CMSC/Math 456: Cryptography (Fall 2022)

Lecture 21

Daniel Gottesman

Administrative

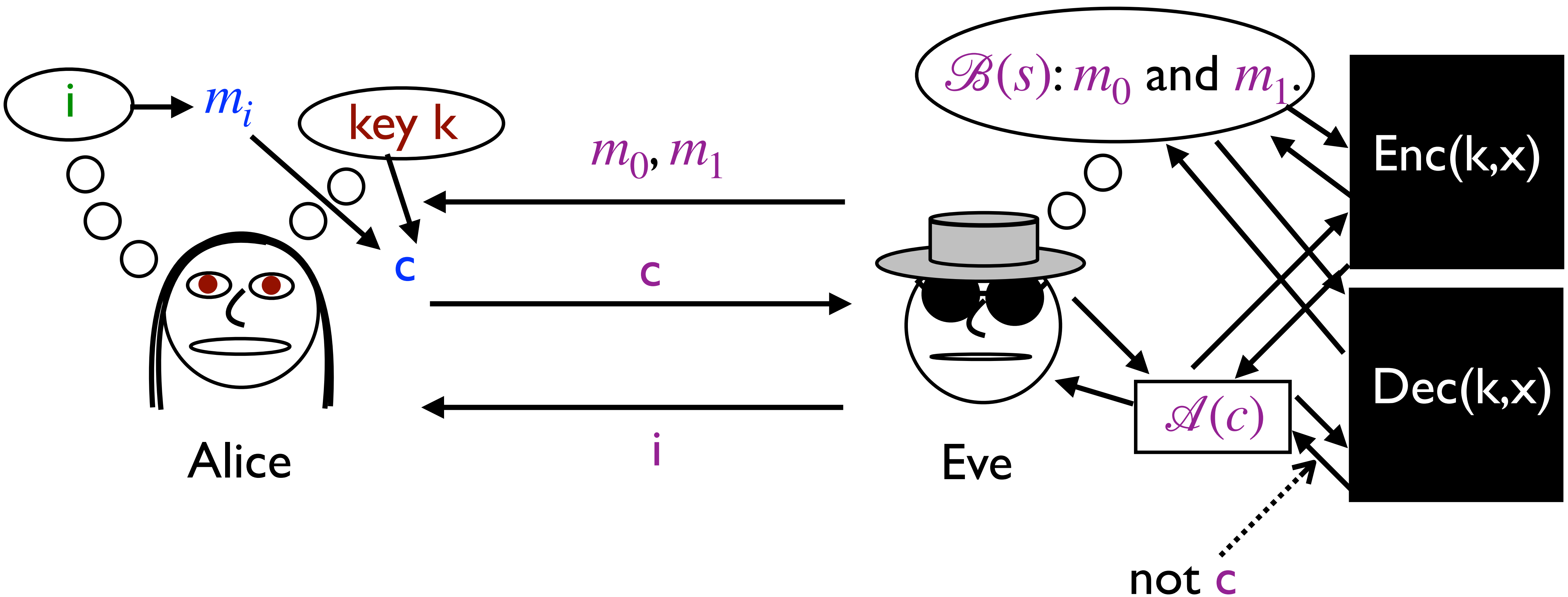
Problem set #6 (a programming assignment) is due tonight at *midnight*.

Problem set #7 will be available tonight.

Problem set #8 will be assigned next Thursday (Nov. 17) but you will have *two weeks* to do it. (Due Dec. 1)

CCA Security

A protocol is CCA-secure if Eve cannot guess the encrypted message even with access to both encryption and decryption oracles.



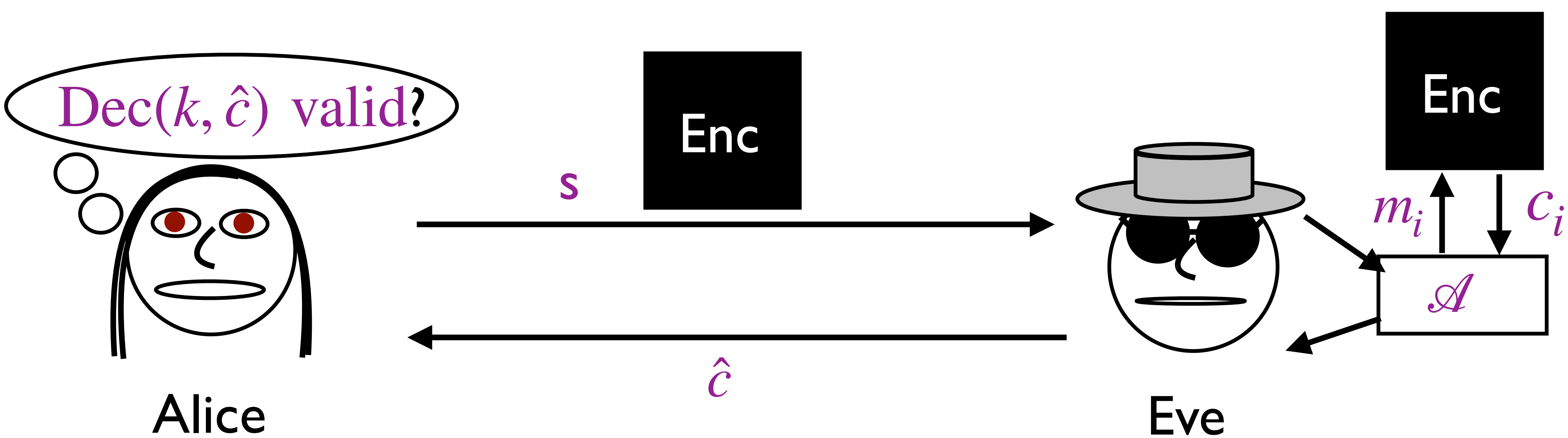
This class is being recorded

Unforgeability

Definition: A encryption protocol (Gen, Enc, Dec) with security parameter s is unforgeable if, for any polynomial-time attack $\mathscr{A}$ with oracle access to $\mathrm{Enc}(k, m)$, where $\mathscr{A}$ outputs $\hat{c}$ with $\hat{m} = \mathrm{Dec}(\hat{c})$ such that $\mathscr{A}$ never queried the oracle for $m = \hat{m}$,

$$\Pr(\hat{c} \text{ is valid}) \leq \epsilon(s)$$

where $\epsilon(s)$ is a negligible function and the probability is averaged over k generated by Gen and the randomness used in any of the functions.



This class is being recorded

Authenticated Encryption

Definition: A private-key encryption scheme provides authenticated encryption if it is CCA-secure and unforgeable.

Theorem: If (Enc, Dec) is a CPA-secure encryption scheme and (Mac, Vrfy) is a strongly secure MAC, then the following encryption scheme is an authenticated encryption protocol:

> Enc': Given message m and keys k and k', the ciphertext is (Enc(k,m), Mac(k',Enc(k,m))).
>
> Dec': Given ciphertext (c,t) and keys (k,k'), output Invalid if $\mathrm{Vrfy}(k', c, t)$ is invalid. Otherwise decrypt c to m = Dec(k,c) and output m.

Strongly Secure MAC

Definition: A MAC is strongly secure if Eve cannot generate (except with negligible probability) a valid message tag pair (m,t) such that if m was queried to the MAC oracle, it did not return the tag t.

That is, Eve cannot forge a new message and also cannot forge a new tag on a message she has seen.

Question: Why is this needed to get authenticated encryption?

Strongly Secure MAC

Definition: A MAC is strongly secure if Eve cannot generate (except with negligible probability) a valid message tag pair (m,t) such that if m was queried to the MAC oracle, it did not return the tag t.

That is, Eve cannot forge a new message and also cannot forge a new tag on a message she has seen.

Question: Why is this needed to get authenticated encryption?

> Otherwise Eve can see the ciphertext (c,t) and forge a new tag t'. Then (c,t') is a new ciphertext, so Eve can query the decryption oracle and receive the message being encrypted. The protocol would then not be CCA-secure.

Security of the Protocol

Why does the encrypt-then-authenticate strategy produce a secure authenticated encryption scheme?

- The protocol is certainly unforgeable because any new (not produced by the Enc oracle) ciphertext an adversary might produce will be invalid for the MAC (since the MAC is strongly secure)
- Consider an attack $\mathscr{A}$. It may make queries to the decryption oracle, but all of them (except with negligible probability or any ciphertexts produced by Enc) will be invalid.
- Therefore any Dec queries are useless to $\mathscr{A}$ and can be easily simulated by Eve herself.
- Thus, the attack $\mathscr{A}$ could be performed by a CPA adversary with access only to the Enc oracle.
- Since the encryption protocol is CPA secure, Eve cannot distinguish messages except with negligible probability.

CCA Security for Public Key Protocols

Since message authentication is a private-key protocol, we can't combine it with a public-key protocol straightforwardly. In particular, the notion of authenticated encryption needs some modification. To understand what should replace it, we will need to discuss digital signatures, the next topic.

But for now, let us focus only on CCA security of public key protocols.

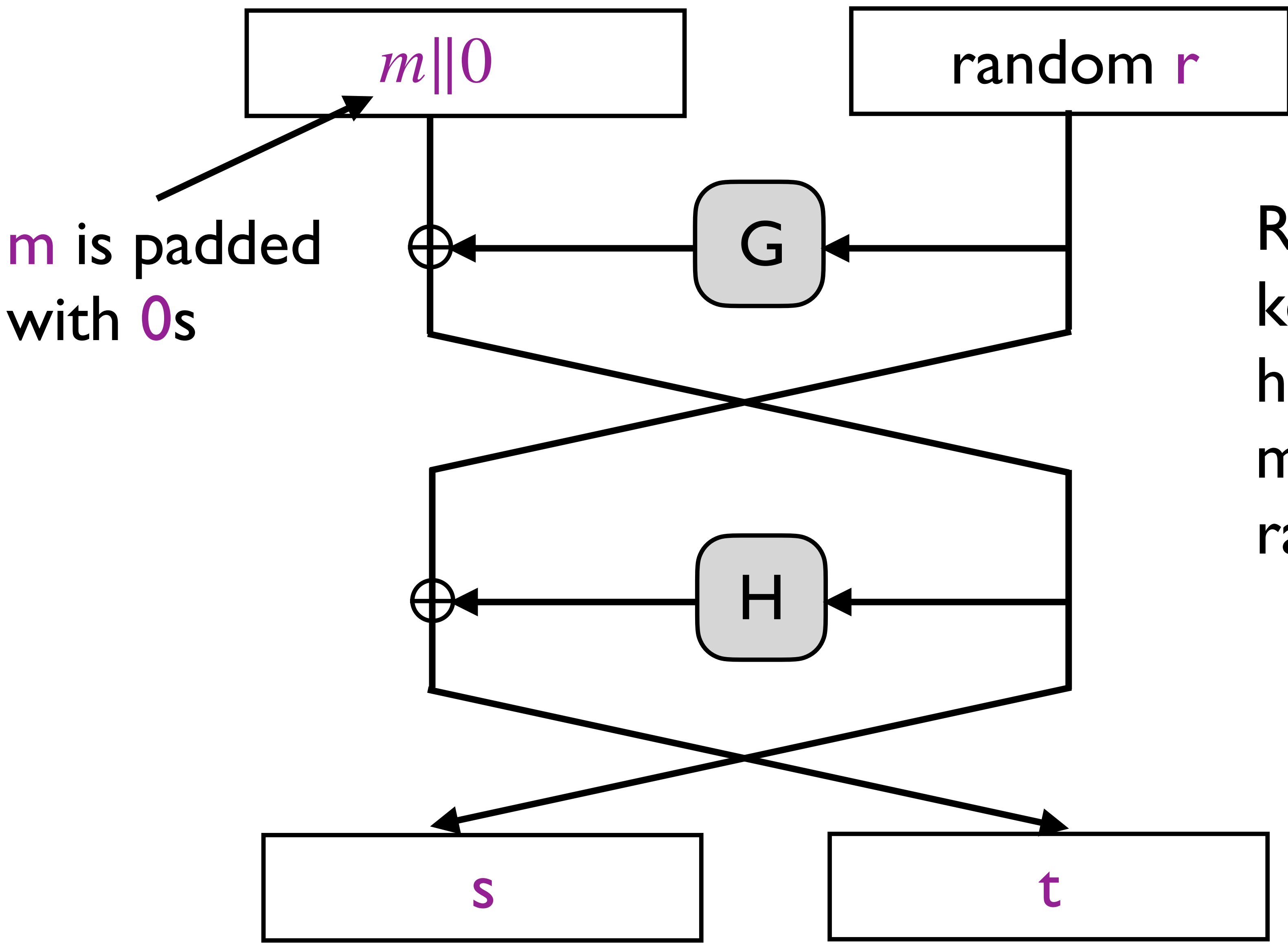
RSA and El Gamal are malleable, so not CCA-secure.

