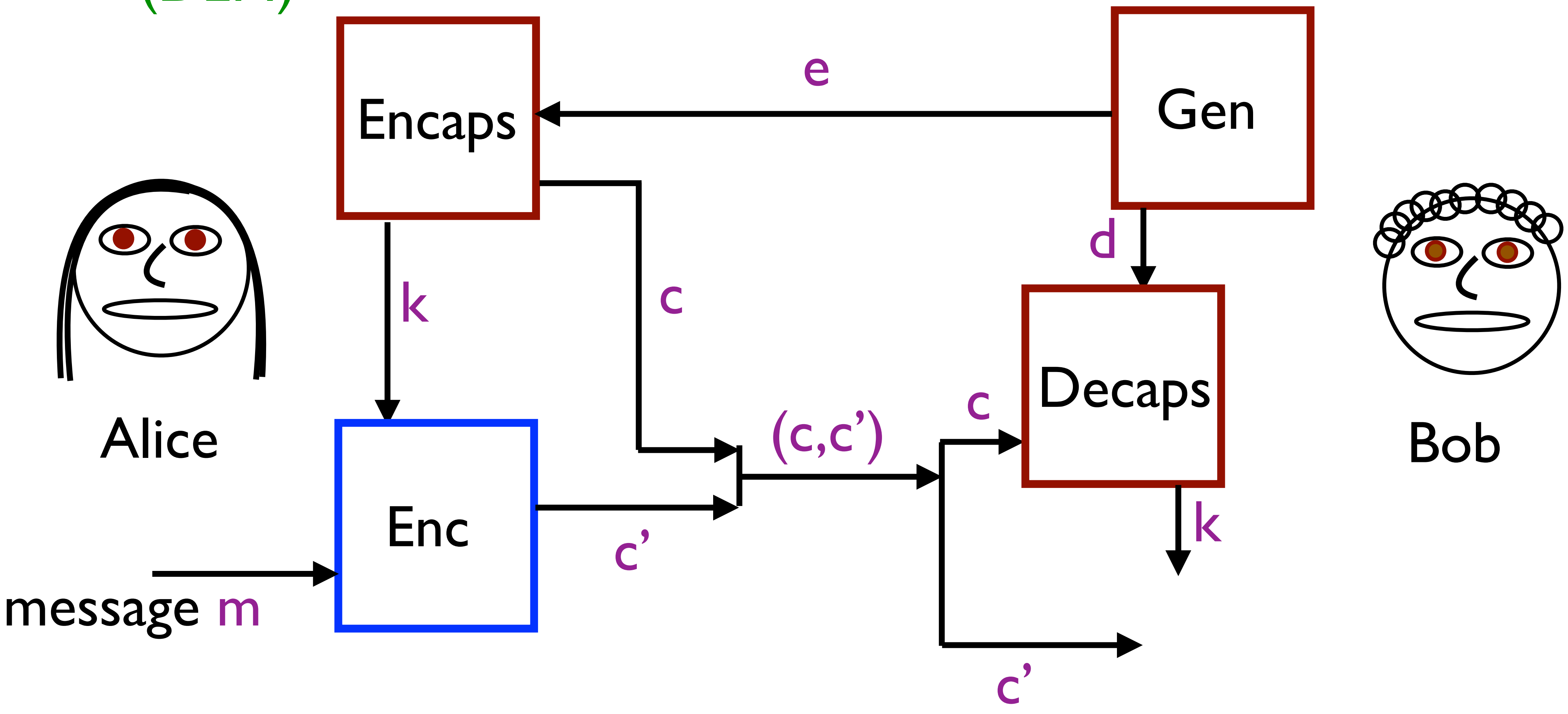


This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)

