CMSC/Math 456: Cryptography (Fall 2023)

Lecture 24

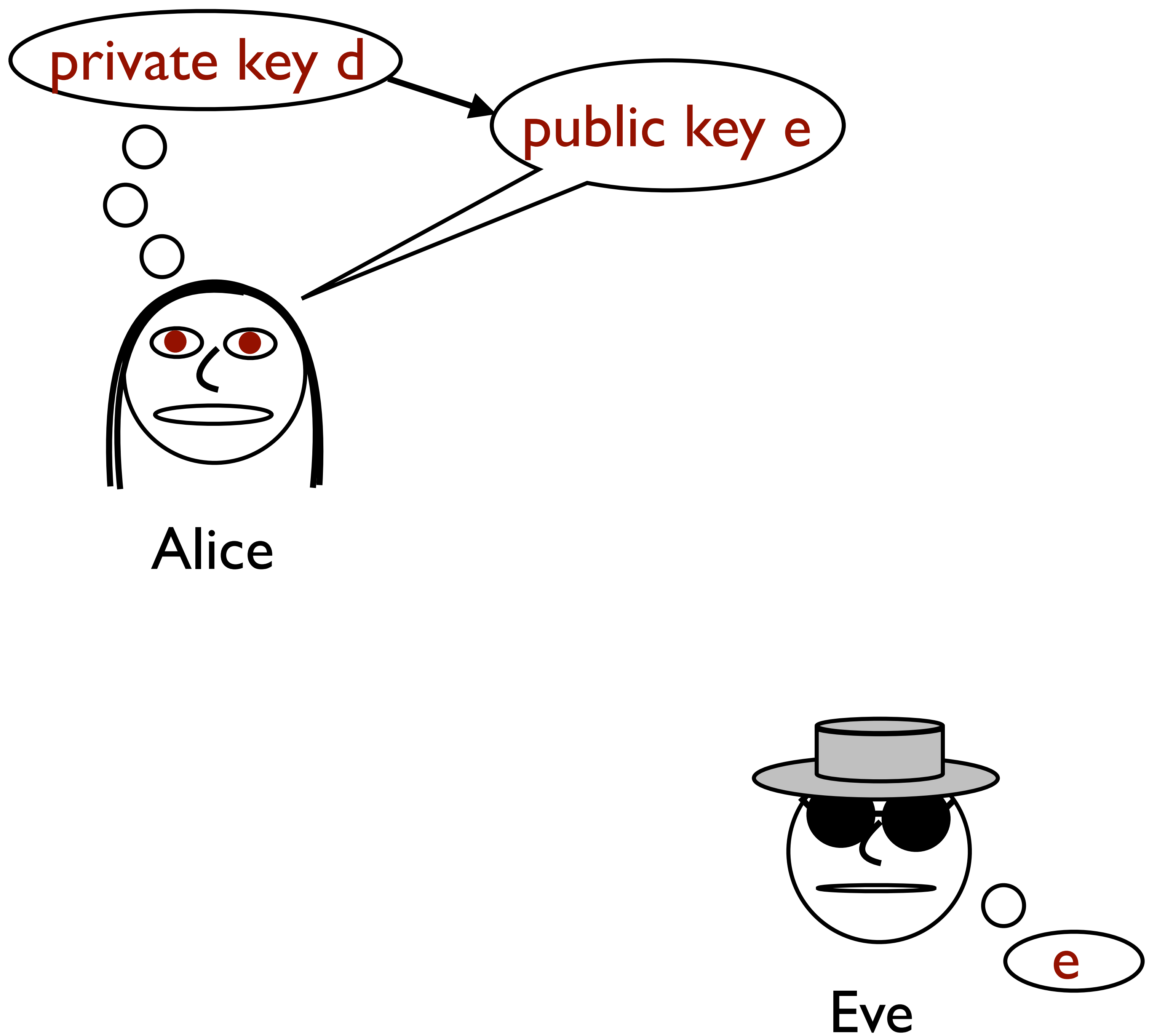
Daniel Gottesman

Administrative

Problem set #9 is due Thursday at noon. There will be one more problem set assigned on Thursday.