There are two basic approaches we can take:

- Modify CPA-secure public key encryption protocols to make them CCA-secure.
- Use KEM/DEM with a CCA-secure private key scheme.

RSA-OAEP

In RSA-OAEP ("Optimal asymmetric encryption padding"), the message m is put through a Feistel network.



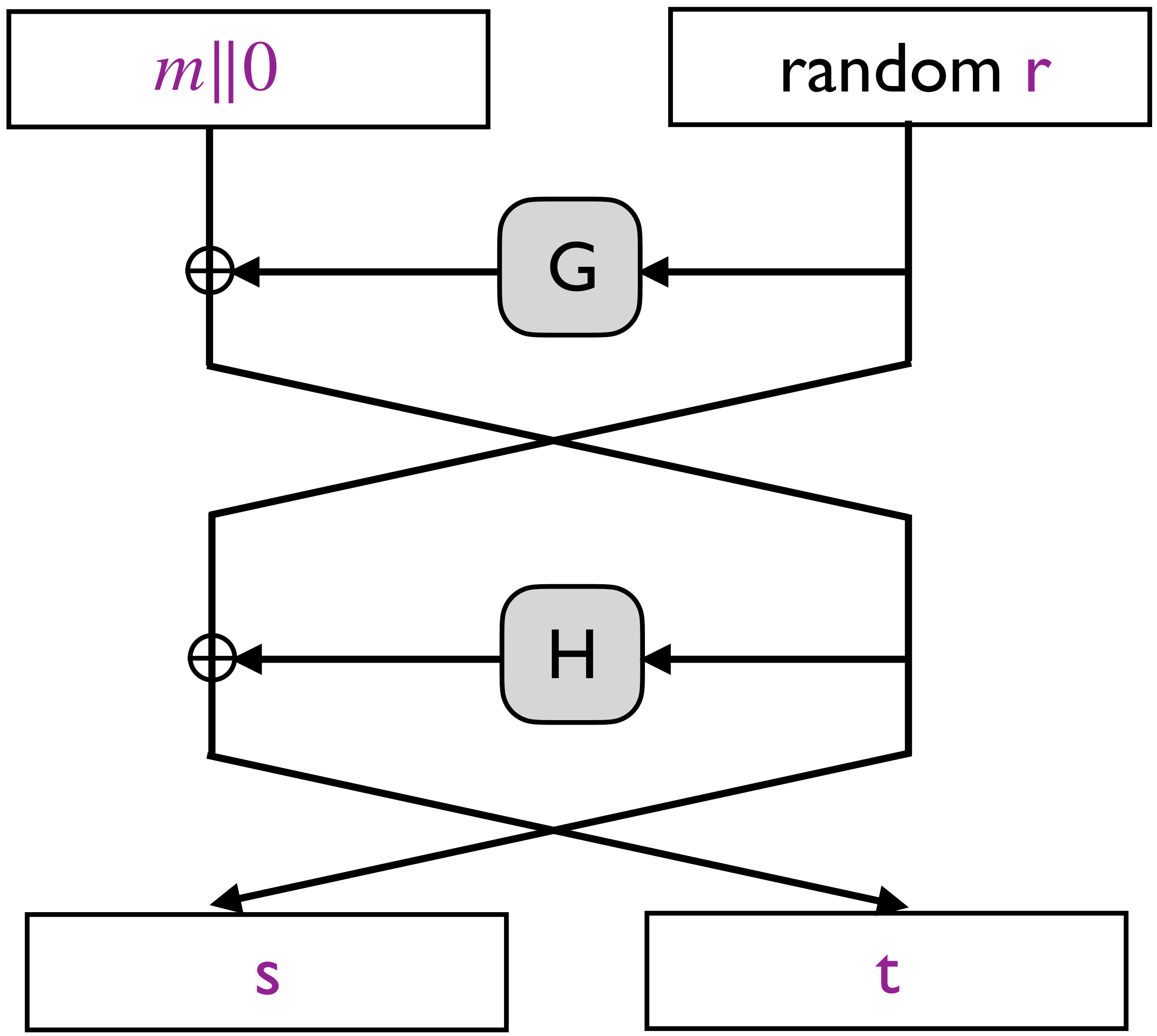
Rather than use a key, G and H are hash functions modeled as random oracles.

Then $s\|t$ is encrypted with RSA: ciphertext $(s\|t)^e \bmod N$.

This class is being recorded

RSA-OAEP Decryption

Decryption recovers $(s\|t) = c^d \bmod N$ and then runs the Feistel network backwards.



Bob rejects the ciphertext as invalid unless the correct number of padding 0s are present.

RSA-OAEP

RSA-OAEP is CCA secure if G and H are modeled as random oracles and the RSA assumption is true (It is hard to find x such that $x^e = y \bmod N$ for random y).

> Because G and H are considered as random oracles, Eve has to know nearly every bit of r and t in order to be able to query them properly. The RSA assumption means that Eve is missing some of this information.
>
> As with the authenticated encryption case, Eve cannot come up with a ciphertext that gives the correct padding.

Caveat: There are two possible "invalid" responses: 1) padding is wrong and 2) the message is not in the right form to be $(s||t)$. These must be indistinguishable (including via side-channel information) or the scheme can be broken.

KEM Approach

When we combine a CCA-secure KEM with a CCA-secure DEM/private key system, we get a CCA-secure public key system.

Reminder:

Definition: A key encapsulation mechanism is a set of three probabilistic polynomial-time algorithms (Gen, Encaps, Decaps).

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).

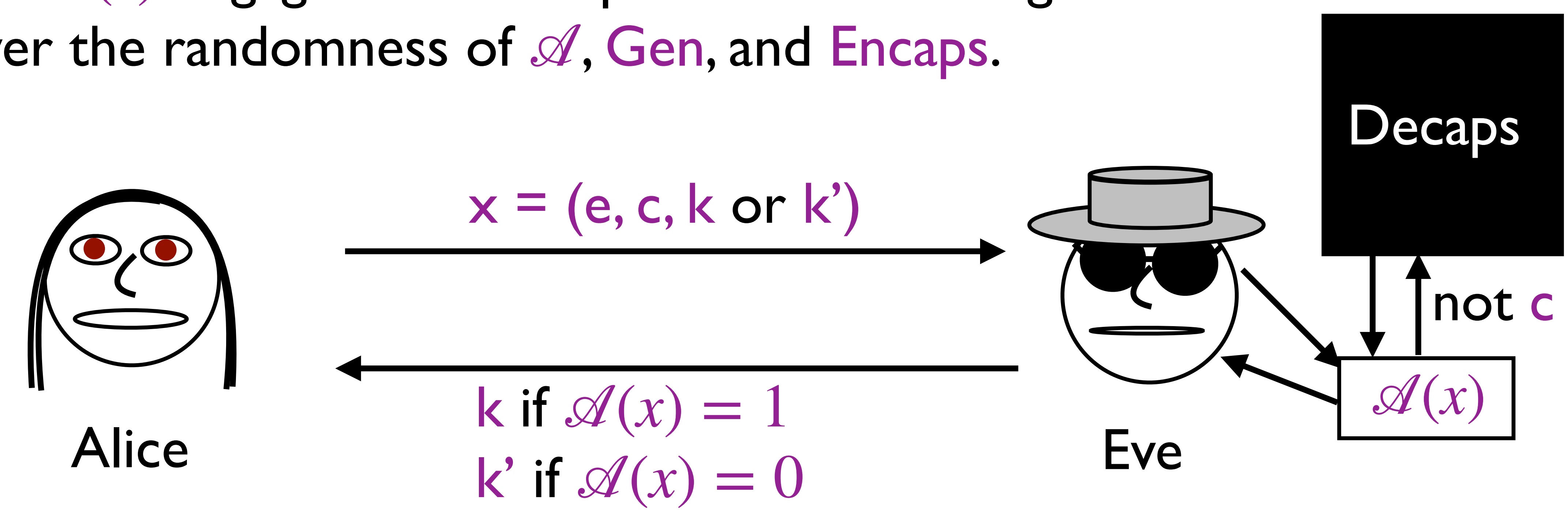
Decaps is the decapsulation algorithm. It takes as input d and c and outputs some $k' \in \{0,1\}^{\ell(s)}$.