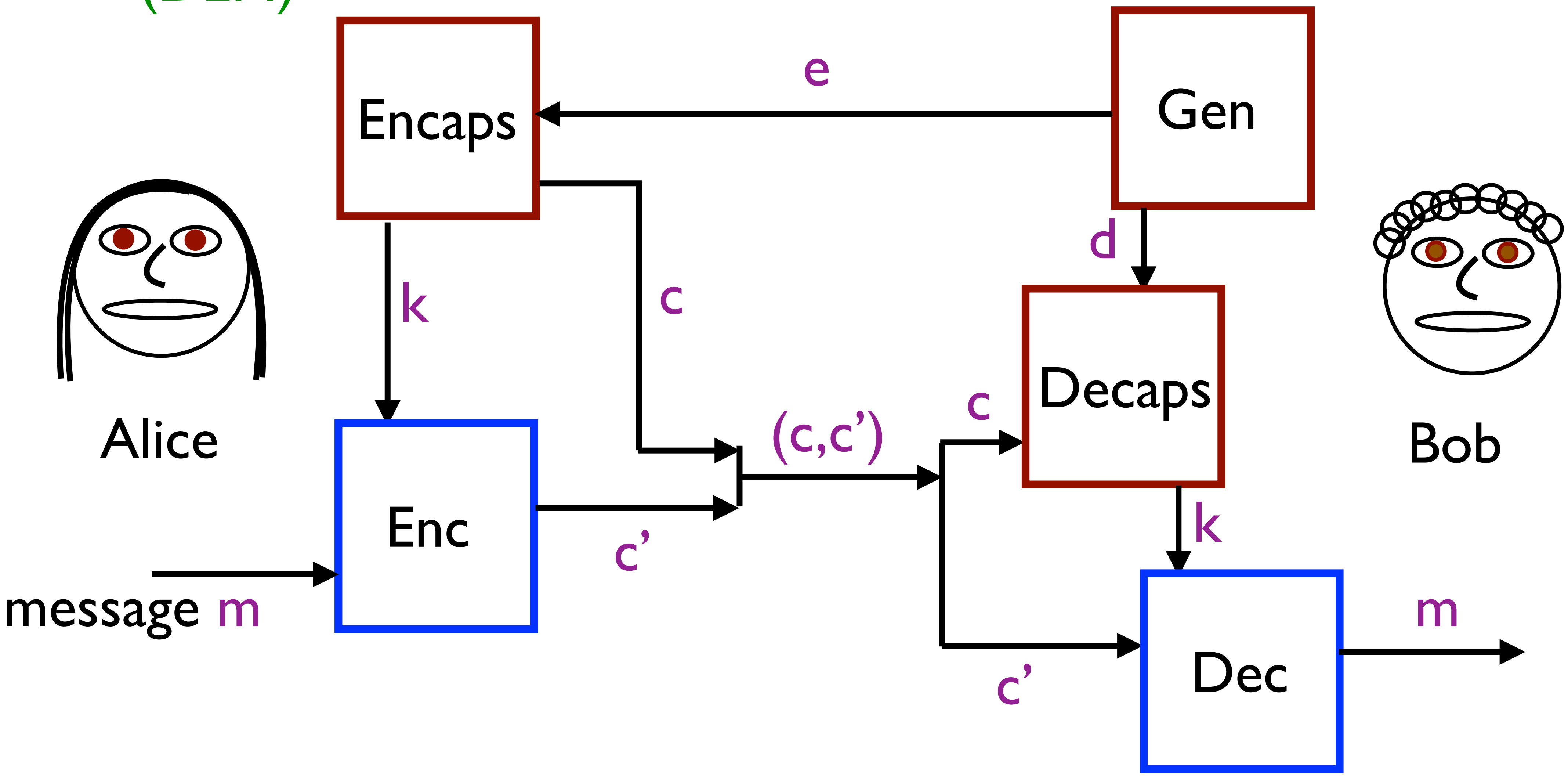


This class is being recorded

KEM/DEM

A KEM creates and sends a random key string to someone whose public key you have.

That key then becomes the key for a private-key encryption system, which is here called a data-encapsulation mechanism (DEM)



This class is being recorded

Why KEM/DEM?

Public key cryptosystems are much slower, more inefficient in terms of communication, and less secure than private key cryptosystems.

Private key cryptosystems require a previously shared key, making it hard to communicate with someone new and making key management a headache with many people.