Digital Signatures

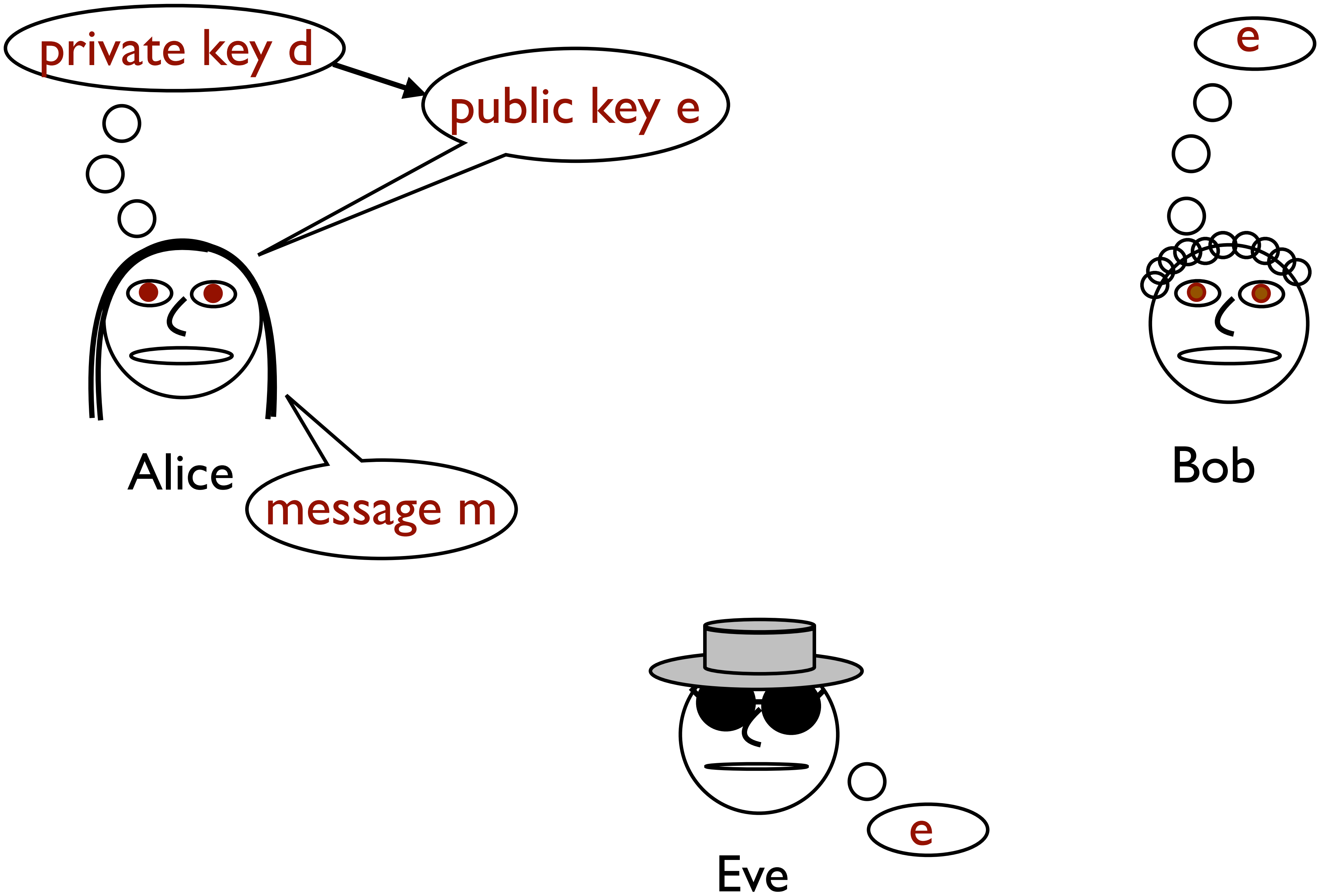
Digital signatures are a public key version of MACs.