CCA 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 CCA secure if for all attacks $\mathscr{A}$ taking as inputs e, c, and either k or k', and with oracle access to Decaps except for c,

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

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



This class is being recorded

DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

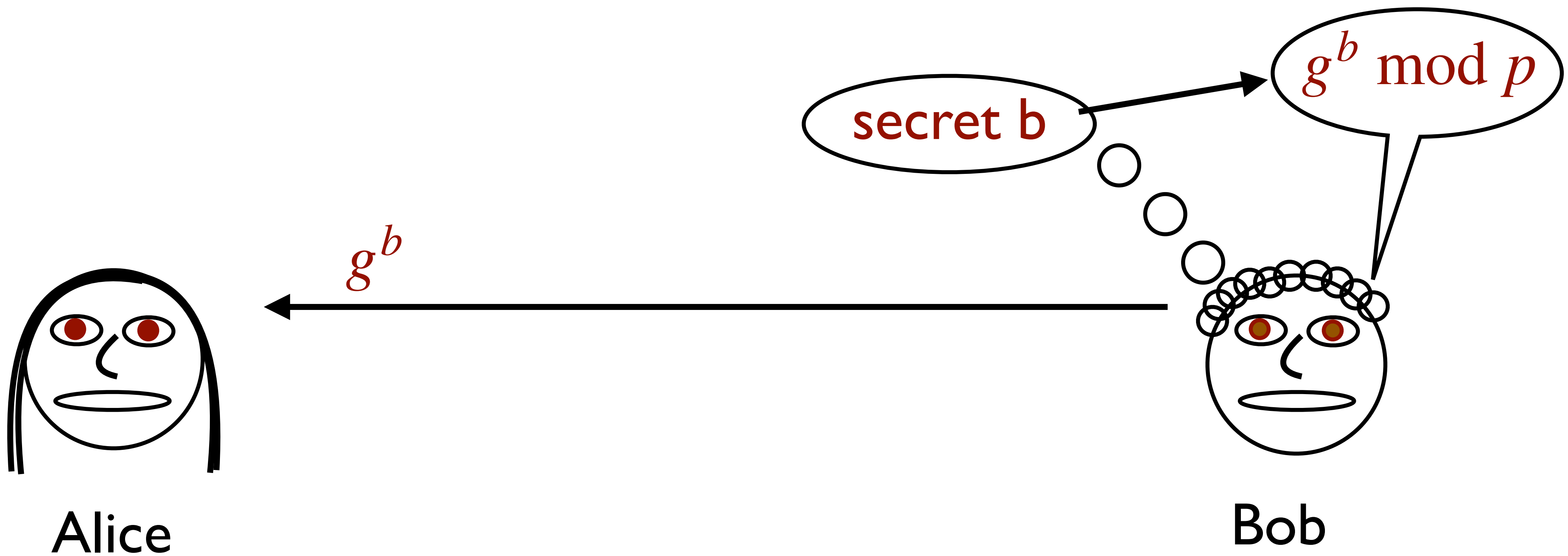
When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).

DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).

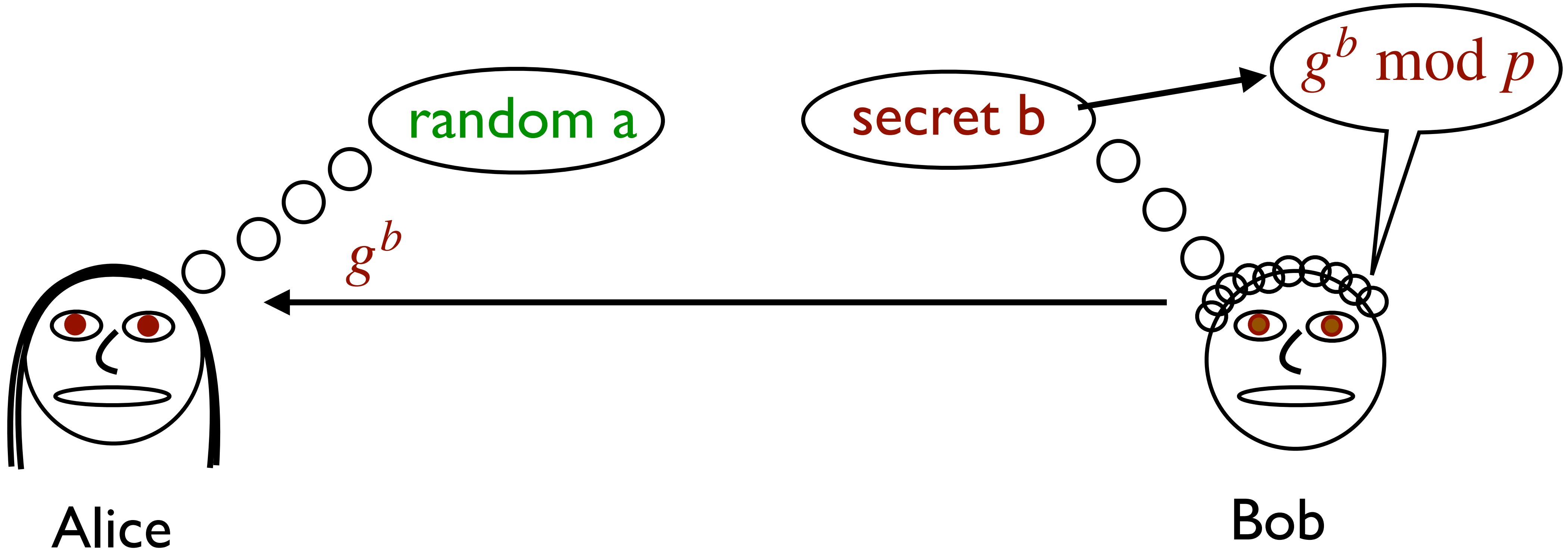


DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).

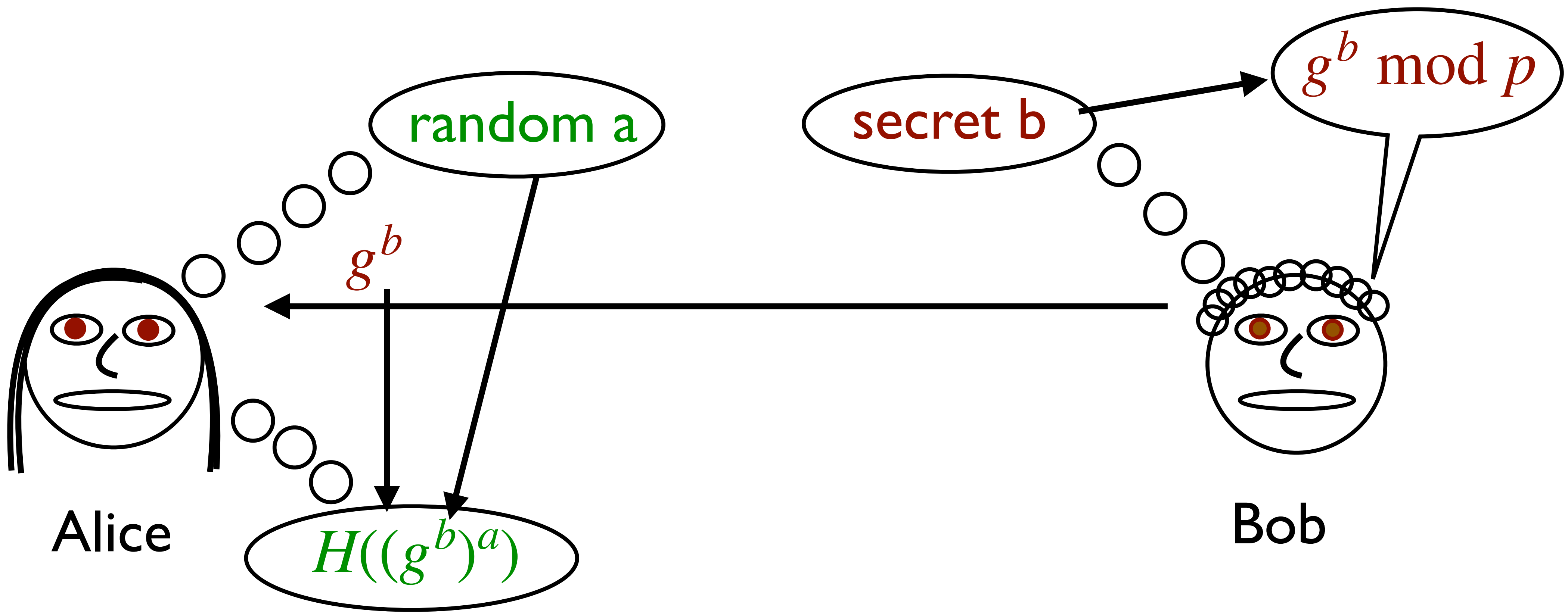


DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).

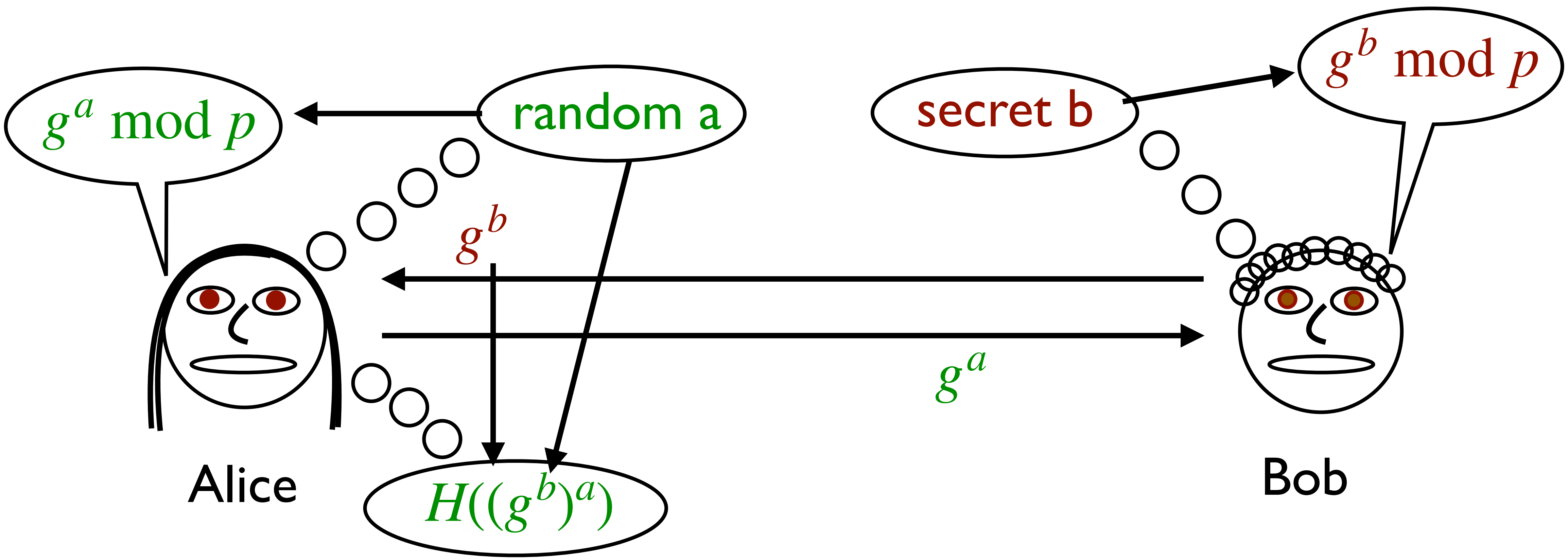


DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).

