CMSC/Math 456: Cryptography (Fall 2023)

Lecture 27

Daniel Gottesman

Administrative

Final exam: Monday, Dec. 18, 1:30-3:30 PM

- Will be open book again (textbook, lecture notes)
- Topics covered: Everything but some topics (quantum cryptography, post-quantum cryptography, multiparty computation) are qualitative understanding only.

Next week, I will have an extended office hour: Tuesday, Dec. 12, 10 AM to 1 PM. (Usual location: Atlantic 3251.)

A list of topics covered in the course and more practice problems from the textbook are available on the course website. Last year's final and solutions are on ELMS.

Course evaluations are now available to fill out.

This class is being recorded

Review Plan

- Principles and basic tools (Kerckhoff's principle, computational complexity, proof by reduction)
- Cryptographic primitives (Pseudorandom generators, pseudorandom functions, hash functions)
- Cryptographic protocols (Private and public-key encryption, key agreement, KEM, MAC, authenticated encryption, digital signature, identification protocol)

Principles and Basic Tools

- Kerckhoff's principle
- Computational complexity
- Proof by reduction

This class is being recorded

Kerckhoff's Principle

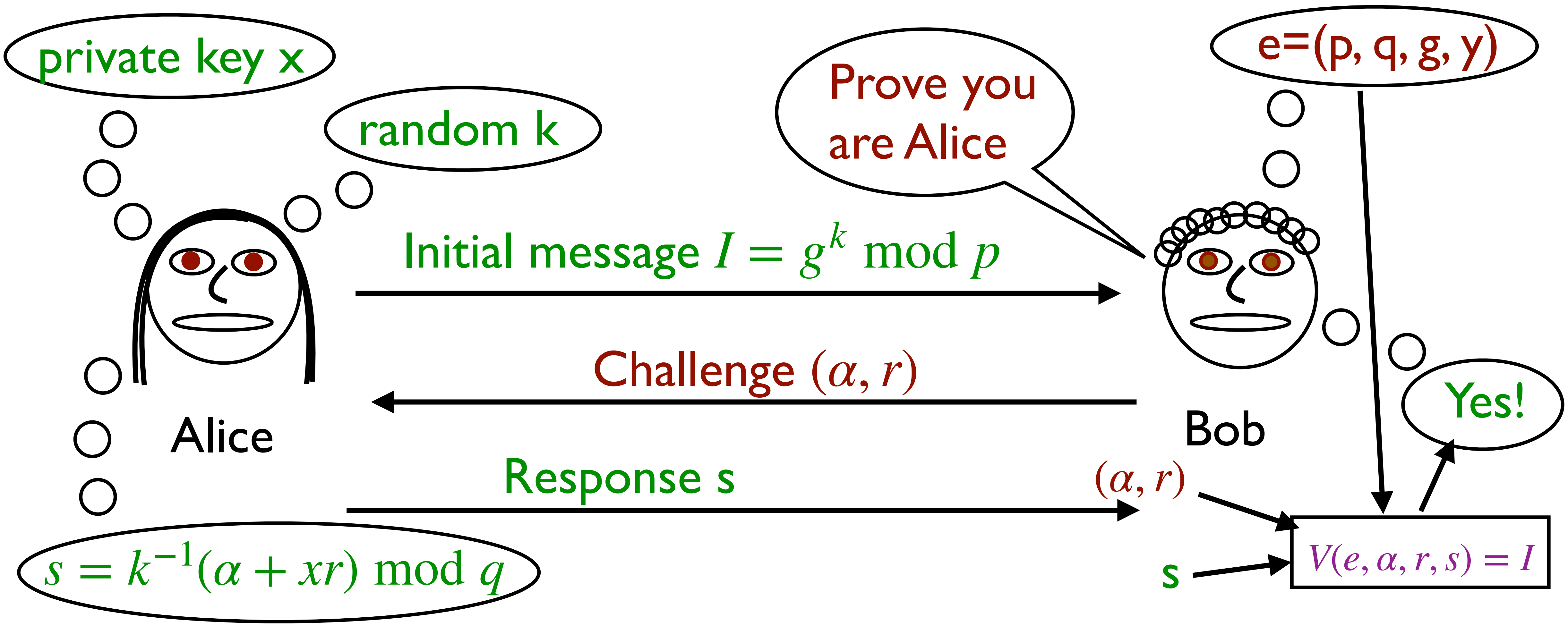
Assume the protocol is known by the adversary. Only the key is secret.

This means that anything that is not explicitly listed as part of the private key (or otherwise is secret) is known to Eve:

- Any parameters of the protocol (e.g., prime q or base g) are known to Eve.
- Any functions involved (e.g., hash function $H(x)$) are known to Eve.
- Public keys are certainly known to Eve.
- Private keys are not known by Eve.
- Random values picked by a participant and not explicitly announced are not known by Eve.

Example: Identification Protocol

The protocol involves the following values: p, q, g, x, y, k, l, α, r, s.
Which are known to Eve and which are not?



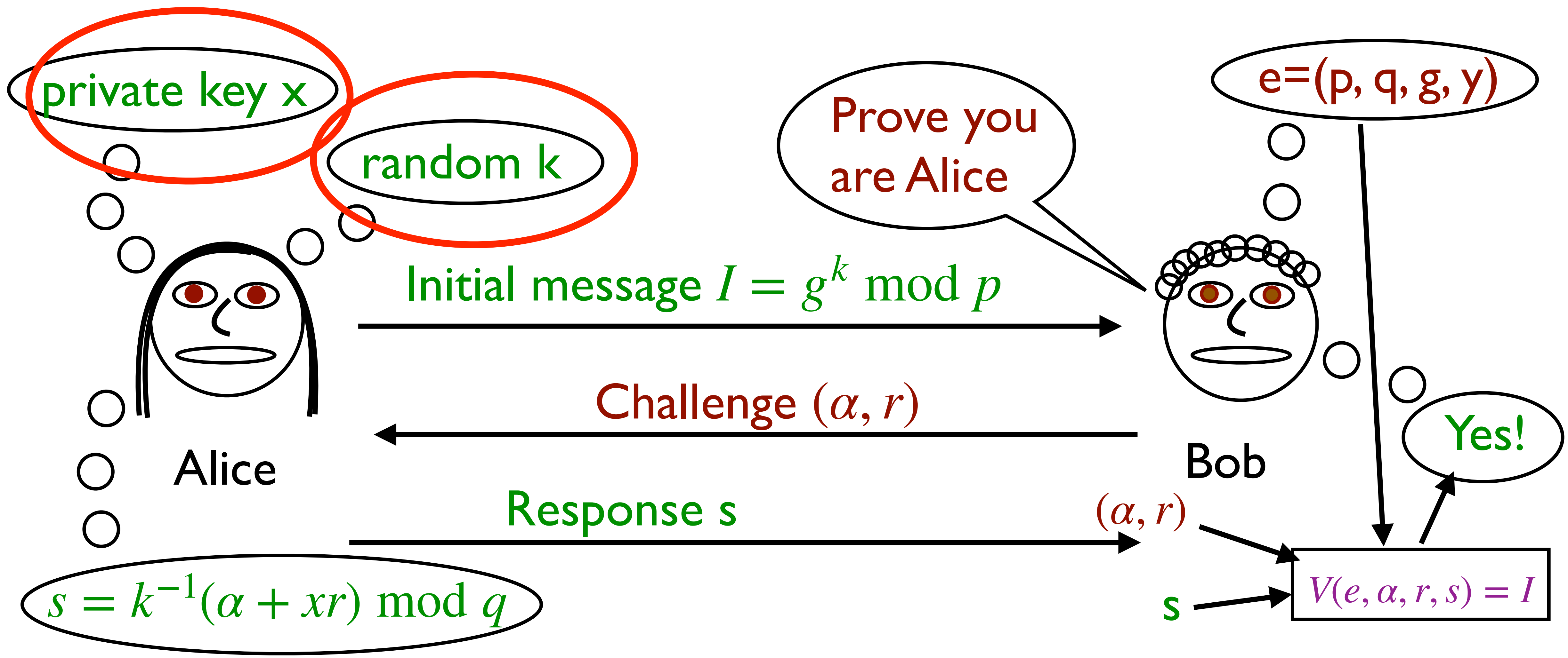
$$V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{r s^{-1}}$$

This class is being recorded

Example: Identification Protocol

The protocol involves the following values: p, q, g, x, y, k, l, α, r, s.
Which are known to Eve and which are not?

Secret: x, k



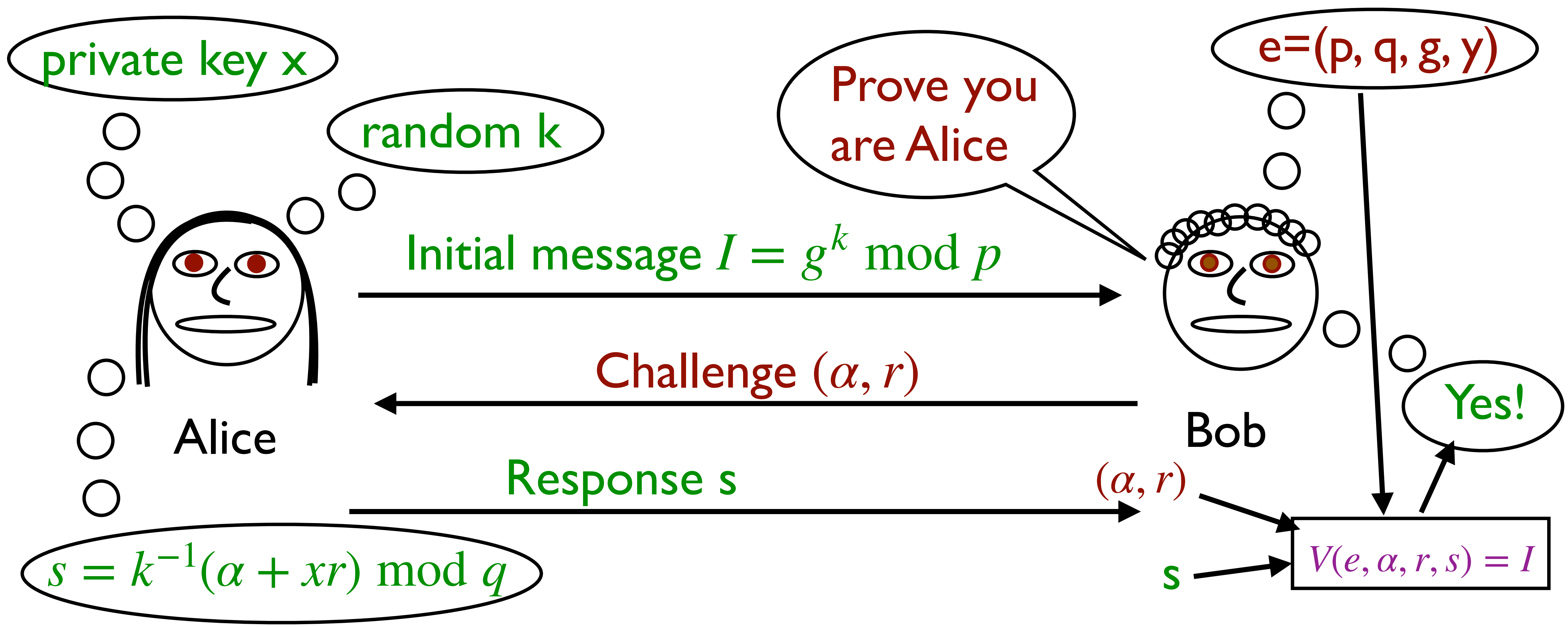
$$V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{r s^{-1}}$$

This class is being recorded

Example: Identification Protocol

The protocol involves the following values: p, q, g, x, y, k, l, α, r, s.
Which are known to Eve and which are not?