This class is being recorded

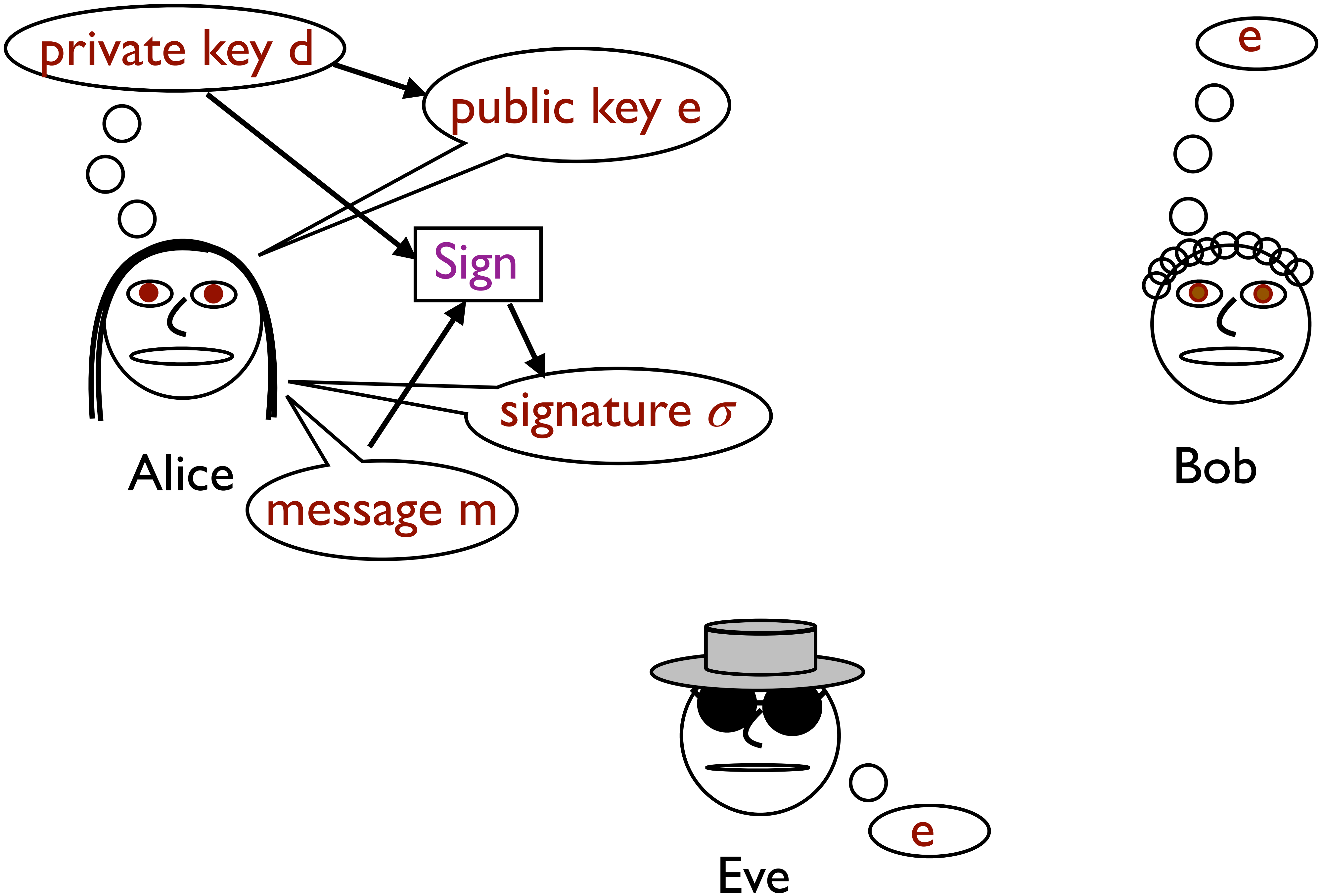
Digital Signatures

Digital signatures are a public key version of MACs.



Digital Signatures

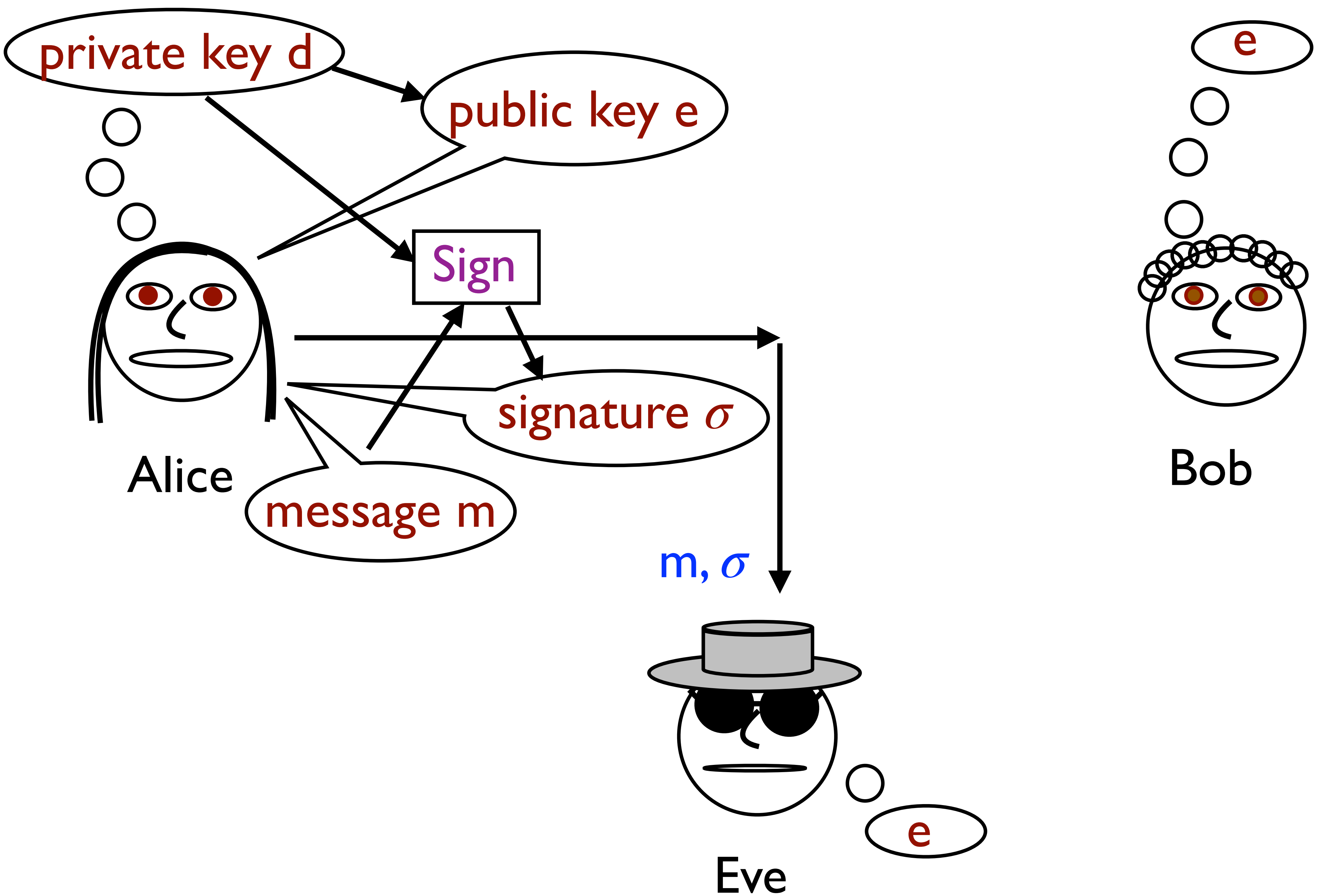
Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signatures