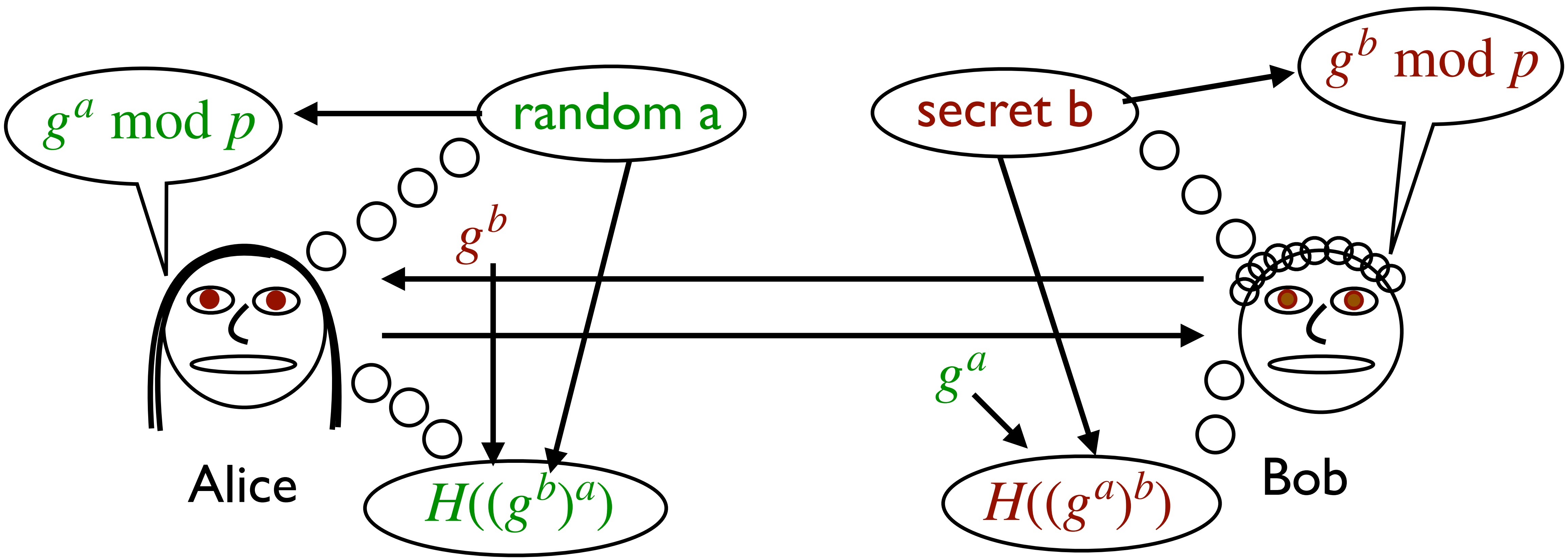


DHIES

The Diffie-Hellman-based KEM is CCA-secure assuming:

- A stronger version of the Diffie-Hellman security assumption.
- H(x) is modeled as a random oracle.

When combined with a CCA-secure private key system, this is DHIES (Diffie-Hellman Integrated Encryption System).



RSA-Based CCA-Secure KEM

Gen: Pick a (public) key derivation function H(x), then as usual for RSA, i.e., generate two random primes p and q which are s bits long. Let N = pq. Choose $e, d \in \mathbb{Z}_N^*$ s.t. $ed = 1 \bmod \varphi(N)$. The public key is (N, e) and the private key is (N, d).

Encaps: Choose random x. The ciphertext is $c = x^e \bmod N$ and the key is H(x).

Decaps: Given c and d, compute $x' = c^d \bmod N$. Then the key is H(x').

This is CCA-secure assuming:

- The standard RSA security assumption.
- H(x) is modeled as a random oracle.

Why Are These KEMs CCA Secure?

In both cases, the underlying public key scheme (El Gamal or RSA) is malleable.

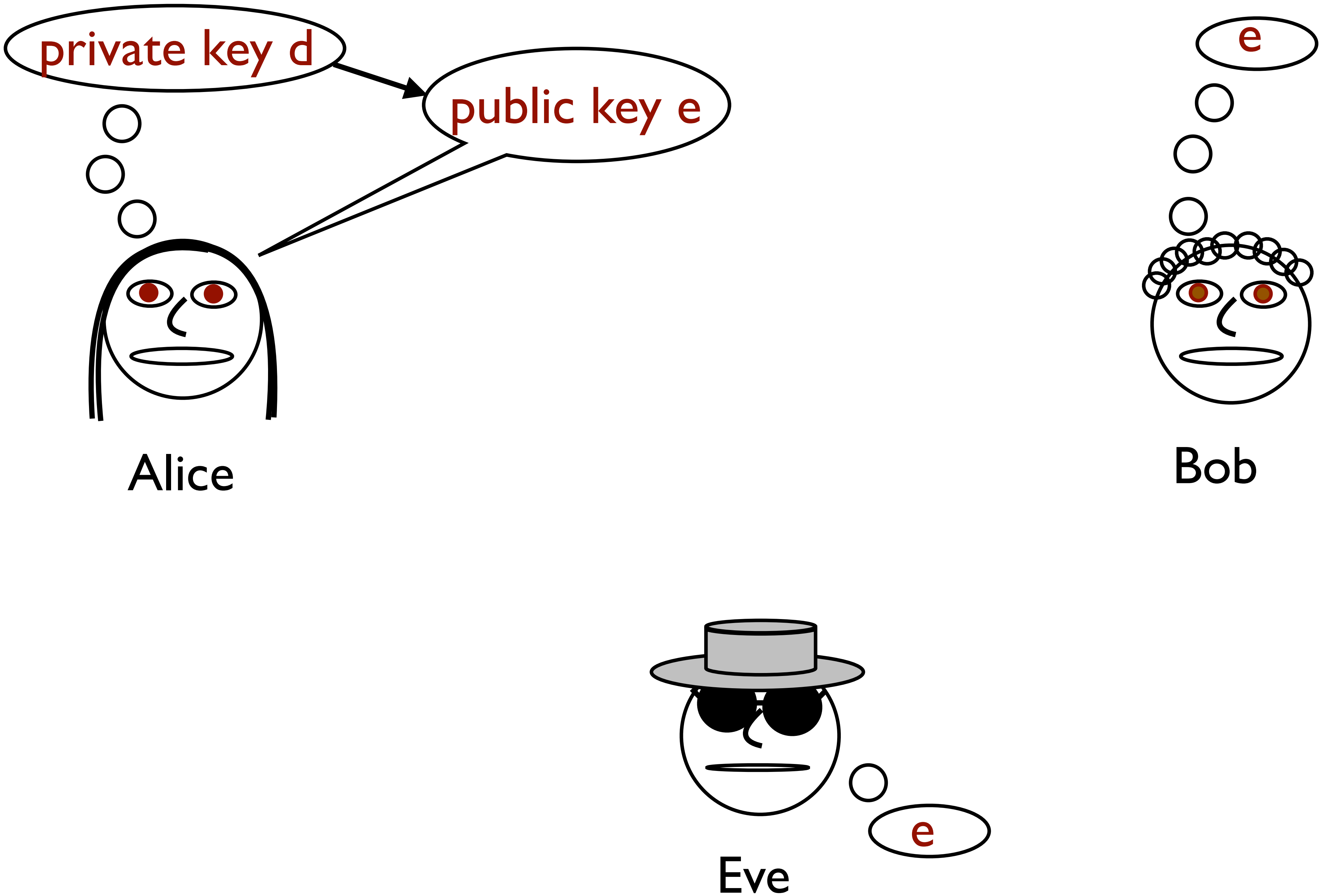
However, when H acts like a random oracle, it breaks the relationship between ciphertexts and encapsulated keys.

In particular, while Eve can easily come up with other valid ciphertexts, they don't tell her anything about the actual key since the output of H(x) is uncorrelated with H(x') when $x \neq x'$.

This is a different mechanism to get CCA security than we saw before.

Digital Signatures

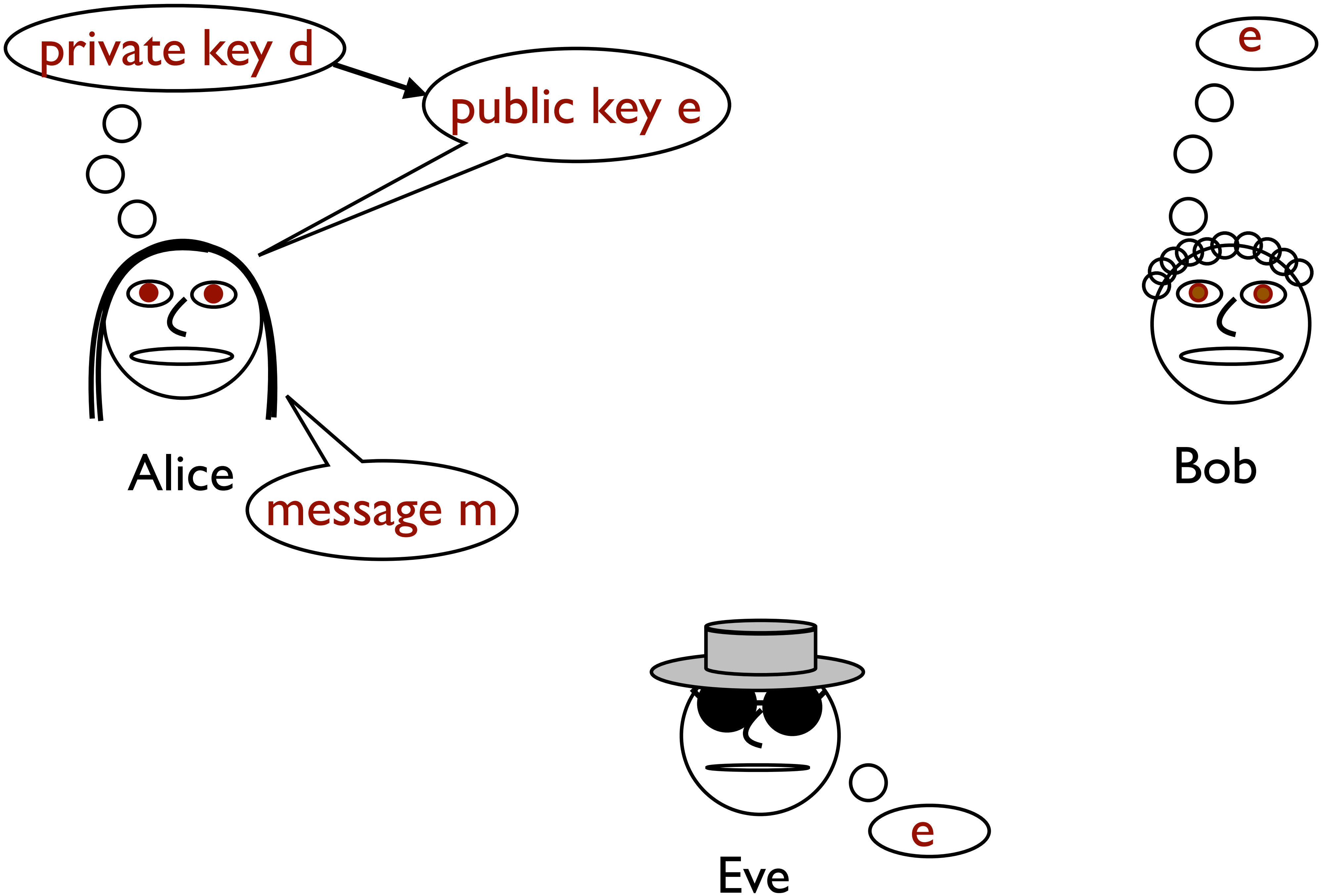
Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signatures