Secret: x, k



$$V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{r s^{-1}}$$

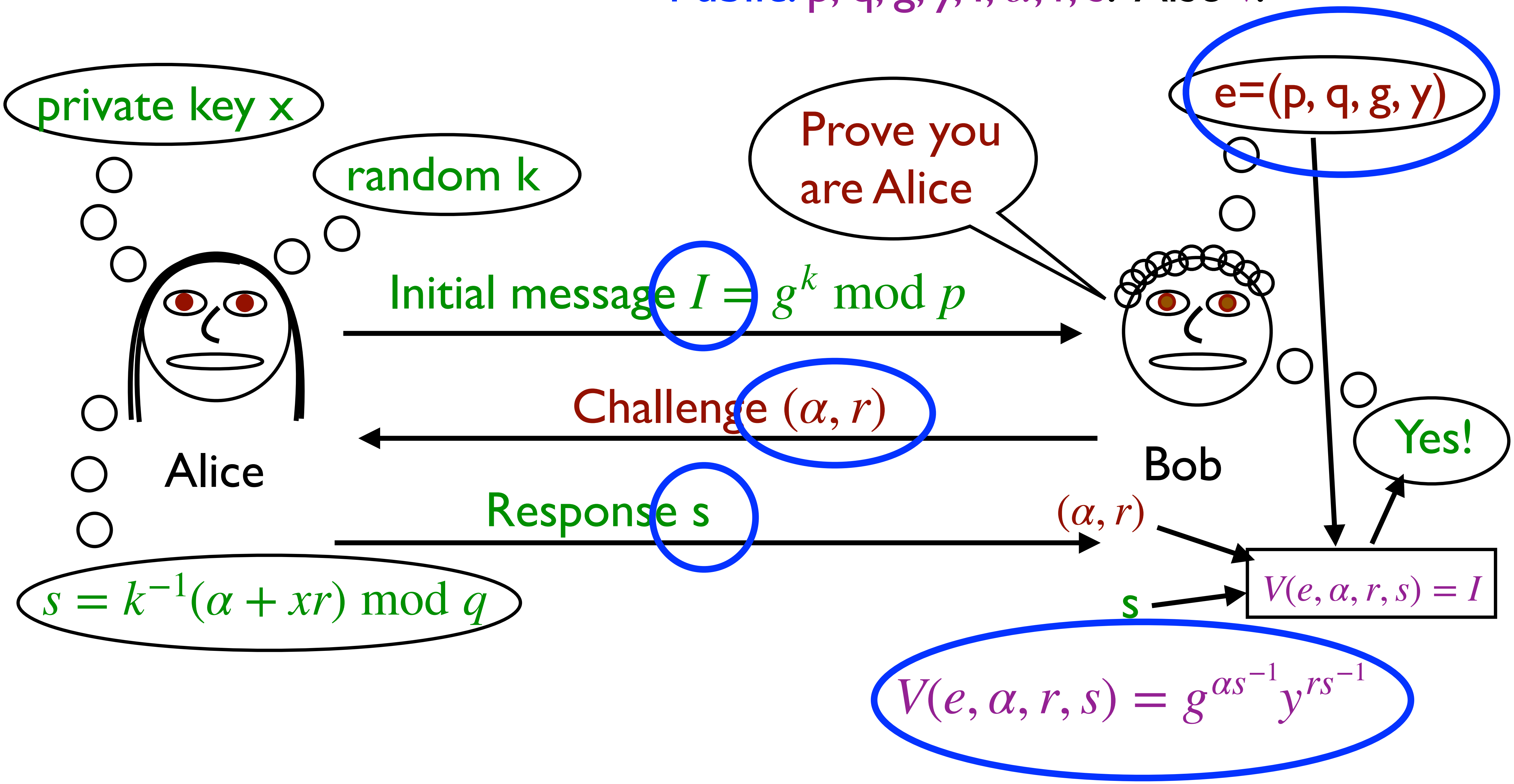
This class is being recorded

Example: Identification Protocol

The protocol involves the following values: p, q, g, x, y, k, l, α, r, s.
Which are known to Eve and which are not?

Secret: x, k

Public: p, q, g, y, l, α, r, s. Also V.



This class is being recorded

Efficient and Negligible

Almost always, we are interested in efficient algorithms, namely ones which run in a time polynomial as a function of the size of the input to the function.

Big-O Notation:

A function $f(x) = O(g(x))$ for the function g(x) if f is *less than or equal to* g in the asymptotic limit of large x. Specifically, $f(x) \leq Cg(x)$ for constant C and sufficiently large x.

E.g.: $f(x) = 1/(100x^3)$ is $O(1)$ and $O(1/x^2)$ but is *not* $O(1/x^5)$.

Efficient and Negligible

Almost always, we are interested in efficient algorithms, namely ones which run in a time polynomial as a function of the size of the input to the function.

Big-O Notation:

A function $f(x) = O(g(x))$ for the function g(x) if f is *less than or equal to* g in the asymptotic limit of large x. Specifically, $f(x) \leq Cg(x)$ for constant C and sufficiently large x.

E.g.: $f(x) = 1/(100x^3)$ is $O(1)$ and $O(1/x^2)$ but is *not* $O(1/x^5)$.

A function is negligible if it goes to 0 faster than *any* polynomial. Specifically, $\lim_{x\to\infty} f(x)p(x) = 0$ for all polynomials p(x).

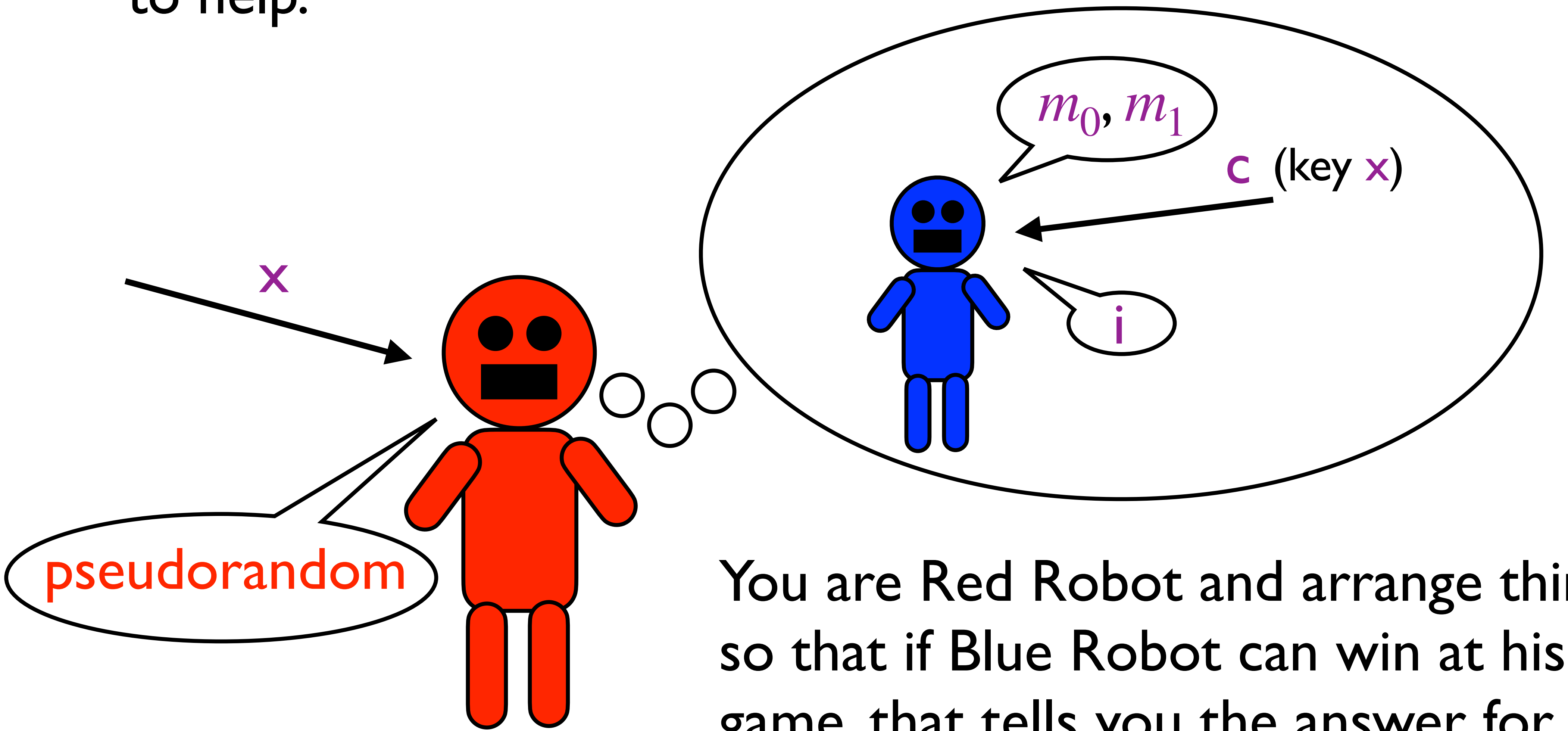
E.g.: $f(x) = 1/(100x^3)$ not negligible

$f(x) = \exp(-\sqrt{x})$ negligible

This class is being recorded

Reductions

Red Robot's job is to play some cryptographic game. He is very bad at his job, so he makes a copy of Blue Robot and runs a simulation of blue robot playing a different cryptographic game to help.



You are Red Robot and arrange things so that if Blue Robot can win at his game, that tells you the answer for your own game.

Red Robot's game reduces to Blue Robot's game.

This class is being recorded

Reduction Example

For example, suppose your job is to break the RSA assumption: Given N, e, and random y, find x such that $x^e = y \bmod N$.