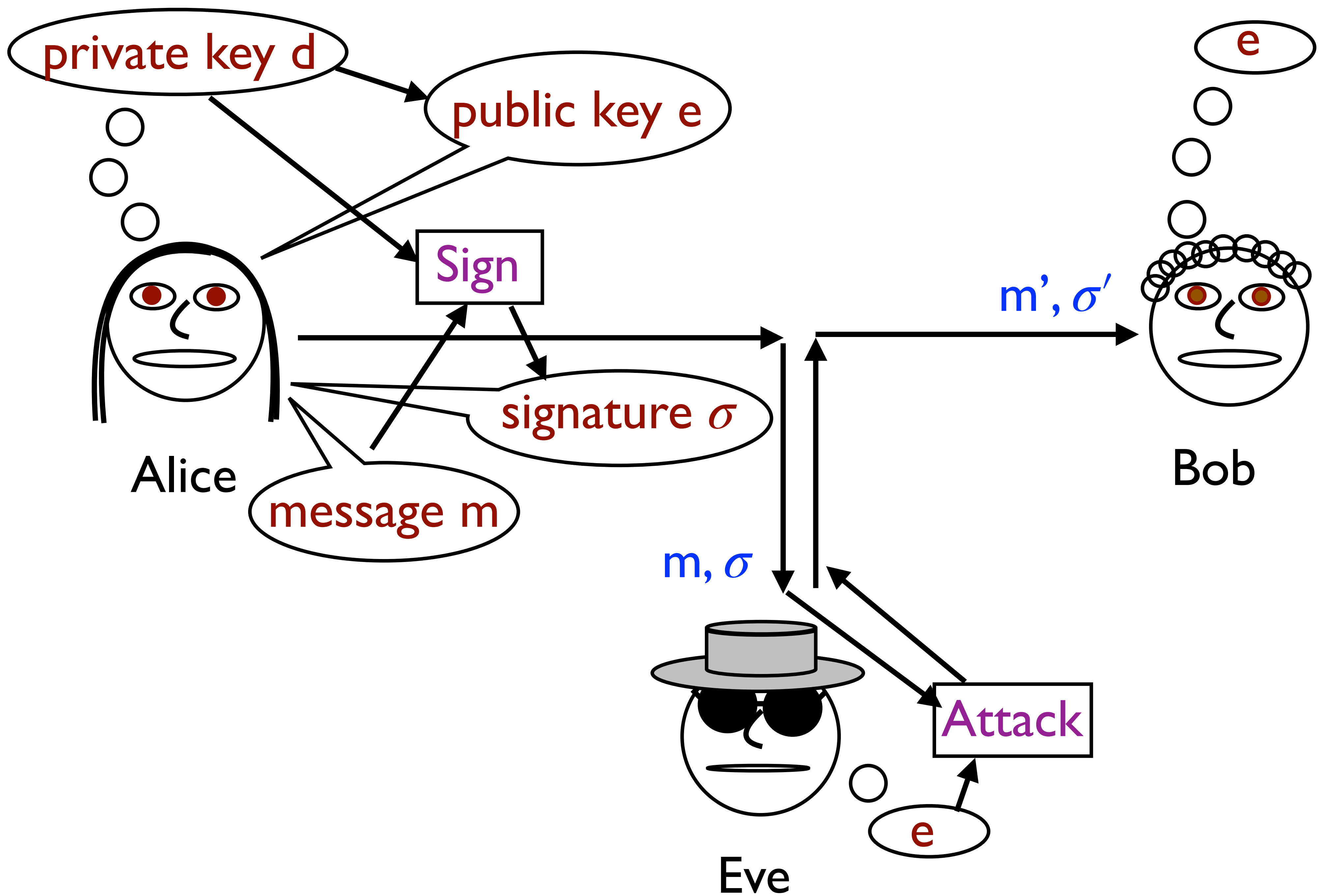
Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signatures