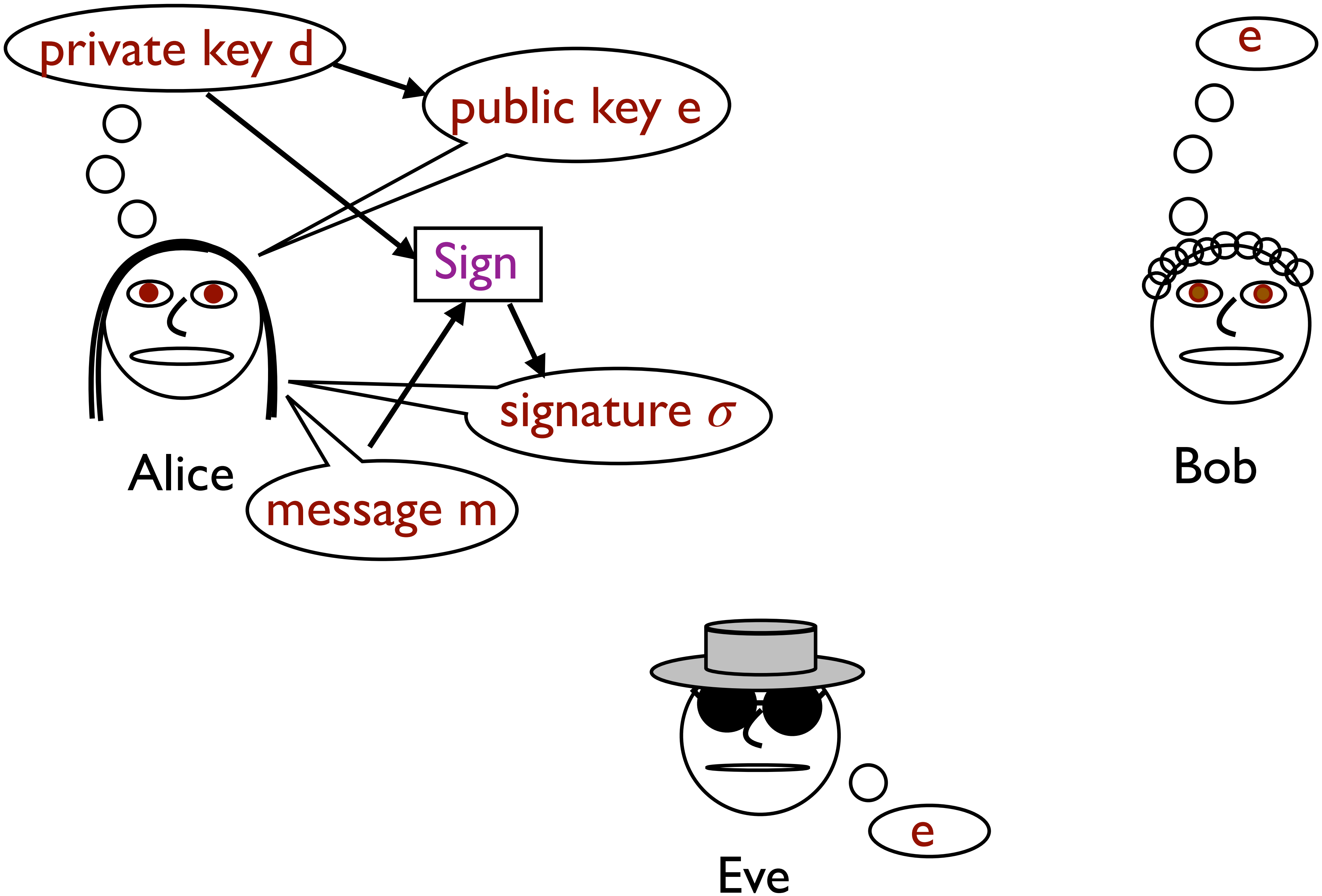
Digital signatures are a public key version of MACs.



This class is being recorded

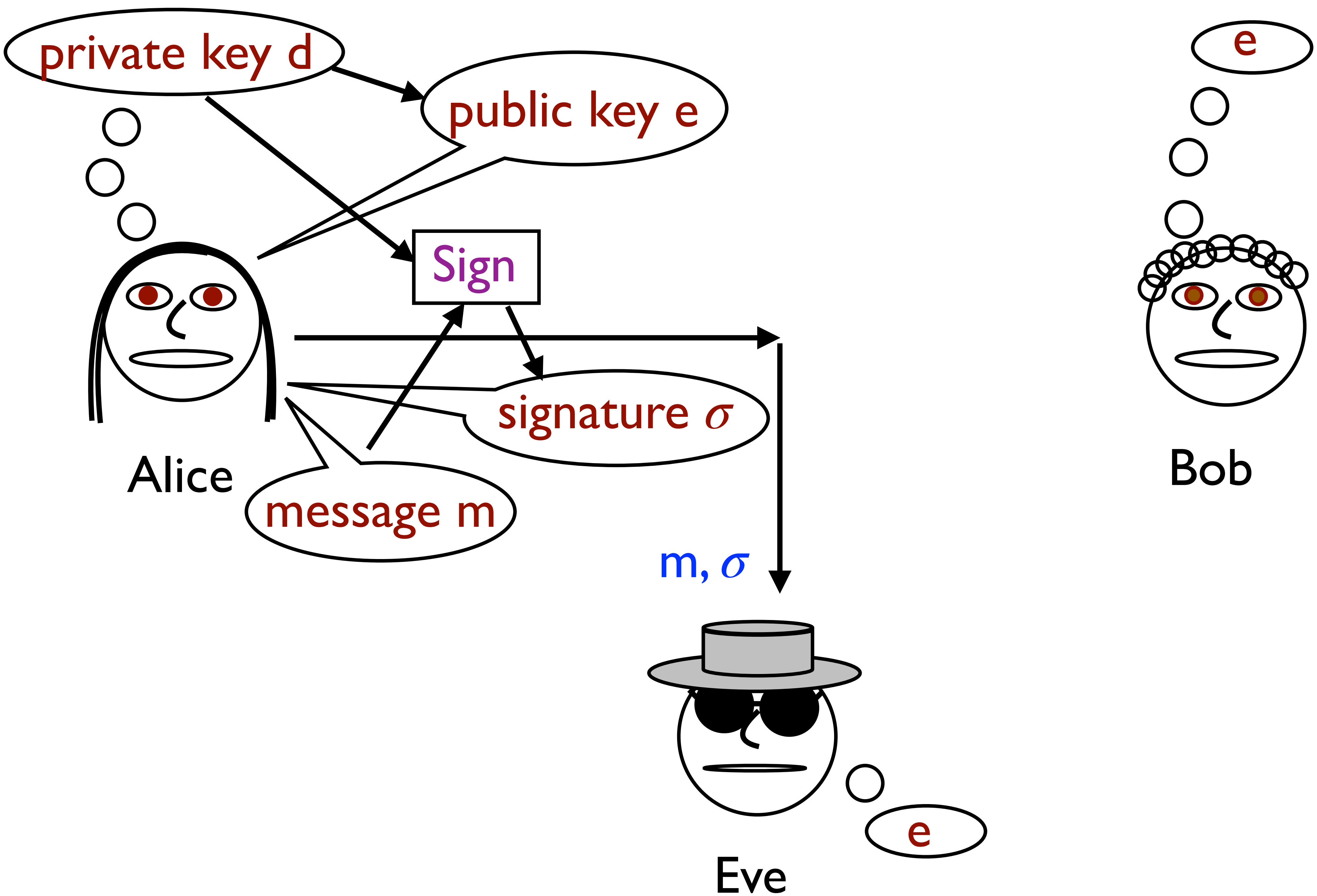
Digital Signatures

Digital signatures are a public key version of MACs.



Digital Signatures

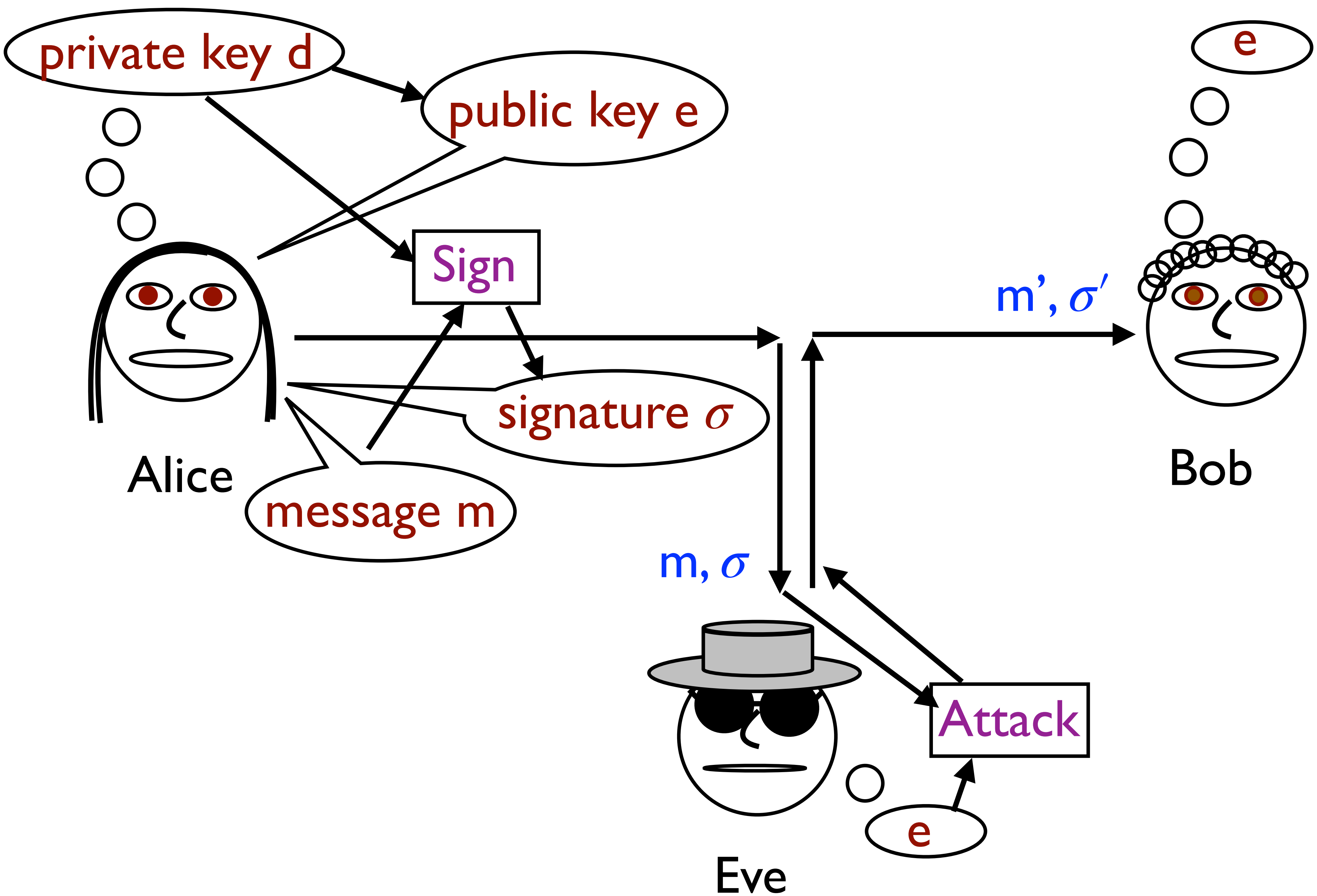
Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signatures