> Blue Robot plays the factoring game: Given N, find a factor of N.

When you are given (N, e, y), you start your simulation and your friend's copy thinks they are going to work. You arrange for them to be given the problem N and they answer p. You take this value out of the simulation and calculate q = N/p, then $\varphi(N)$. Then you use Euclid's algorithm to find $d = e^{-1} \bmod \varphi(N)$ and compute $y^d \bmod N$ and use that for your answer x.

> You have reduced breaking RSA to factoring.

Note: The simulation must be identical to Blue Robot's real job or the copy will realize it is a copy.

Hardness via Reductions

Unfortunately for you, your friend is also bad at their job. The answers they give are not real factors, or maybe they don't answer at all. Either way, your algorithm will fail.

> Disappointed, you conclude that there is no way to do your job.

Is this a valid conclusion?

Hardness via Reductions

Unfortunately for you, your friend is also bad at their job. The answers they give are not real factors, or maybe they don't answer at all. Either way, your algorithm will fail.

> Disappointed, you conclude that there is no way to do your job.

Is this a valid conclusion?

No. There could be a different robot that is better at factoring, or maybe there is a way to beat RSA that doesn't involve making an unauthorized copy of anyone.

Hardness via Reductions

Unfortunately for you, your friend is also bad at their job. The answers they give are not real factors, or maybe they don't answer at all. Either way, your algorithm will fail.

> Disappointed, you conclude that there is no way to do your job.

Is this a valid conclusion?

No. There could be a different robot that is better at factoring, or maybe there is a way to beat RSA that doesn't involve making an unauthorized copy of anyone.

> But: If your friend finds out about your plan, he might be able to conclude that his job — factoring — is hopeless. If RSA is unbreakable, then factoring must be hard as well.

Cryptographic Primitives

- Pseudorandom generators
- Pseudorandom functions
- Hash functions

Pseudorandomness

Pseudorandom generator G(s)

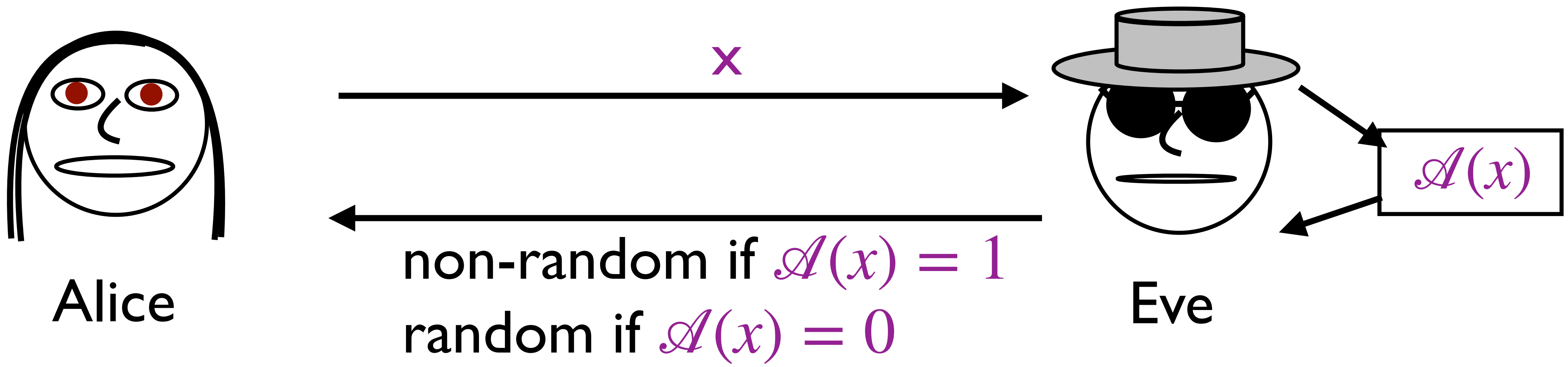
- One input s, "seed"
- Output looks like a random string when s unknown
- Output should be longer than the seed
- Stream cipher is a more flexible version

Pseudorandom function $F_k(r)$

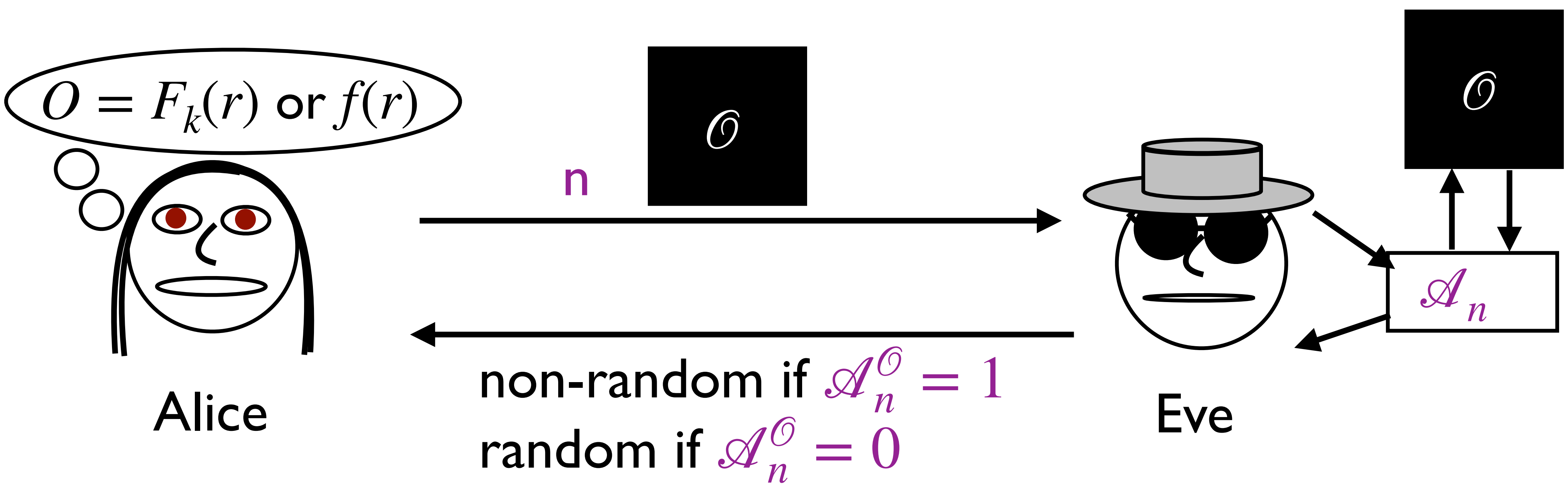
- Two inputs: k (key) and r
- For fixed but unknown k, looks like a random function of r
- Output can be the same size or smaller than the input
- Block cipher is a fixed-size version, but must be a permutation (with computable inverse for known k)

Pseudorandomness Games

Pseudorandom generator:



Pseudorandom function:



This class is being recorded

Hash Functions

Hash function H(x)

- Unlike pseudorandom functions and generators, function and input are often both known (i.e., no key)
- Output must be shorter than the input
- Main cryptographic property is collision resistance, meaning it is hard to find two inputs x_1, x_2 with the same output $H(x_1) = H(x_2)$
- Does not need to look like a random function
- But is often modeled as a random oracle anyway
- Note: but (unlike a pseudorandom function) it is always easy to distinguish from a truly random function since we can just test it on specific inputs
- Often take arbitrary-length inputs

Cryptographic Protocols

- Private-key encryption
- Public-key encryption
- Key agreement
- Key encapsulation mechanism (KEM)
- MAC
- Authenticated encryption
- Digital signature
- Identification protocol

Cryptographic Protocols Comparison

Protocol	Purpose	Pub./Priv.	Interactive?
Private-key encryption	Encryption	Private	No
Public-key encryption	Encryption	Public	No
Key agreement	Gen. key	None	Yes
KEM	Gen. key	Public	No
MAC	Authenticate	Private	No
Authenticated encrypt.	Enc. + Auth.	Private	No
Digital signature	Authenticate	Public	No
Identification protocol	Authenticate	Public	Yes

Public Key vs. Private Key

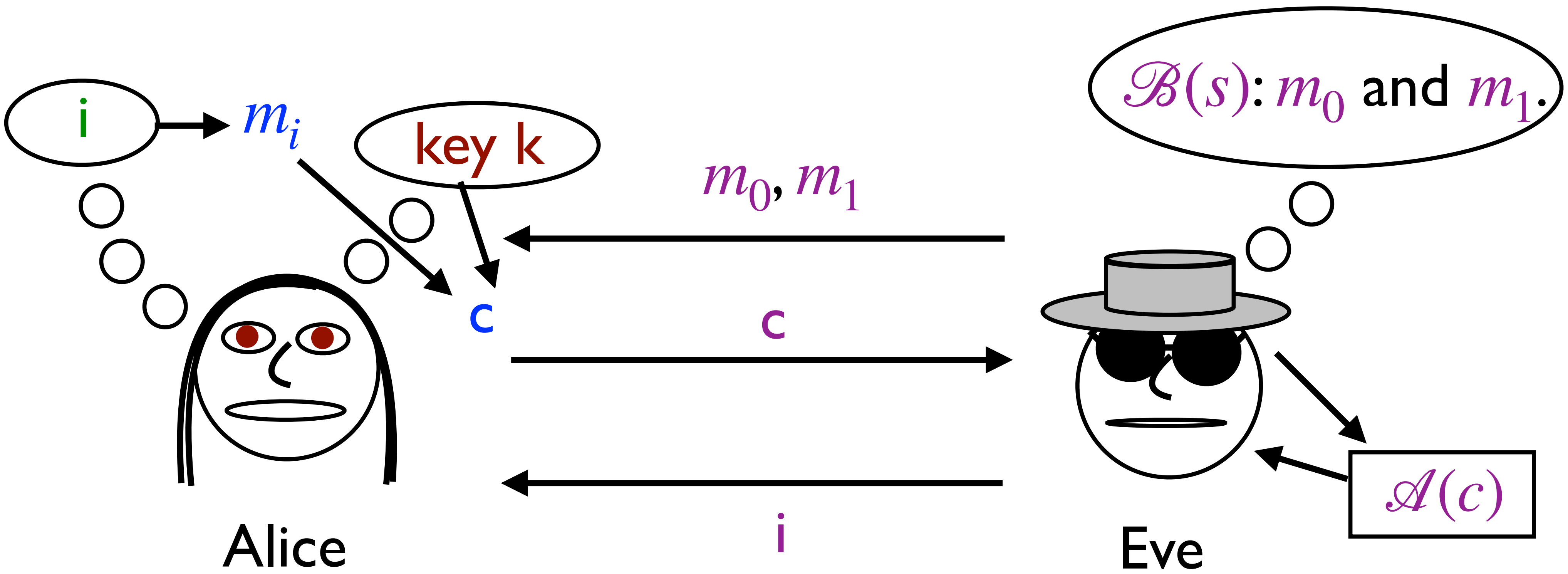
Private Key	Public Key
Highly secure (short keys sufficient)	Less secure (long keys needed)
Very efficient	Slow
Symmetric (same key)	Asymmetric (public and private key pair)
Need different key for each pair of people	Public key can be distributed to many people
	May have additional properties

This class is being recorded

Encryption Security Definitions

Eve chooses two messages m_0 and m_1 and must identify an encryption of one of them.