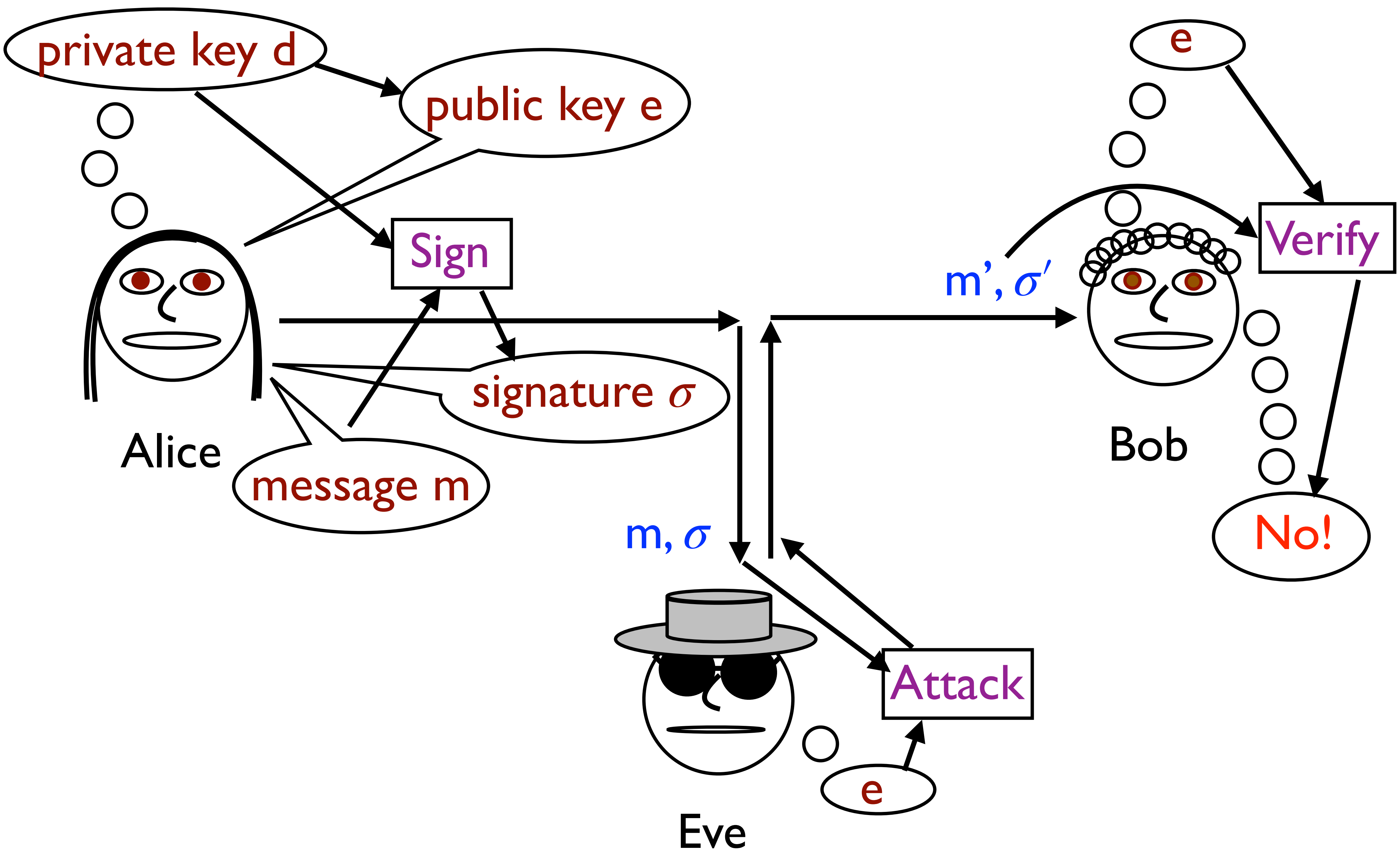
Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signatures

Digital signatures are a public key version of MACs.



This class is being recorded

Digital Signature Definition

Definition: A digital signature is a set of three probabilistic polynomial-time algorithms (Gen, Sign, Vrfy):

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

Sign is the signing algorithm. It takes as input the private key d and a message $m \in \{0,1\}^*$ and outputs a signature $\sigma \in \{0,1\}^*$.

Vrfy is the verification algorithm. It takes as input the public key e and (m, σ) and outputs "valid" or "invalid."