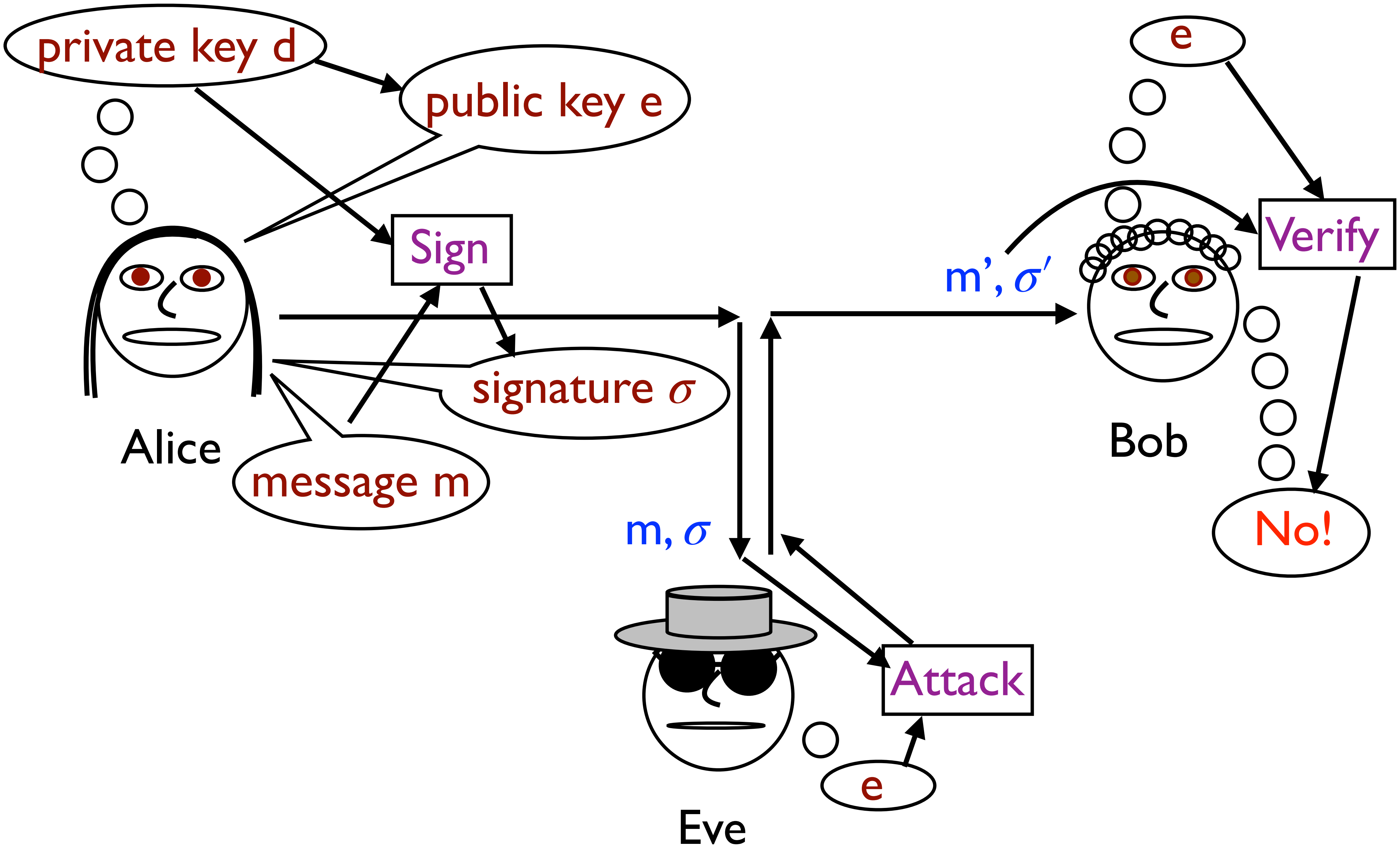
Digital signatures are a public key version of MACs.



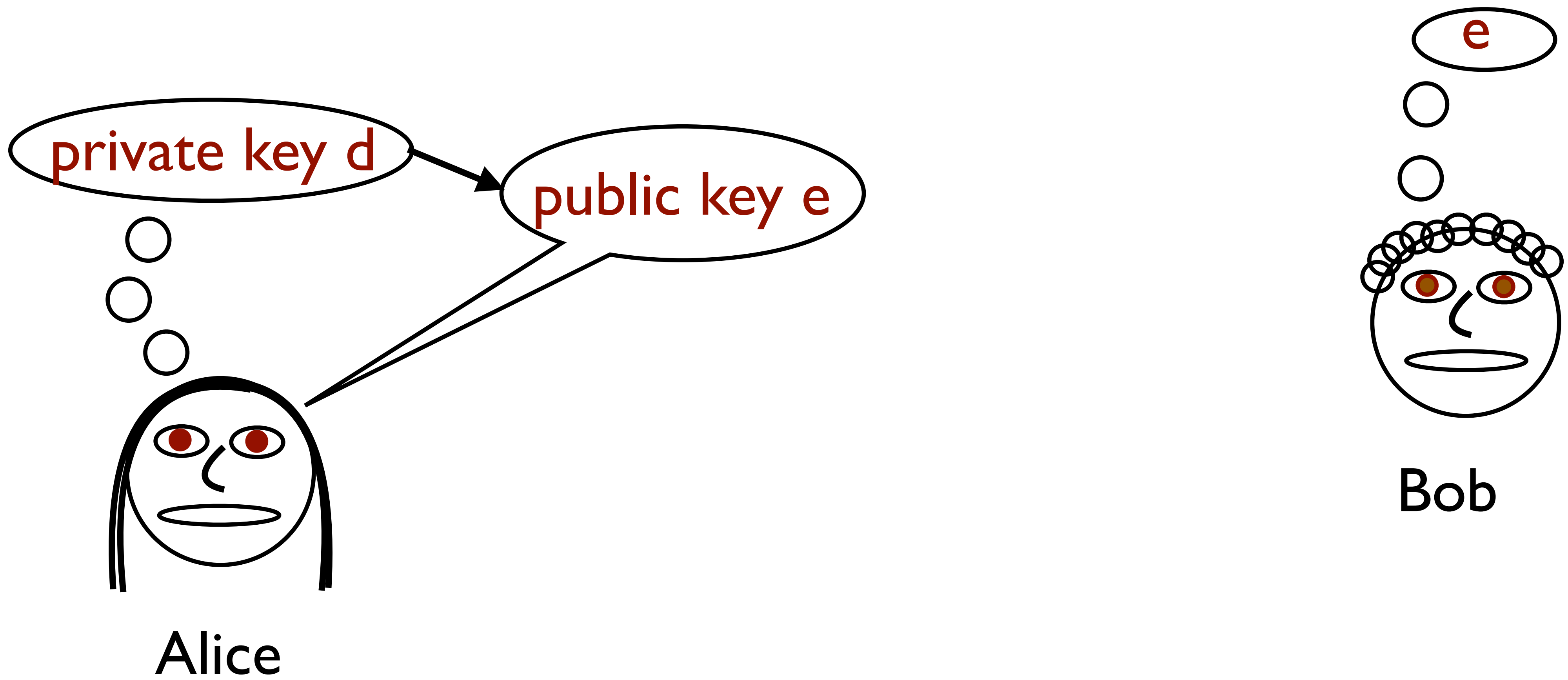
This class is being recorded

Digital Signatures

Digital signatures are a public key version of MACs.