KEM/DEM combines the efficiency of private key systems with the convenience of public key systems!

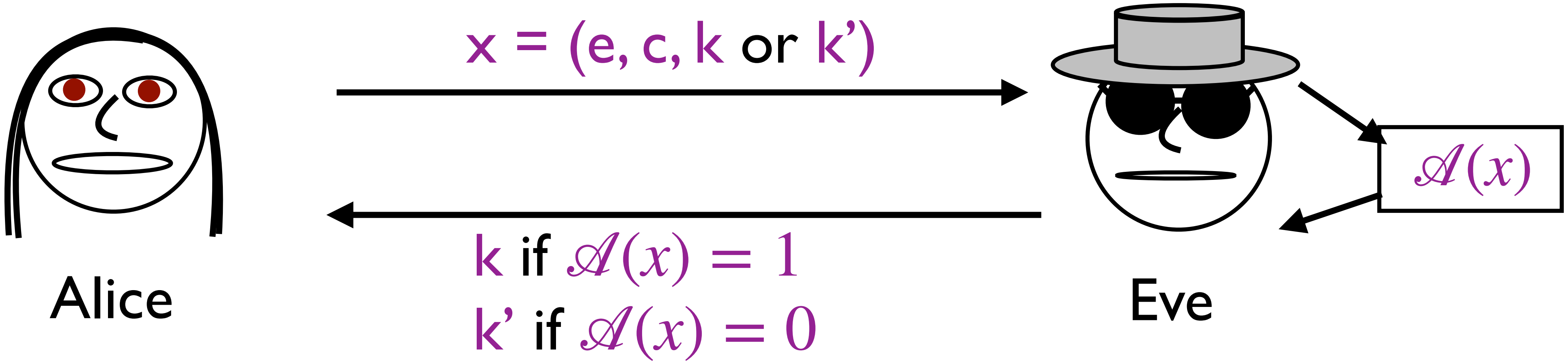
TLS uses KEM/DEM. TLS is used in https, so is extremely common.

CPA Security for KEM

Definition: Consider a KEM (Gen, Encaps, Decaps). Suppose Gen produces public key e and Encaps(e) produces ciphertext c and key k. Let k' be a uniformly randomly generated key. Then the KEM is CPA secure if for all attacks $\mathscr{A}$ with a 1-bit output and taking as inputs e, c, and either k or k',

$$|\Pr(\mathscr{A}(e,c,k)=1) - \Pr(\mathscr{A}(e,c,k')=1)| \leq \epsilon(s)$$

with $\epsilon(s)$ negligible and the probabilities averaged over k' and over the randomness of $\mathscr{A}$, Gen, and Encaps.



x = (e, c, k or k')

$\mathscr{A}(x)$

k if $\mathscr{A}(x) = 1$

k' if $\mathscr{A}(x) = 0$

Alice

Eve

This class is being recorded

Why is This CPA Security?

The term "CPA security" implies that it should be secure against chosen-plaintext attacks.

> Why don't we need to give Eve an oracle to let her choose plaintexts?

Why is This CPA Security?

The term "CPA security" implies that it should be secure against chosen-plaintext attacks.

> Why don't we need to give Eve an oracle to let her choose plaintexts?

Two reasons:

- Eve has access to e, so once again she can run Encaps if she wants.
- There is no plaintext here to choose. Encaps only takes e as input, no message.

Security of KEM/DEM

Theorem: If we have a KEM/DEM encryption scheme in which the KEM protocol is CPA secure and the DEM protocol is EAV secure, then the full KEM/DEM scheme is CPA secure.

Note: DEM only needs to be EAV secure.

Intuition and Proof Sketch:

The key from the CPA secure KEM looks the same to Eve as a random bit string. Therefore, if it used as the key in an EAV-secure encryption protocol, the protocol behaves the same way as if the key was fully random. That means that Eve cannot distinguish between messages.

Since Eve has access to the public key, this is all that is needed for CPA security of the public key protocol.

Note: Eve can create ciphertexts for the full protocol, but not for the DEM with the specific key used by Alice.

This class is being recorded