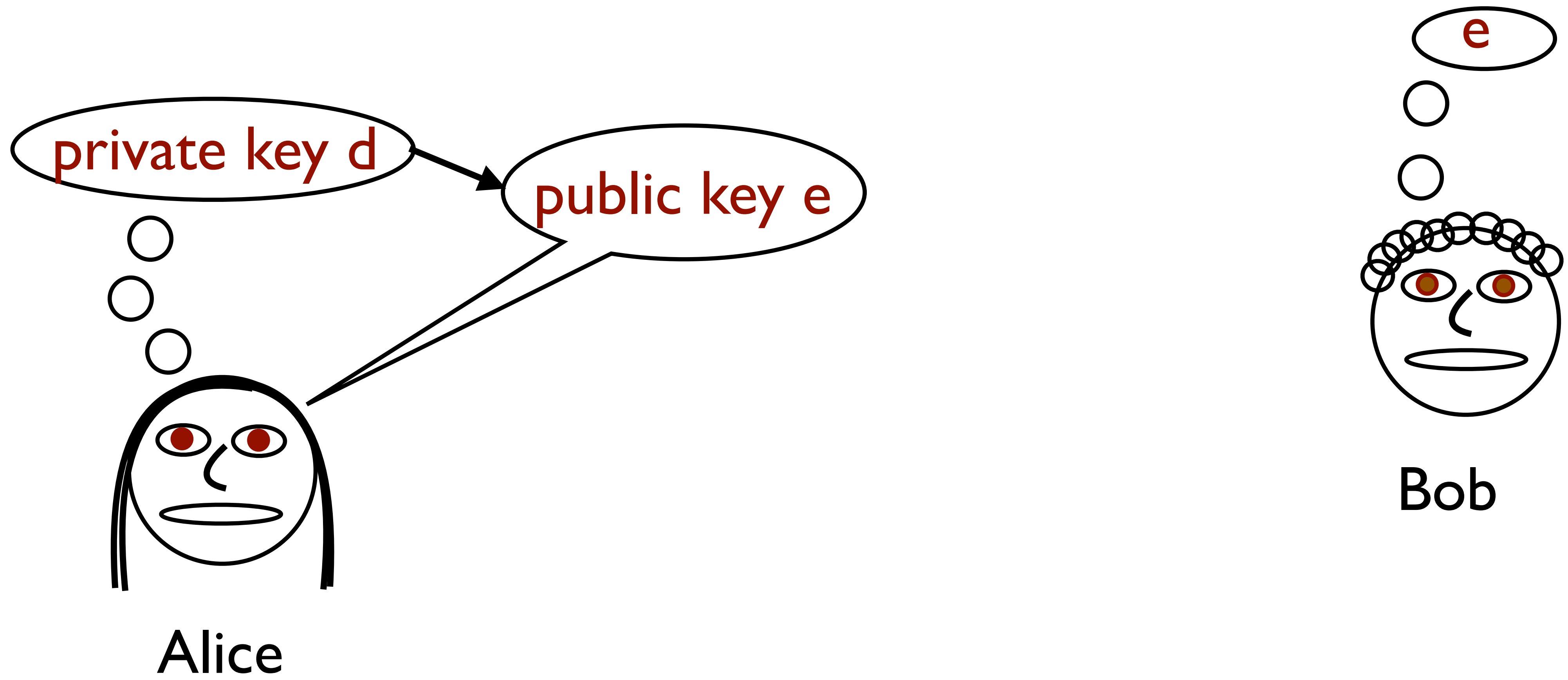
The digital signature scheme is correct if

$$\mathrm{Vrfy}(e, m, \mathrm{Sign}(d, m)) = \mathrm{valid}$$

Note: Unlike a MAC, Vrfy cannot just generate a new signature, since that would require the private key.