EAV security:

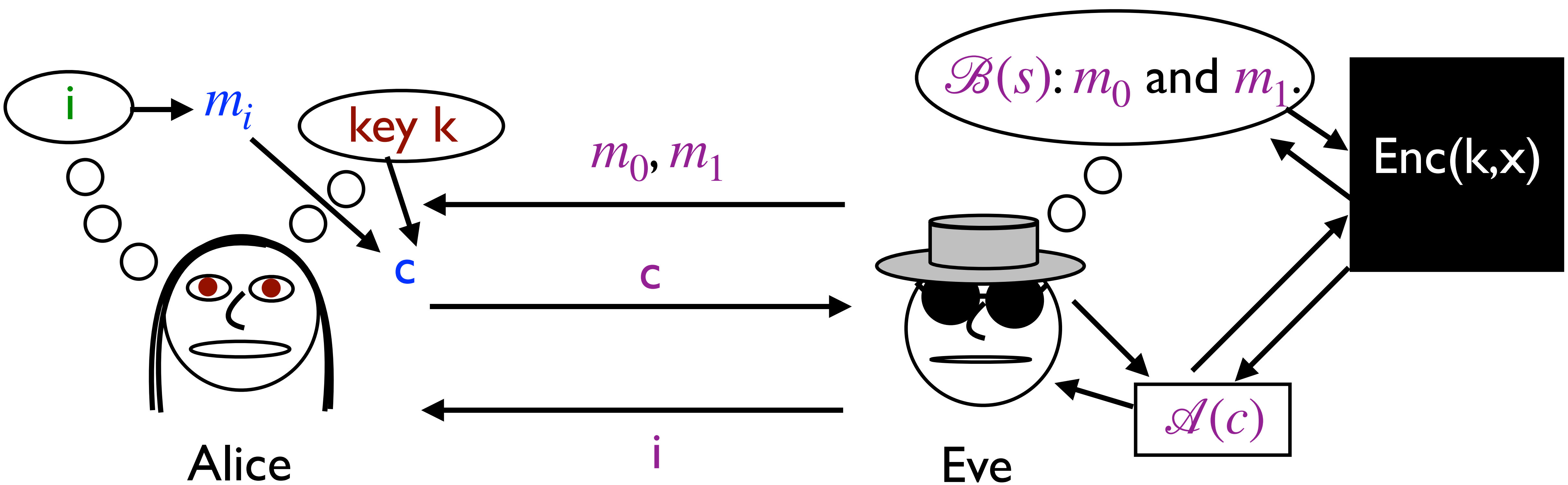


This class is being recorded

Encryption Security Definitions

Eve chooses two messages m_0 and m_1 and must identify an encryption of one of them.

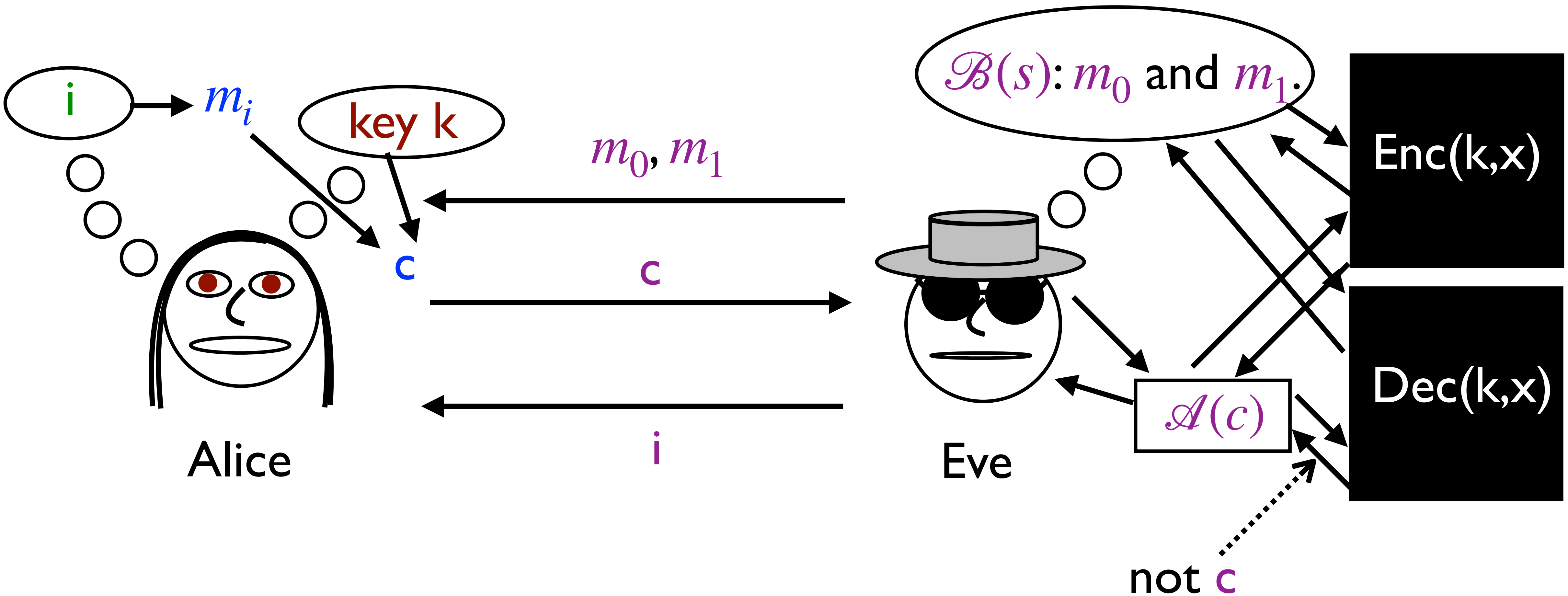
CPA security (private key):



Encryption Security Definitions

Eve chooses two messages m_0 and m_1 and must identify an encryption of one of them.

CCA security (private key):

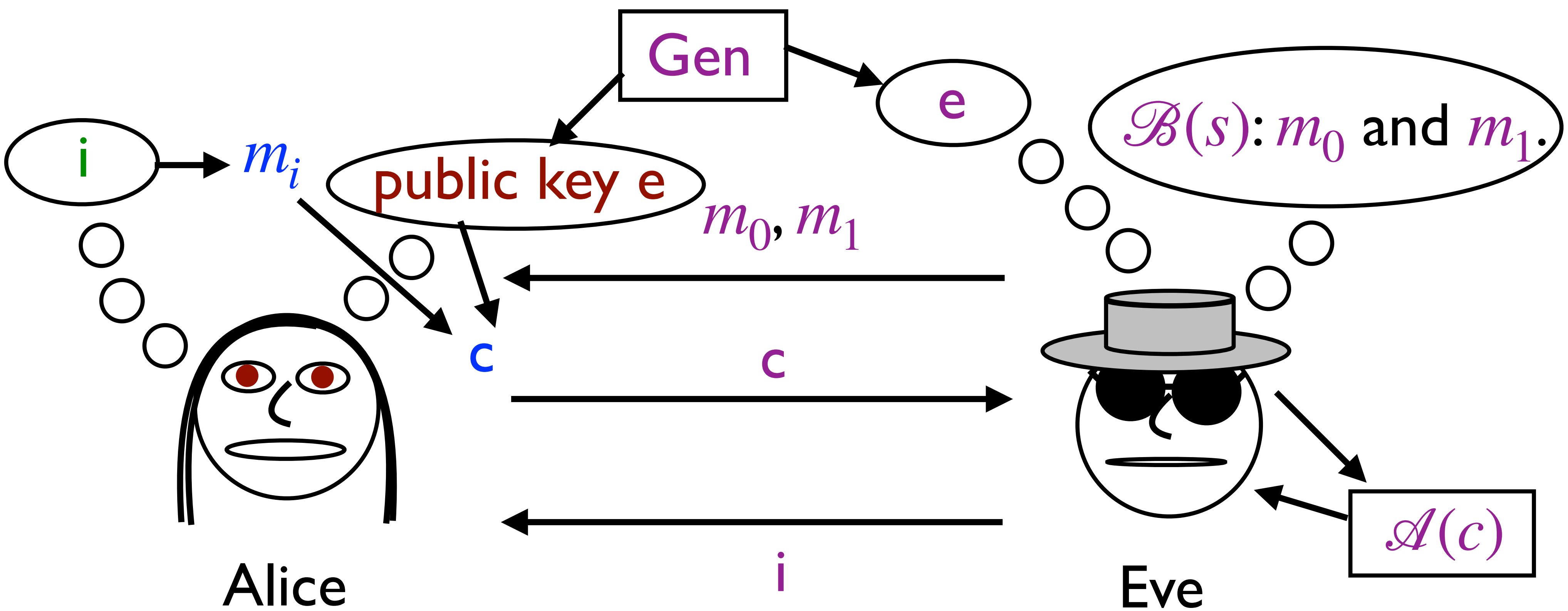


This class is being recorded

Encryption Security Definitions

Eve chooses two messages m_0 and m_1 and must identify an encryption of one of them.