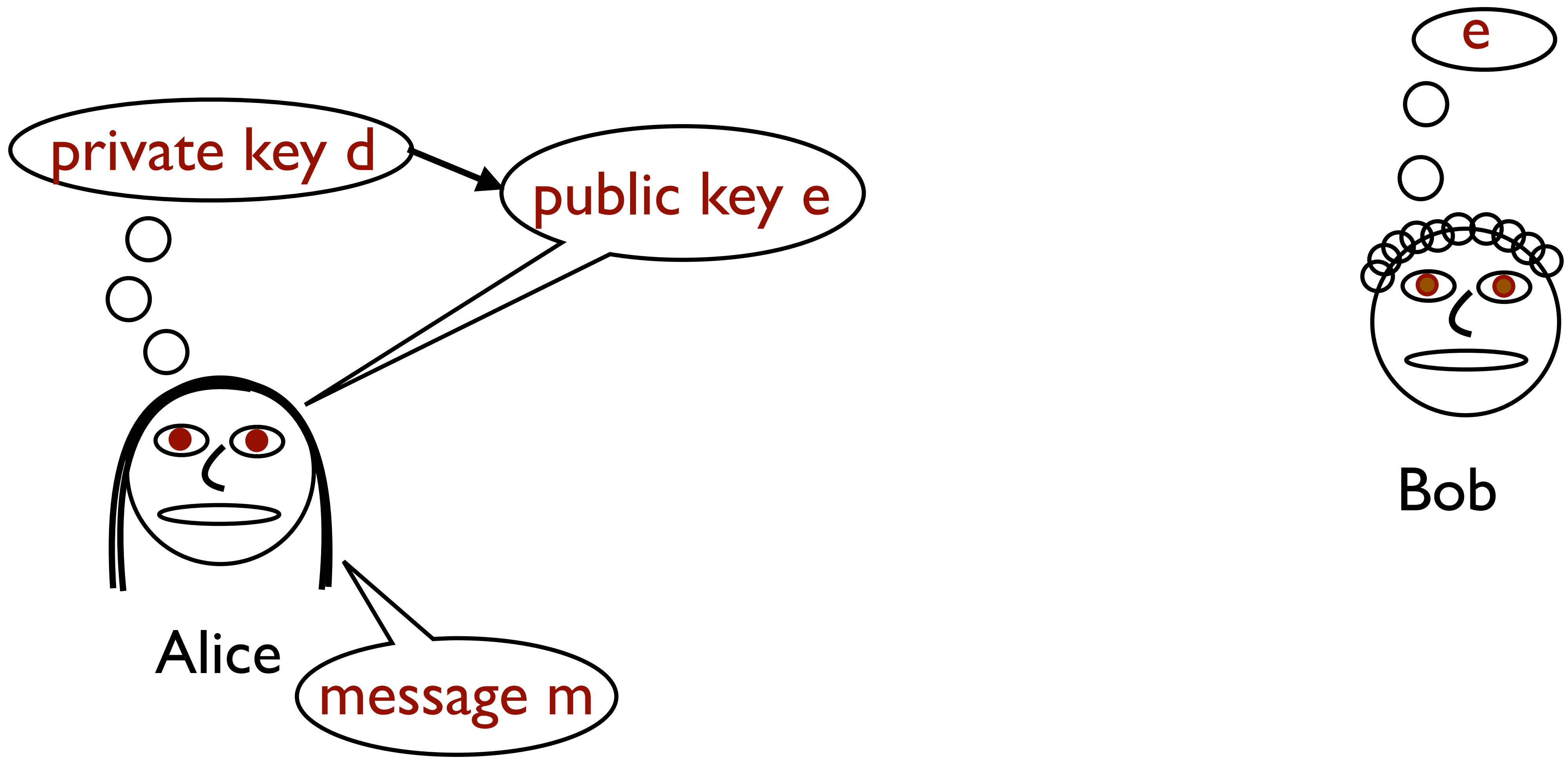


Transferability



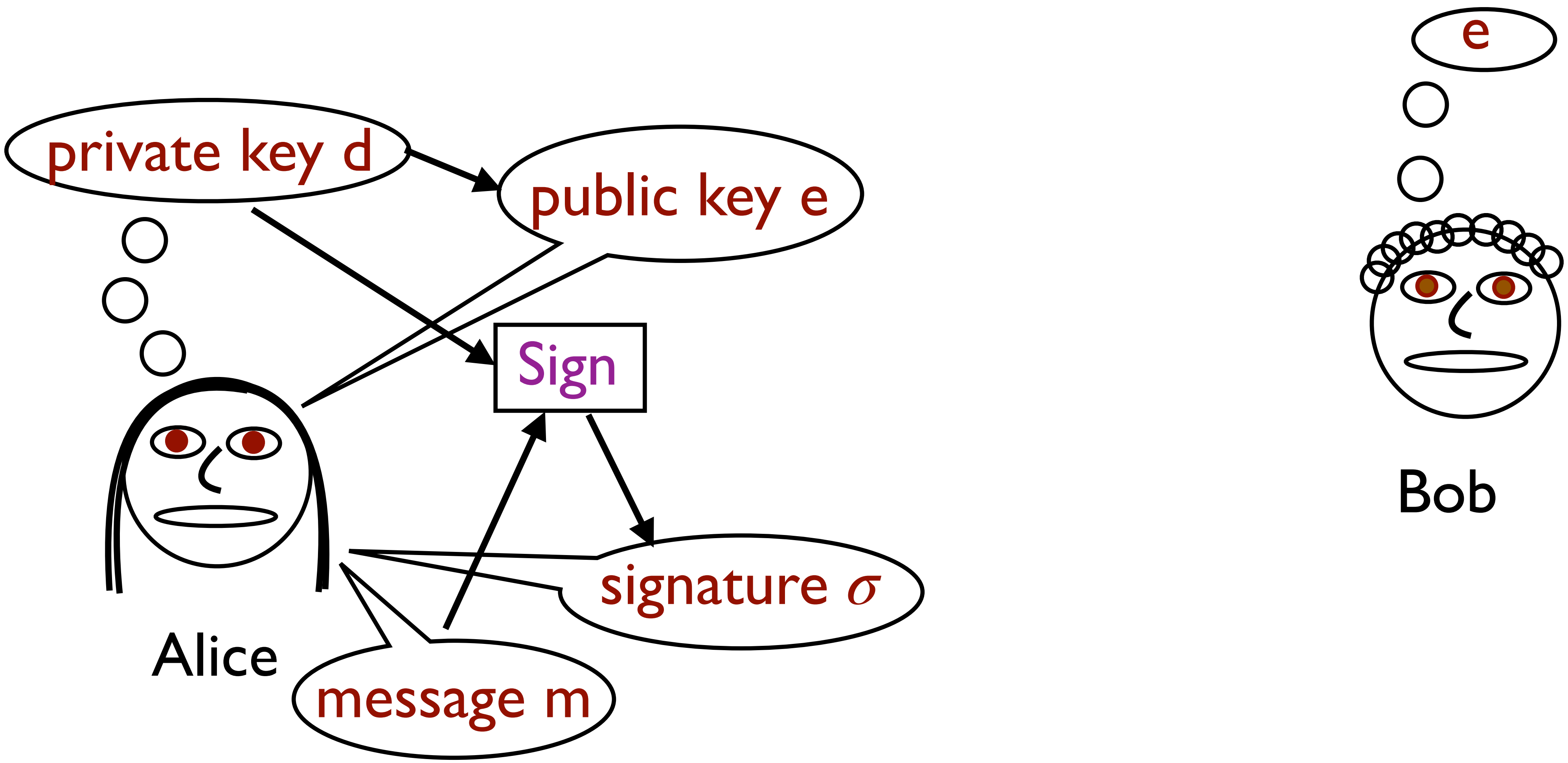
Because signatures use a public key, they are transferable between recipients.

Transferability



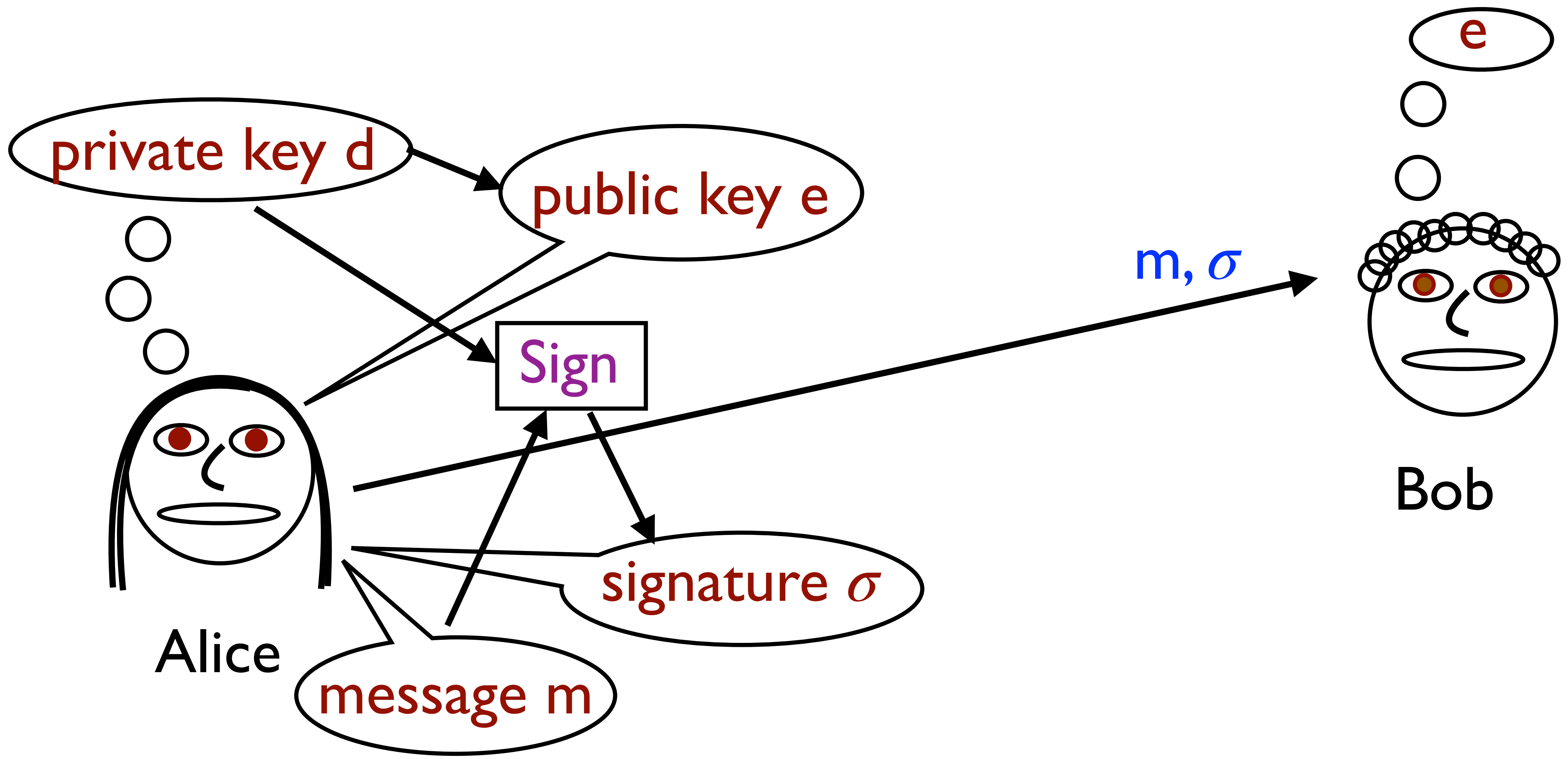
Because signatures use a public key, they are transferable between recipients.