This class is being recorded

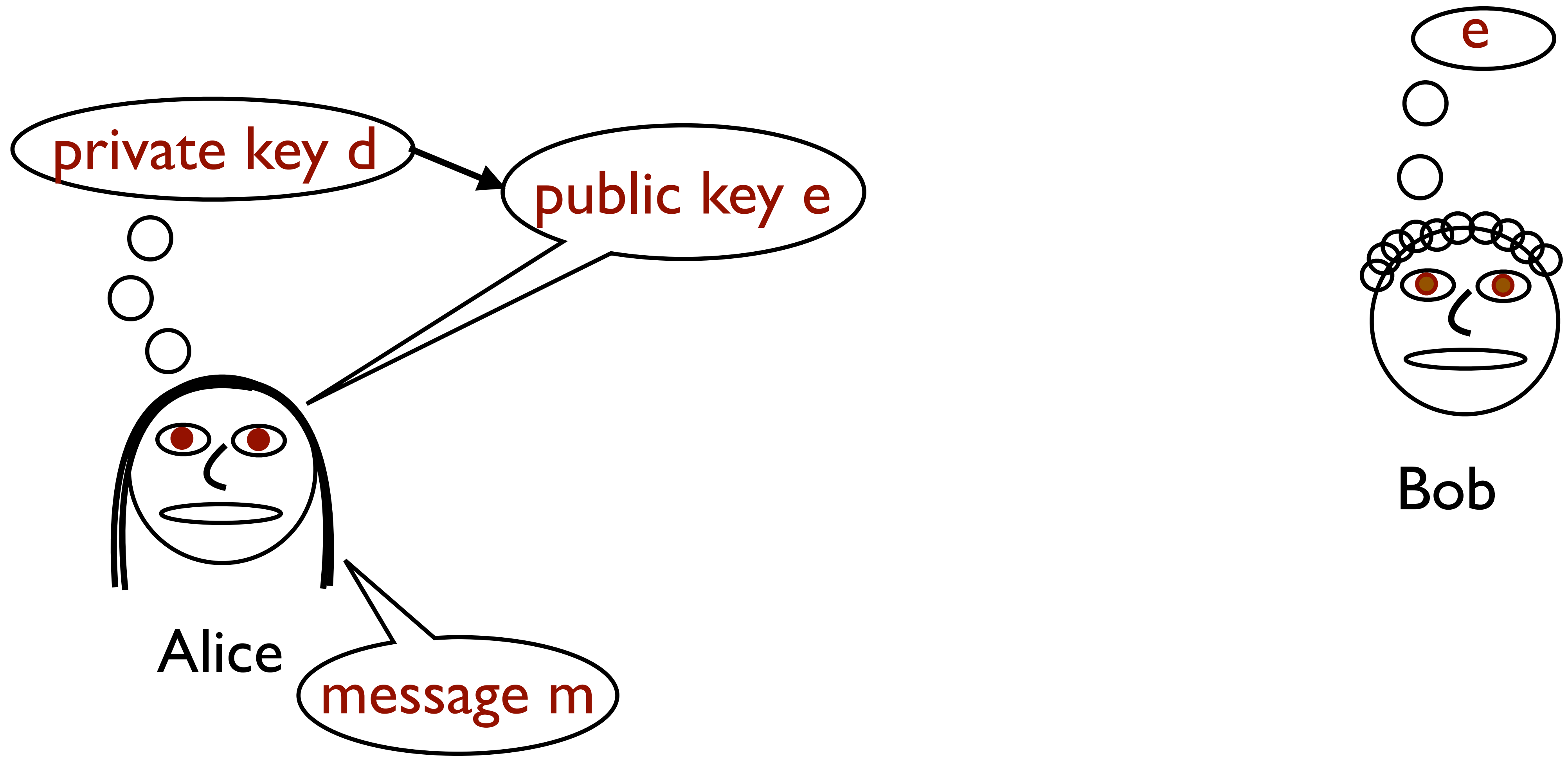
Transferability and Non-Repudiation



Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

This class is being recorded

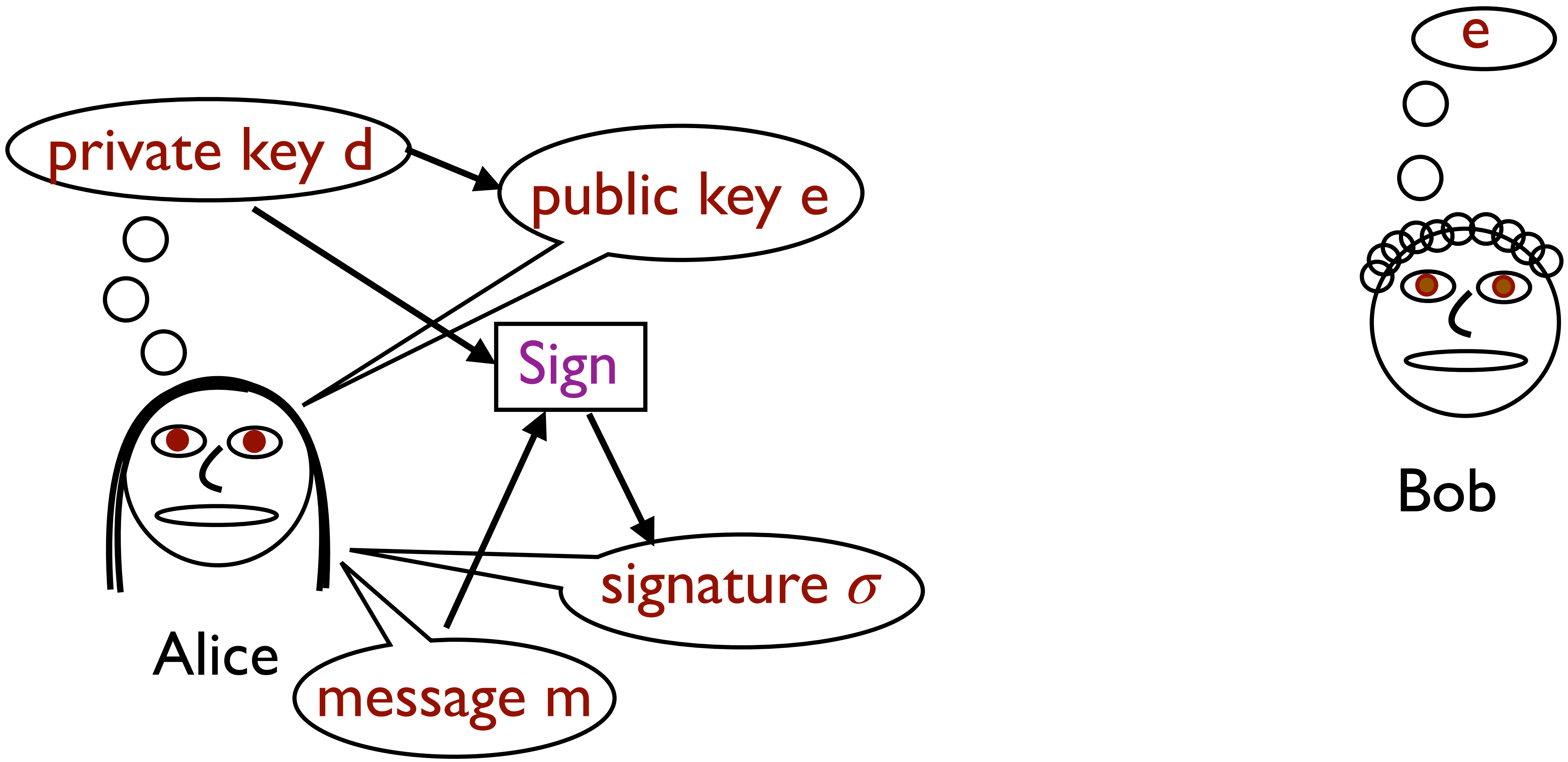