CPA security (public key):

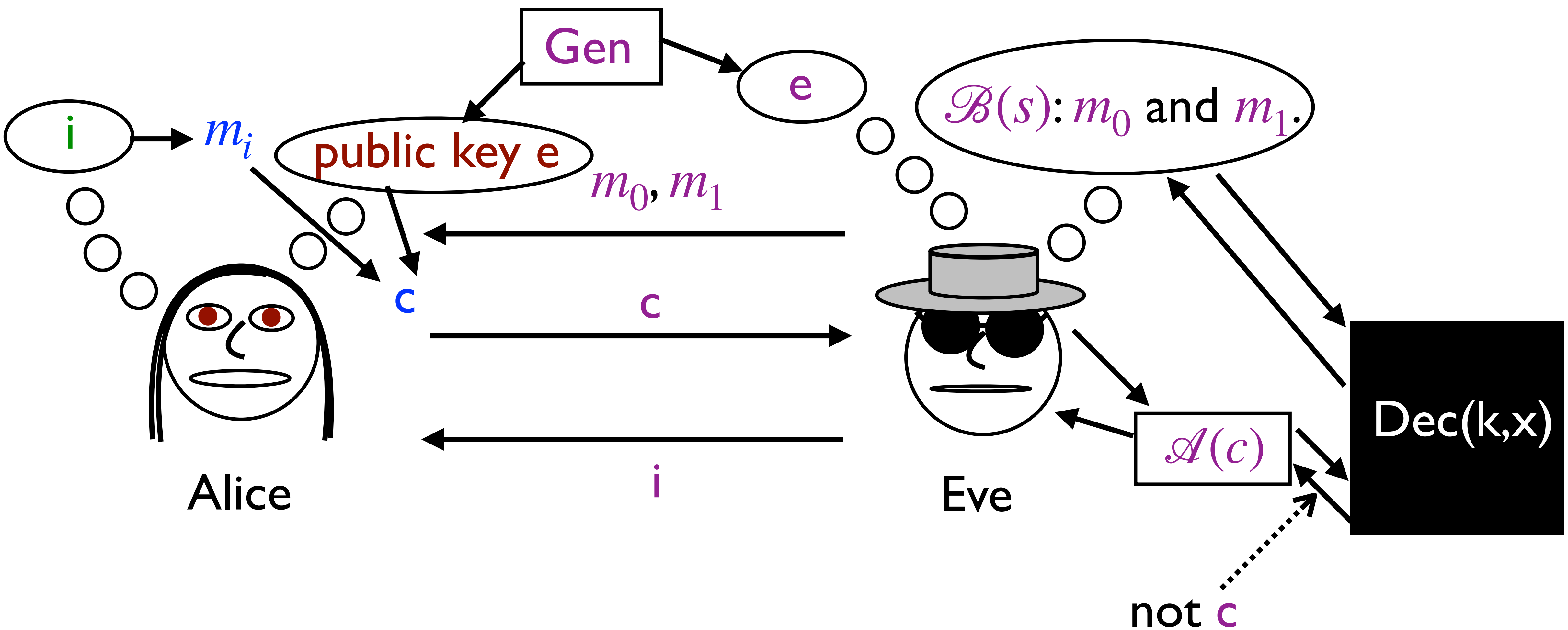


This class is being recorded

Encryption Security Definitions

Eve chooses two messages m_0 and m_1 and must identify an encryption of one of them.

CCA security (public key):

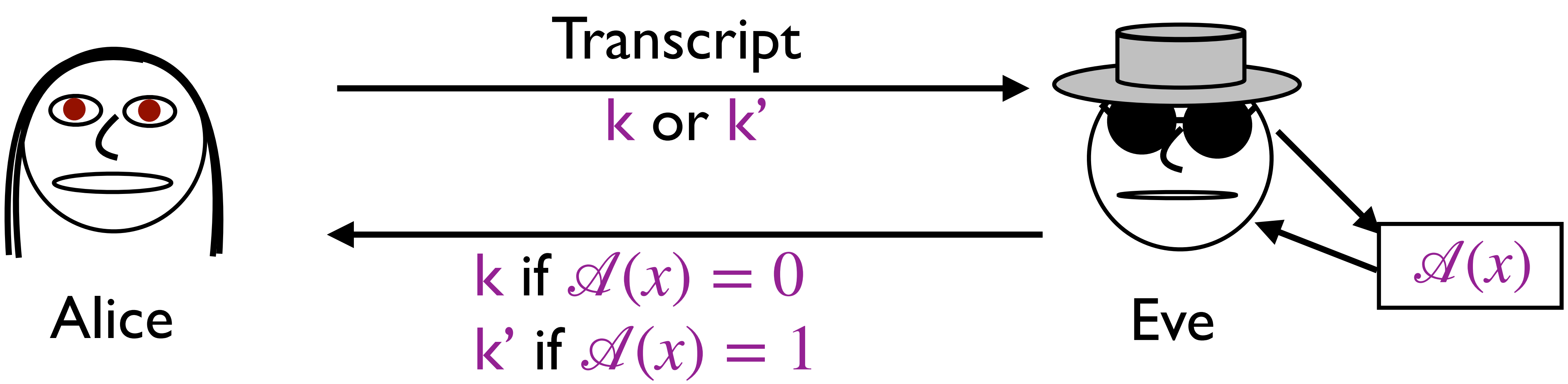


This class is being recorded

Security Definitions for Key Gen.

Eve must distinguish between k generated by the protocol and a uniformly random k'.

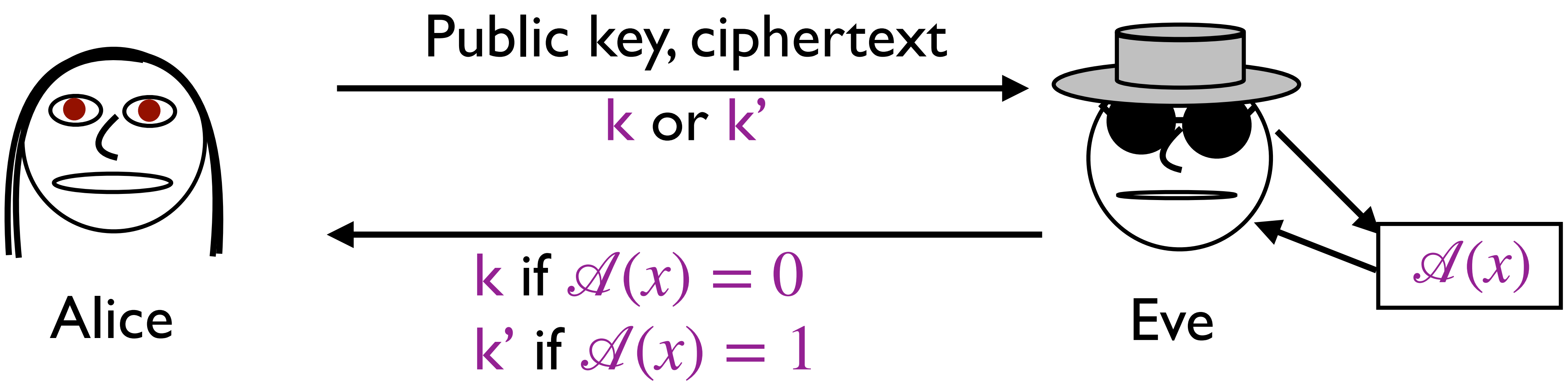
Key agreement security:



Security Definitions for Key Gen.

Eve must distinguish between k generated by the protocol and a uniformly random k'.

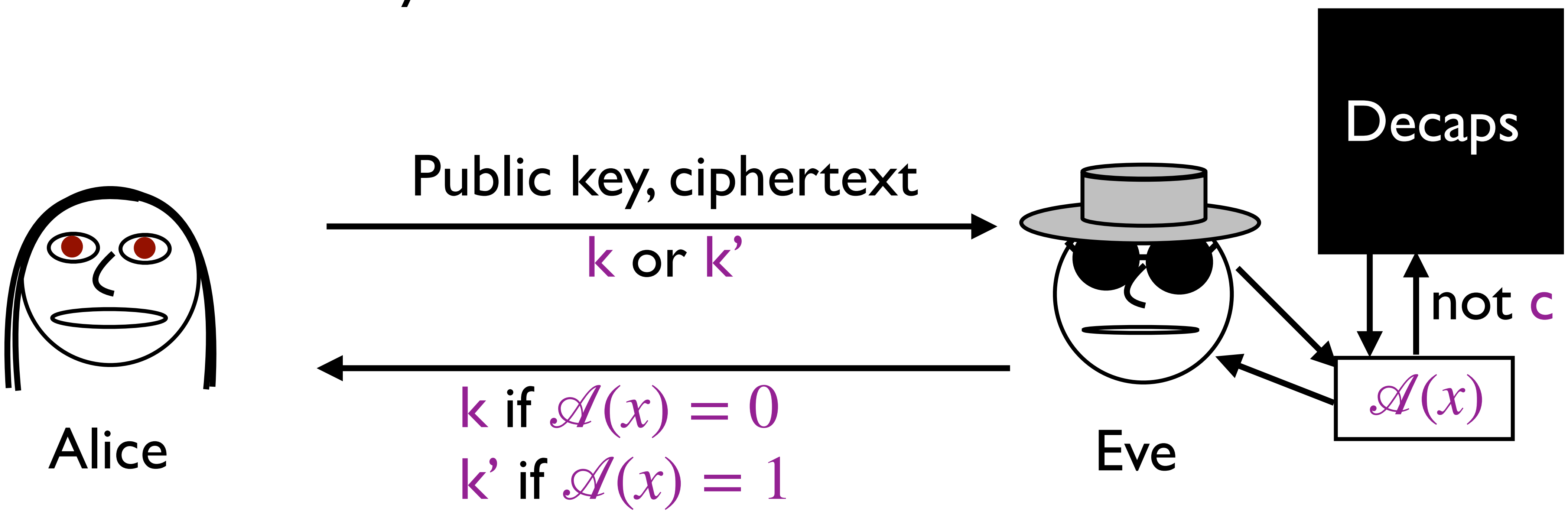
KEM CPA security:



Security Definitions for Key Gen.

Eve must distinguish between k generated by the protocol and a uniformly random k'.

KEM CCA security:

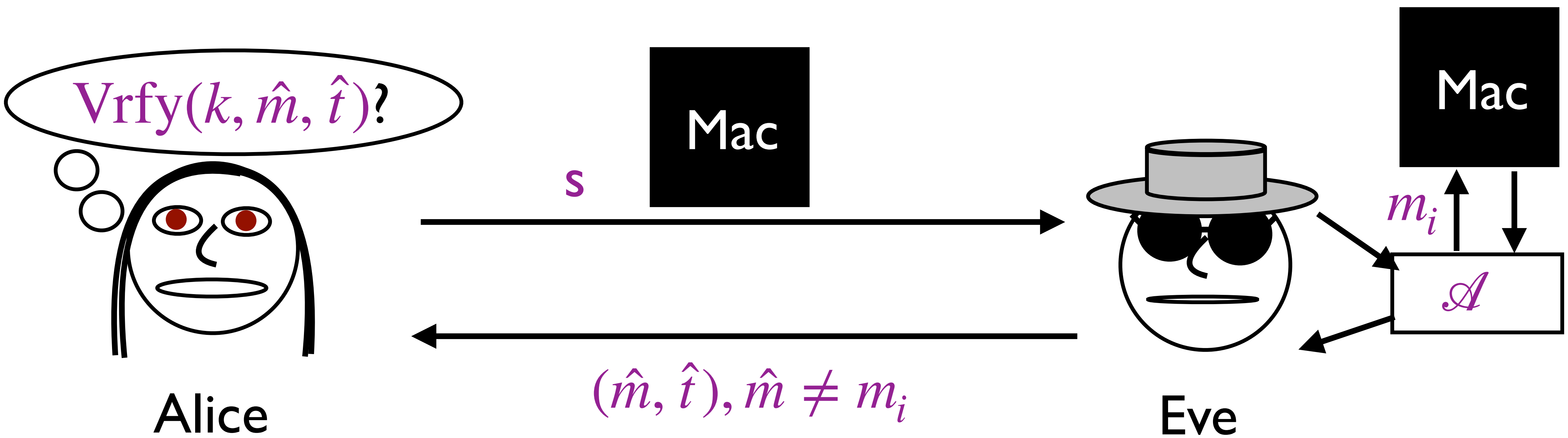


This class is being recorded

Authentication Security Definitions

Eve must forge a message which she hasn't queried to her oracle.