Transferability



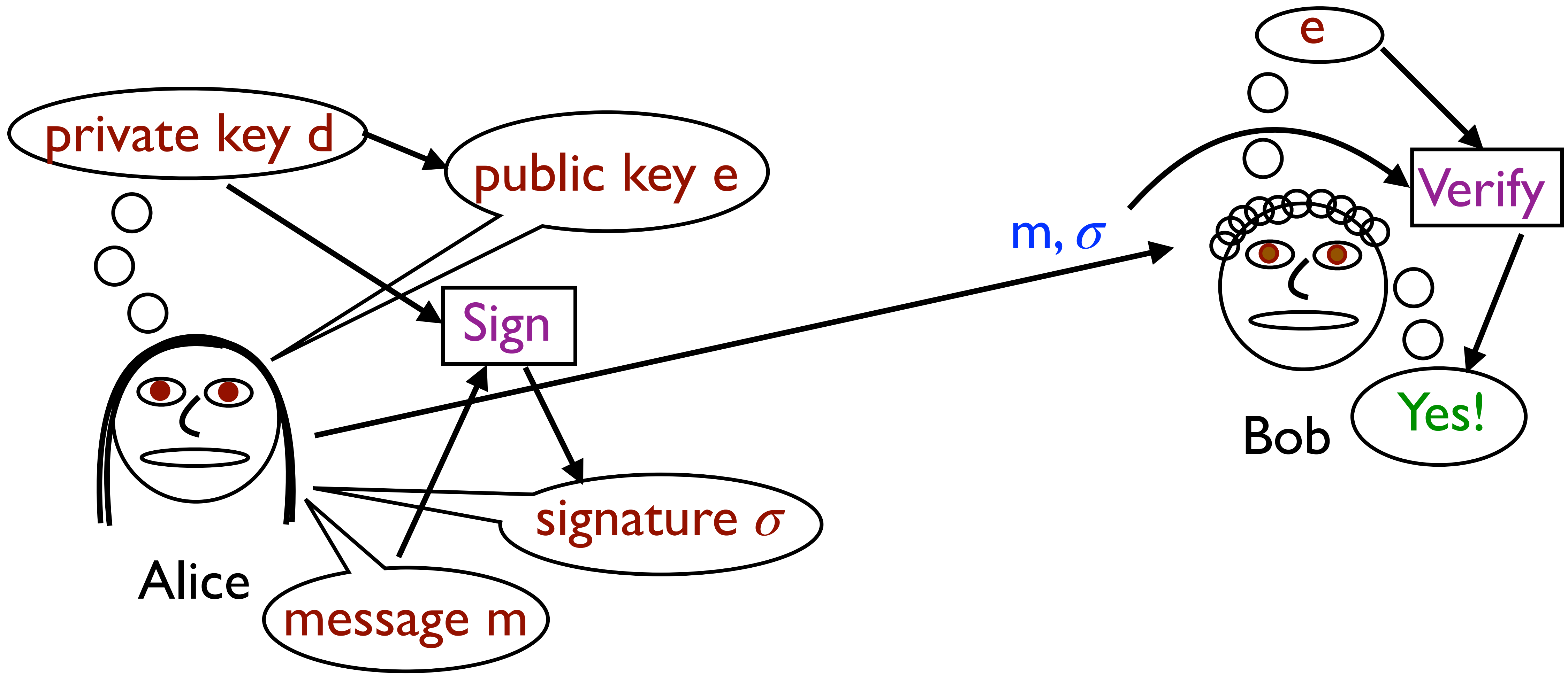
Because signatures use a public key, they are transferable between recipients.

Transferability



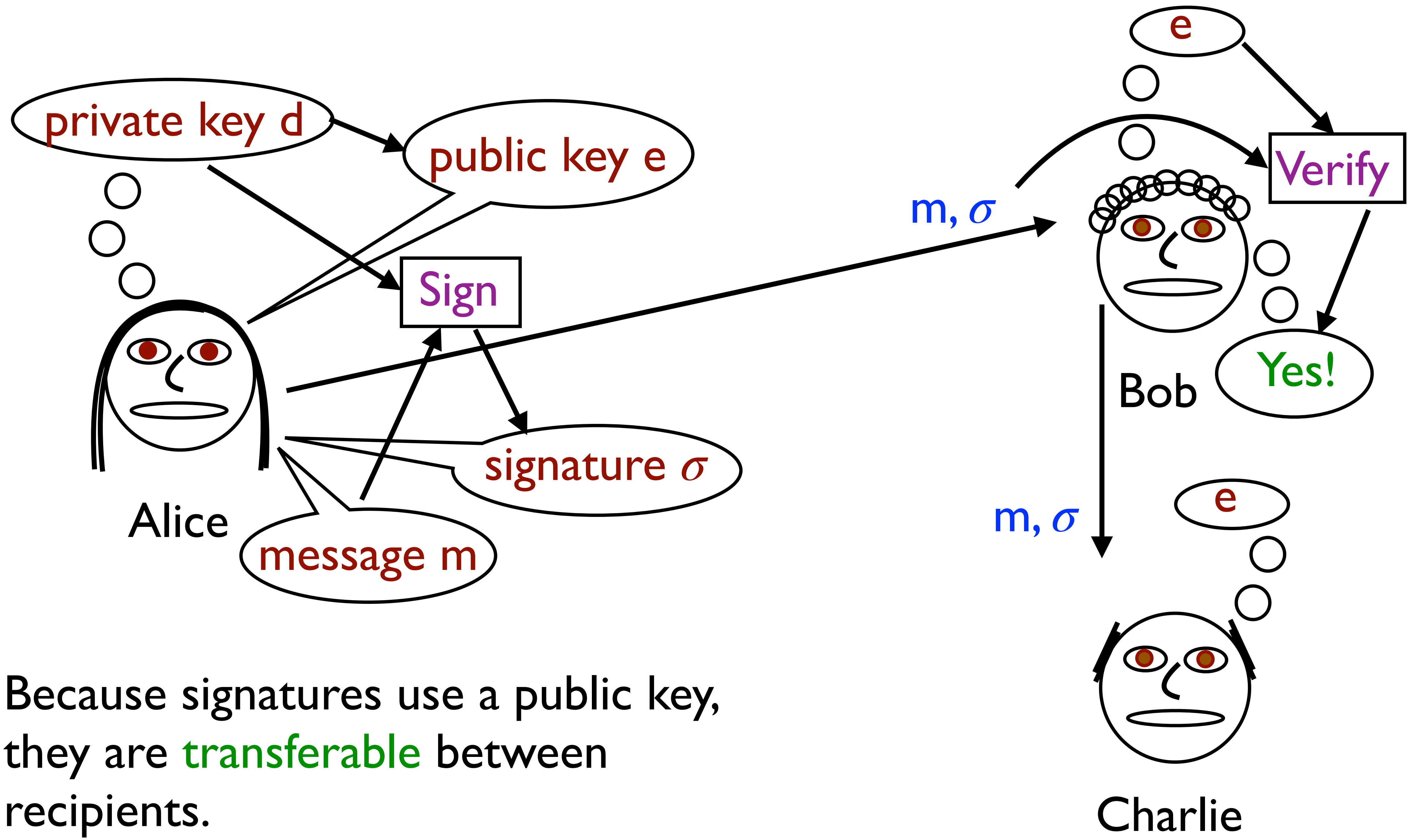
Because signatures use a public key, they are transferable between recipients.

Transferability



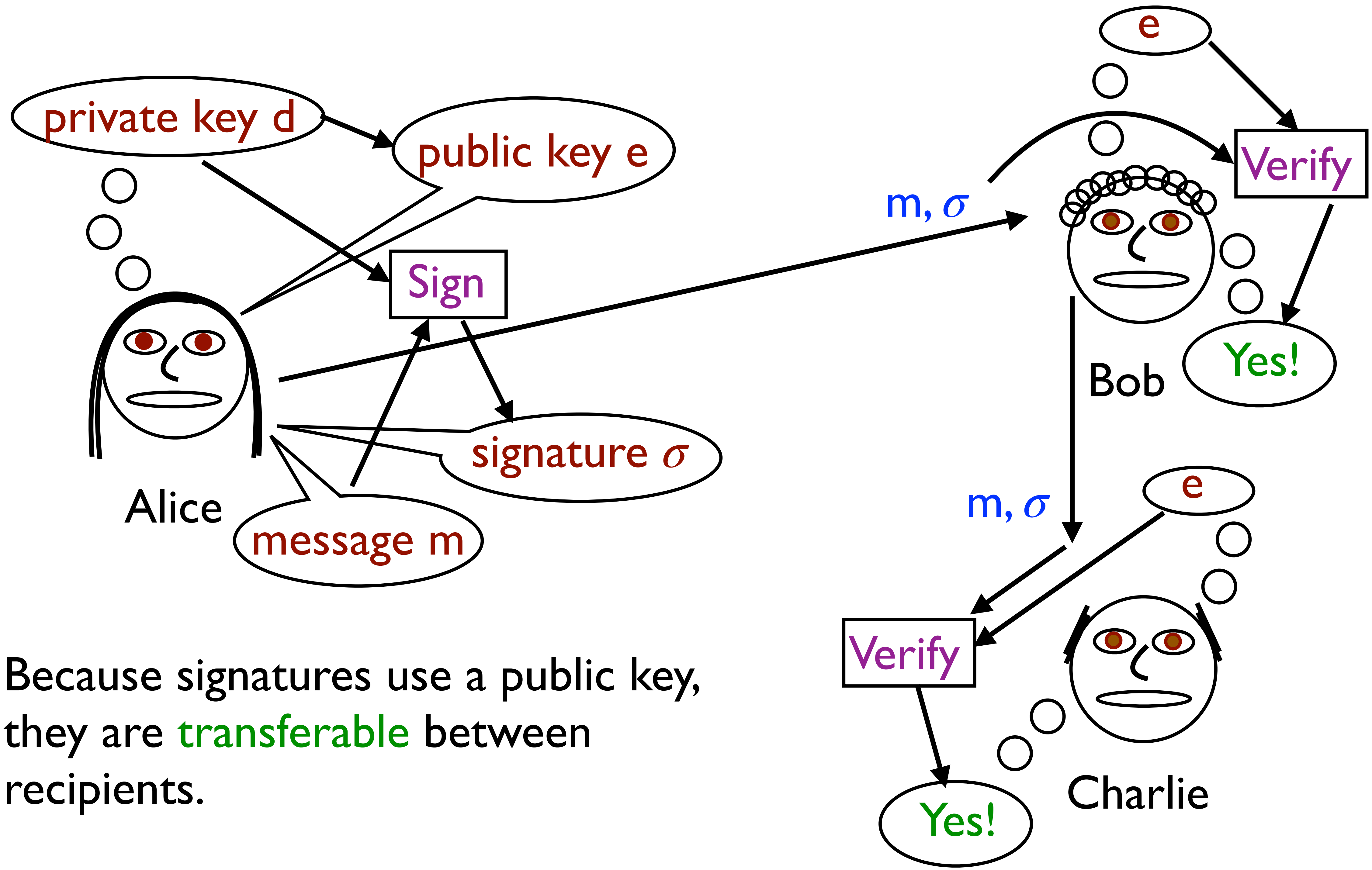
Because signatures use a public key, they are transferable between recipients.

Transferability



Because signatures use a public key, they are transferable between recipients.

Transferability



Because signatures use a public key, they are transferable between recipients.

Digital Signature vs. MAC

In a MAC, each prospective recipient must use a different key. Otherwise, any valid recipient can also forge messages.

This, in turn, means that to send the same message to many recipients, you have to use a different tag for each recipient.

In a digital signature, the same public key is used for everyone. That means that the same signature can be used for any number of recipients. All of them verify it using the same information.

- Digital signatures are more efficient when sending to many recipients.
- Digital signatures allow you to sign a message to someone you don't know (provided they have your public key).
- Transferability allows you to pass on a digitally signed message to someone else.
  - This is critical for certificate authorities and public key infrastructure.

Non-Repudiation

The non-repudiation property of digital signatures says that Alice cannot later deny having sent the message.

In particular, Bob can present the signed message to a third party Charlie who also has Alice's public key and Charlie will agree with Bob that the message is valid and was signed by Alice.

This is important, e.g., for legal contracts:

If necessary, Bob can prove in court that Alice agreed to the contract.

Note that a MAC does not have this property: Anyone who can verify can forge messages.

Vote: Are digital signatures more or less secure than hardcopy signatures? (More secure/Less secure/Similar security)