MAC security:

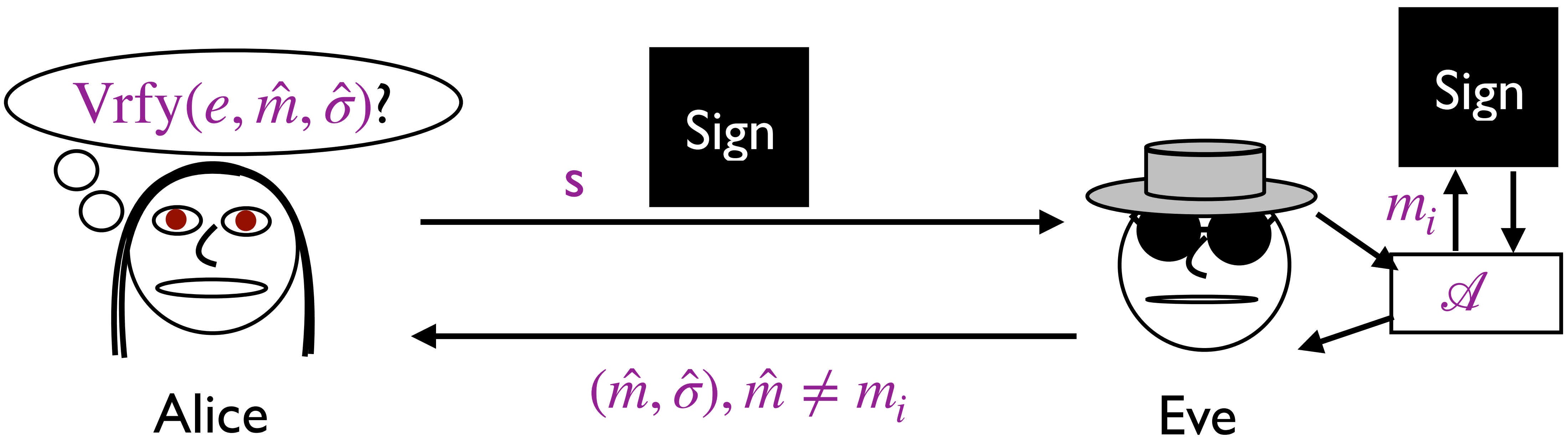


This class is being recorded

Authentication Security Definitions

Eve must forge a message which she hasn't queried to her oracle.

Digital signature security:

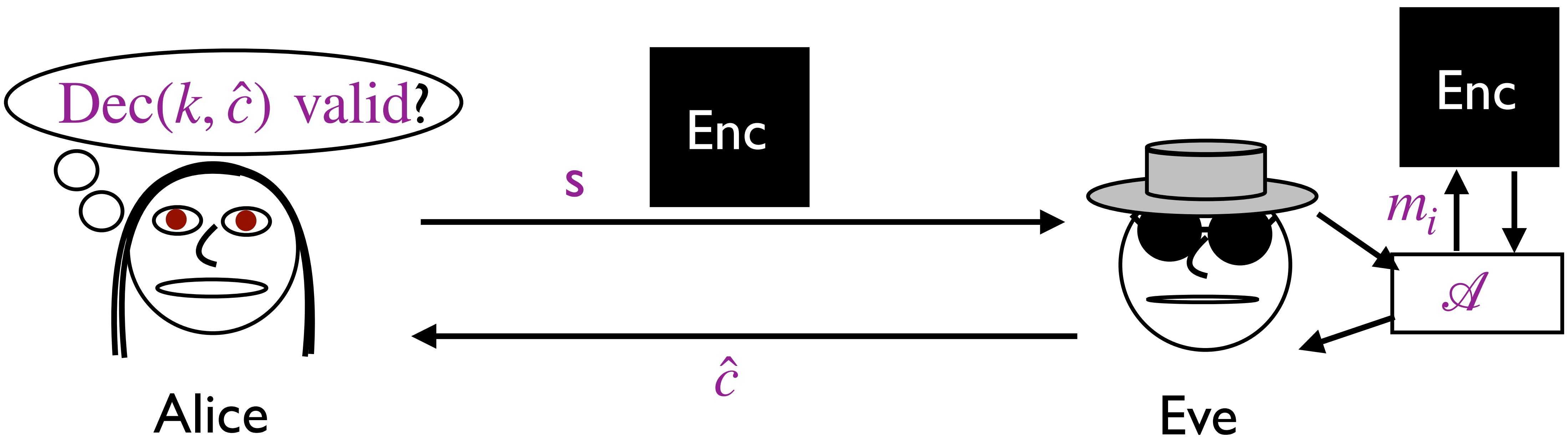


This class is being recorded

Authentication Security Definitions

Eve must forge a message which she hasn't queried to her oracle.

Unforgeability (for encryption):



Recall that authenticated encryption is CCA security plus unforgeability.

General Encryption Constructions

EAV security:

Pseudorandom generator G(s)

Pseudo one-time pad
$c = G(k) \oplus m$

CPA security:

Pseudorandom function $F_k(r)$

$(r, F_k(r) \oplus m)$
Need random IV r to avoid repeating ciphertext

CBC mode or CTR mode for longer messages

CCA security/AE:

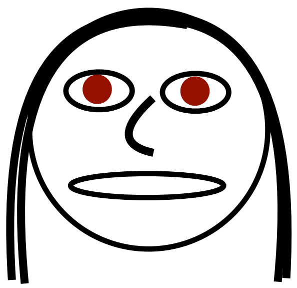
MAC

Encrypt then authenticate

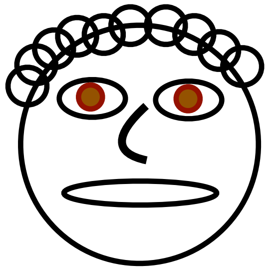
This class is being recorded

KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



Alice

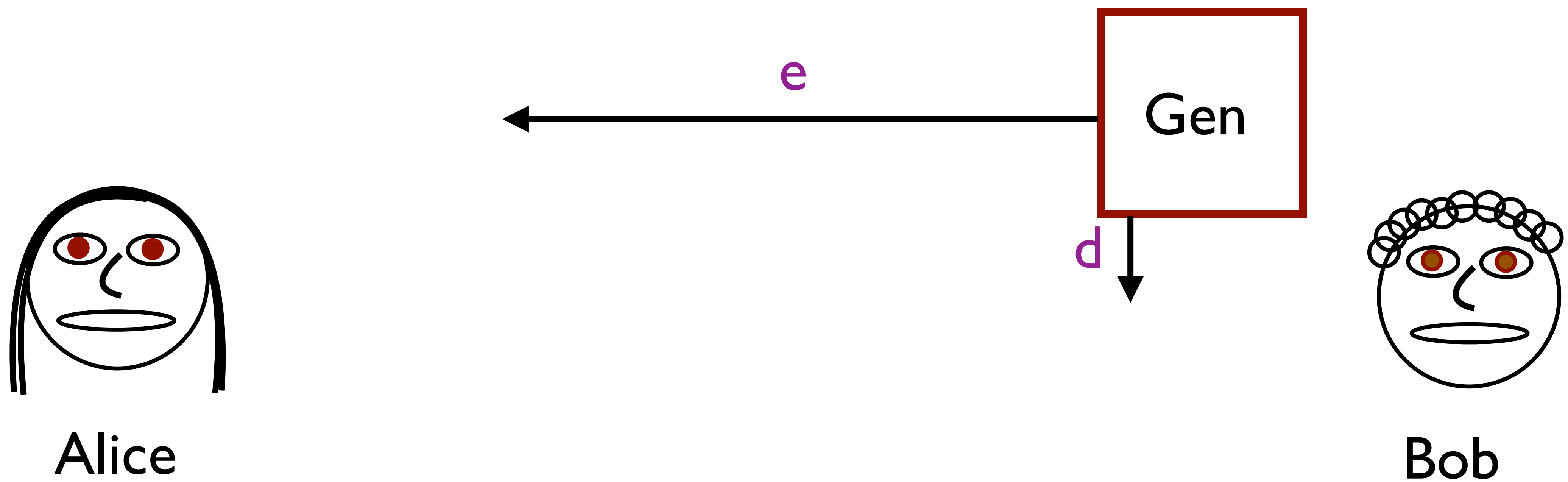


Bob

This class is being recorded

KEM/DEM

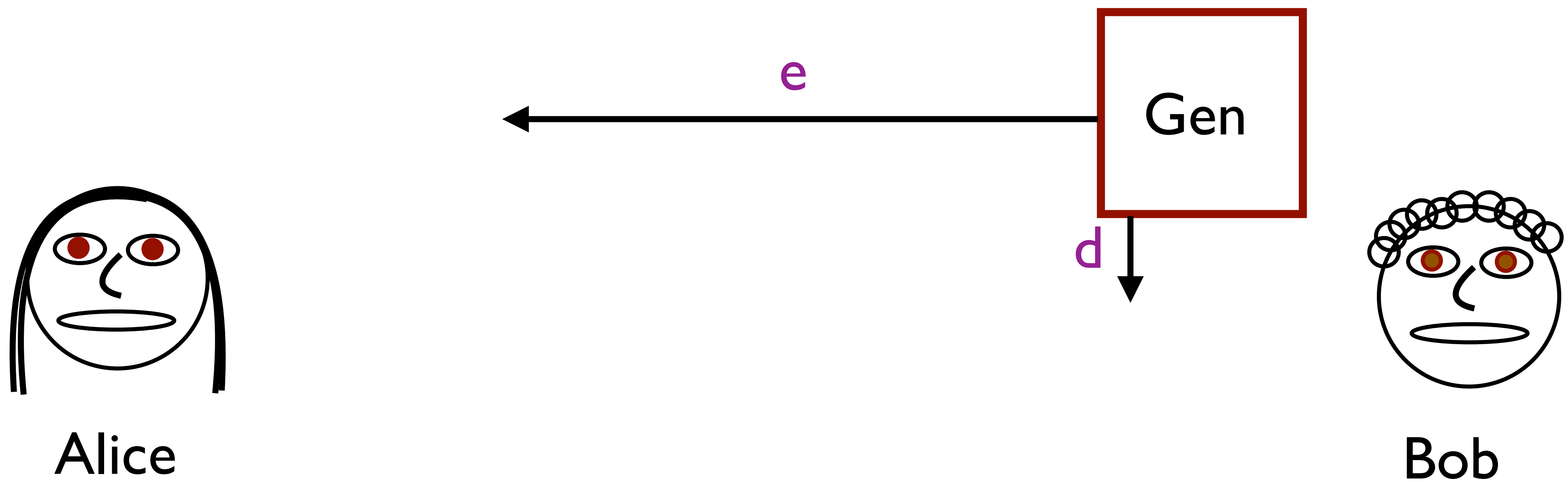
Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



This class is being recorded

KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:

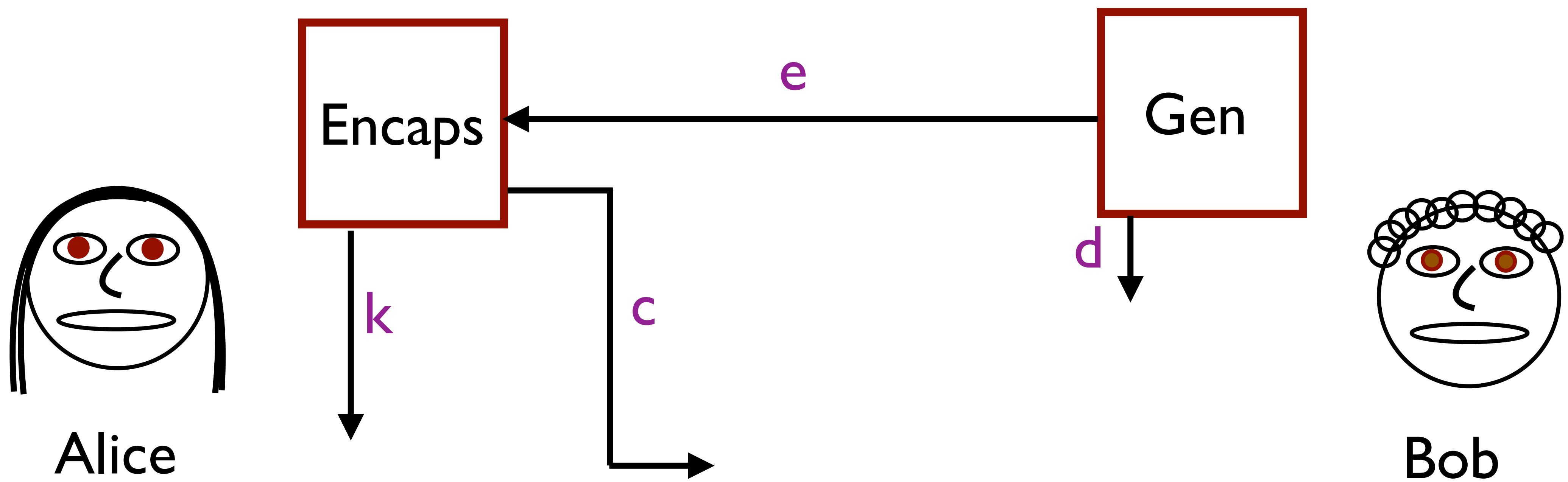


message m

This class is being recorded

KEM/DEM

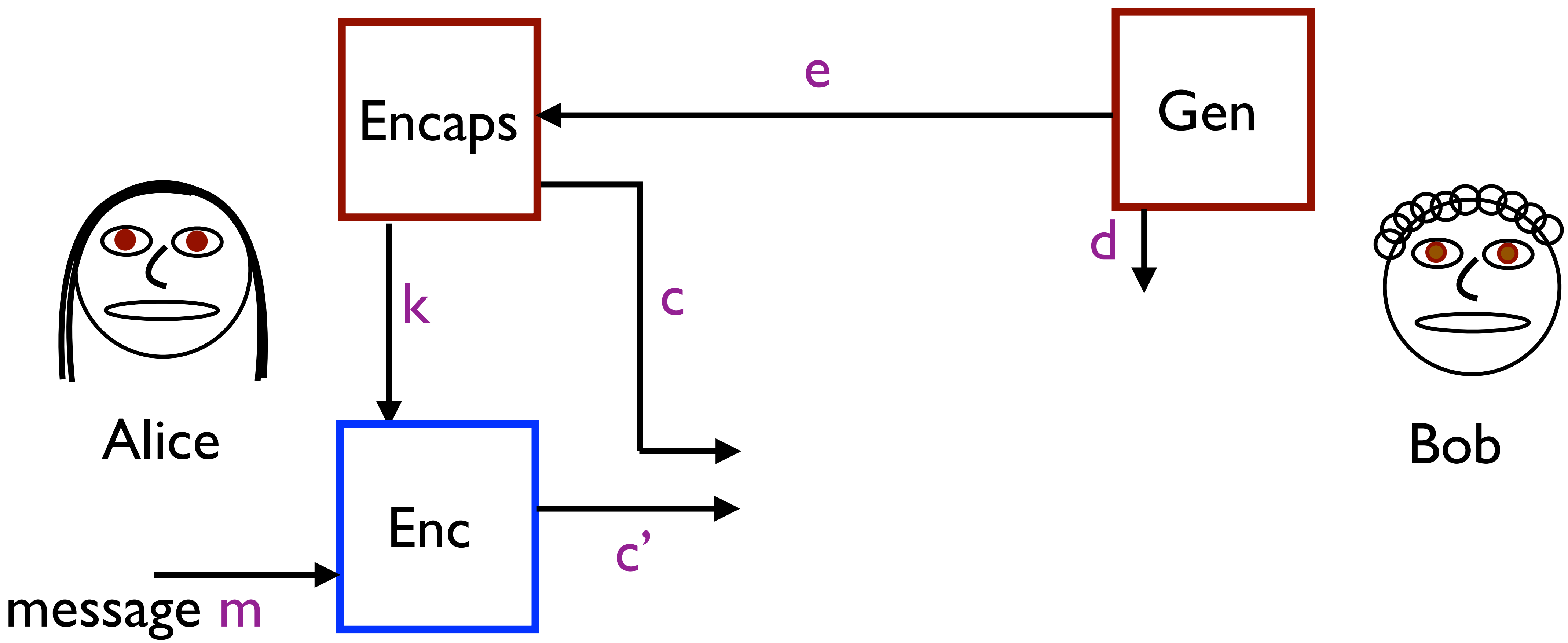
Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



message m

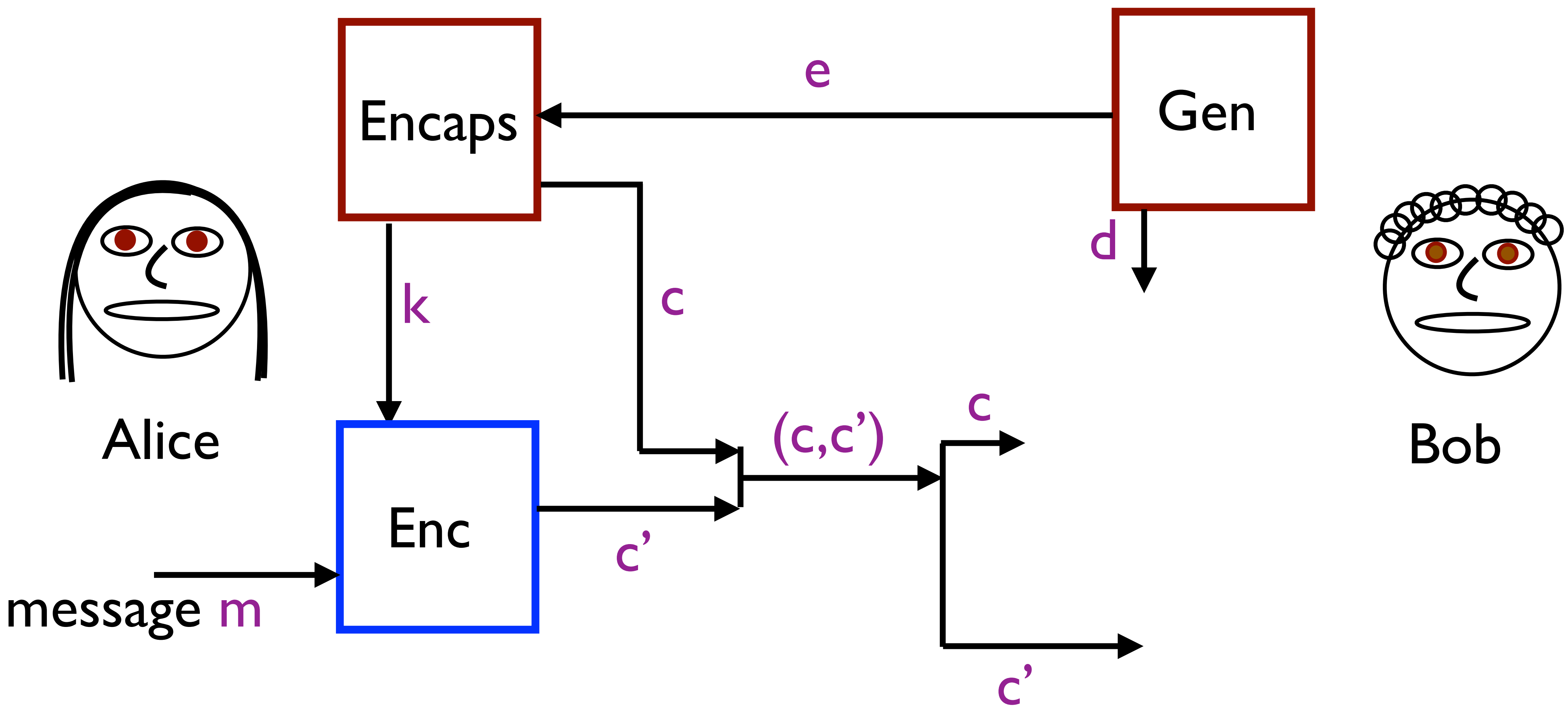
KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



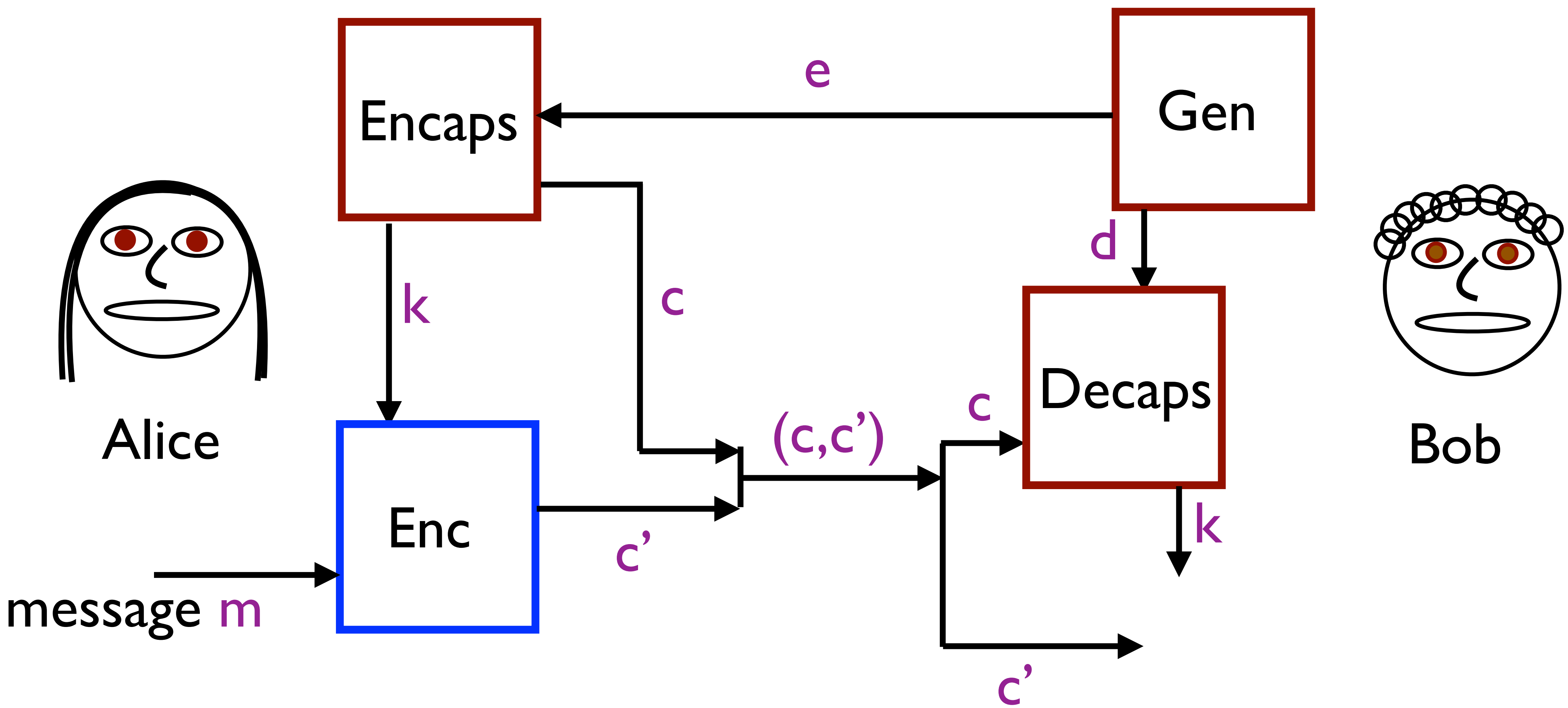
KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



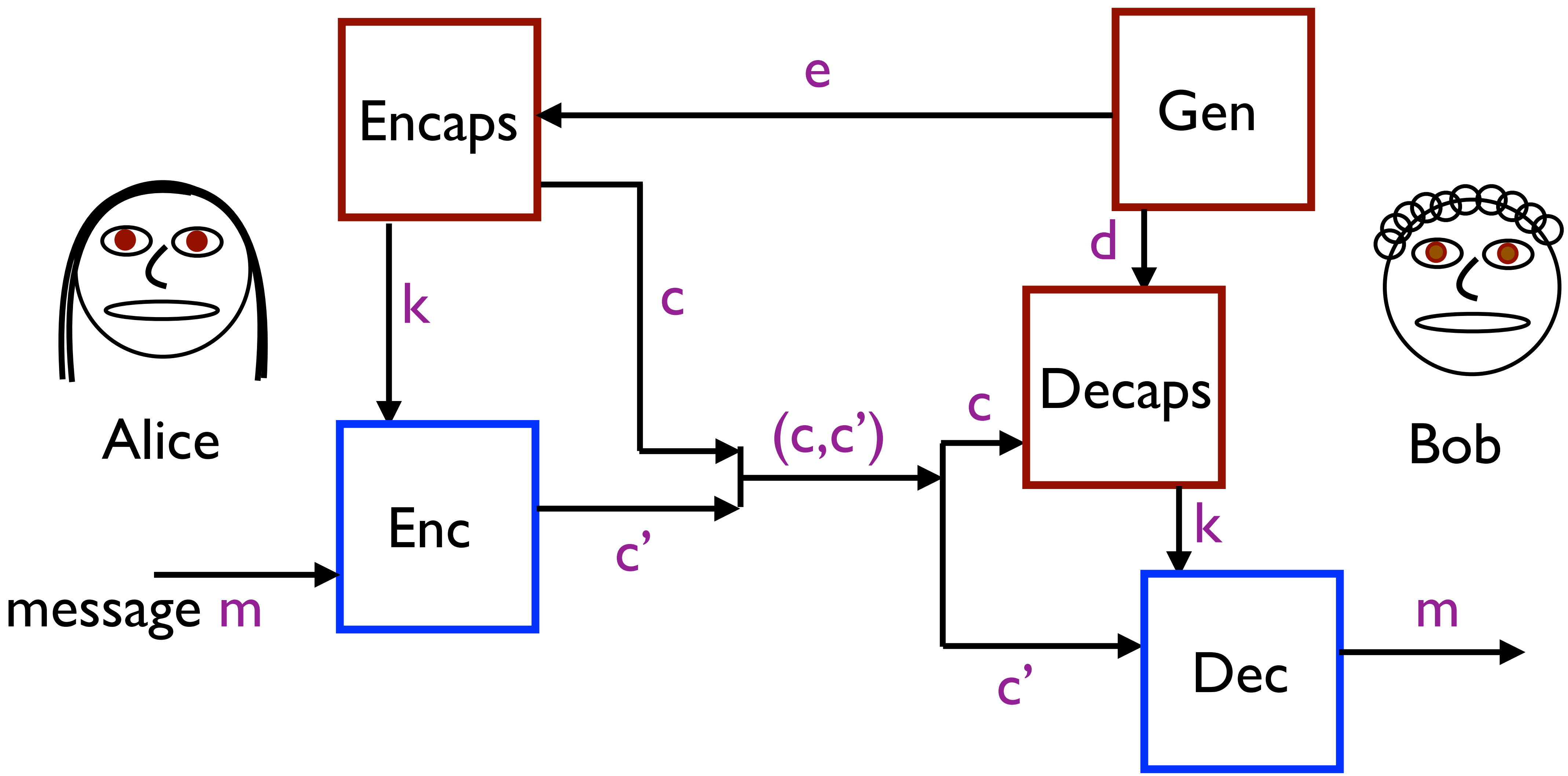
KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



KEM/DEM

Using a KEM, we can effectively upgrade these private-key encryption protocols into public-key encryption protocols:



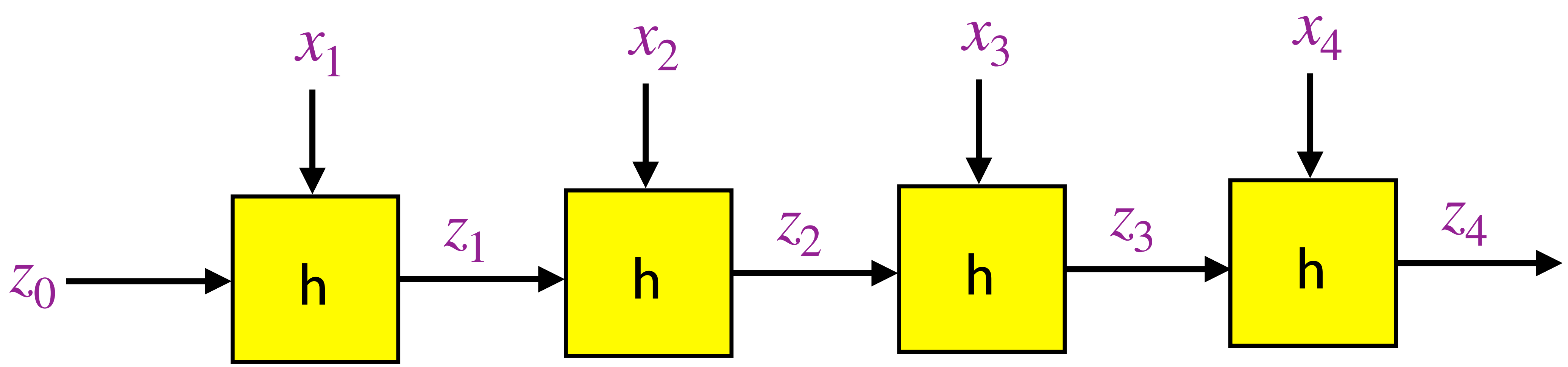
Generic MAC Constructions

Pseudorandom function $F_k(r)$	$\mathrm{Mac}(k, m) = F_k(m)$
CBC-Mac	For longer messages; no IV, include length as first input, tag is only output of last block
Hash-and-Mac	$\mathrm{Mac}(k, H(m))$

This class is being recorded

Hash function construction

Merkle-Damgard construction makes hash functions for arbitrary input out of a compression function of fixed size.



Need to pad the input appropriately.

DH Encryption and Signature

Diffie-Hellman: Alice sends $A = g^a \bmod p$, Bob sends $B = g^b \bmod p$, key is $A^b = B^a = g^{ab} \bmod p$.

El Gamal: Public key is $y = g^b \bmod p$, private key is b, encryption is $my^a \bmod p$ (for secret random a).

DH KEM: Public key is $y = g^b \bmod p$, private key is b, ciphertext is $A = g^a \bmod p$, key is $H(A^b) = H(y^a) = H(g^{ab}) \bmod p$.

DSA: Public key is $y = g^b \bmod p$, private key is b. Signature is (r,s): $s = k^{-1}(H(m) + br) \bmod q$ for random k, $r = g^k \bmod p$.

(Verify by checking that $r = g^{H(m)s^{-1}} y^{rs^{-1}} \bmod p$)

RSA and DSA signatures

RSA encryption: Public key is (N, e), private key is d such that $de = 1 \bmod \varphi(N)$. Encryption is $\tilde{m}^e \bmod N$ (with m appropriately padded to $\tilde{m}$).

RSA KEM: Public key is (N, e), private key is d such that $de = 1 \bmod \varphi(N)$. Encryption is $x^e \bmod N$, key is $H(x)$.

RSA signature: Public key is (N, e), private key is d such that $de = 1 \bmod \varphi(N)$. Signature is $\sigma = H(m)^d \bmod N$.