Transferability and Non-Repudiation



Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

This class is being recorded

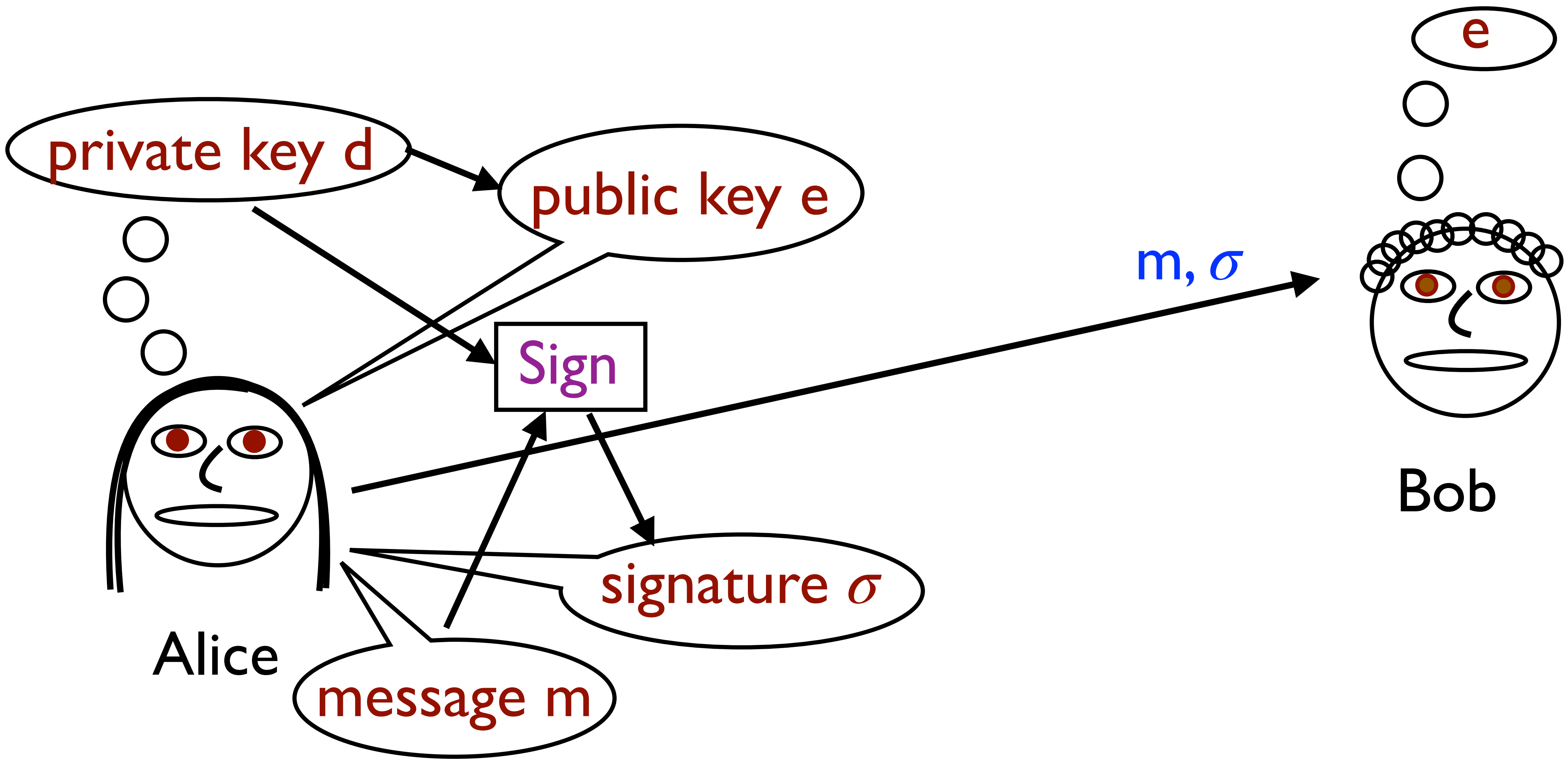
Transferability and Non-Repudiation



Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

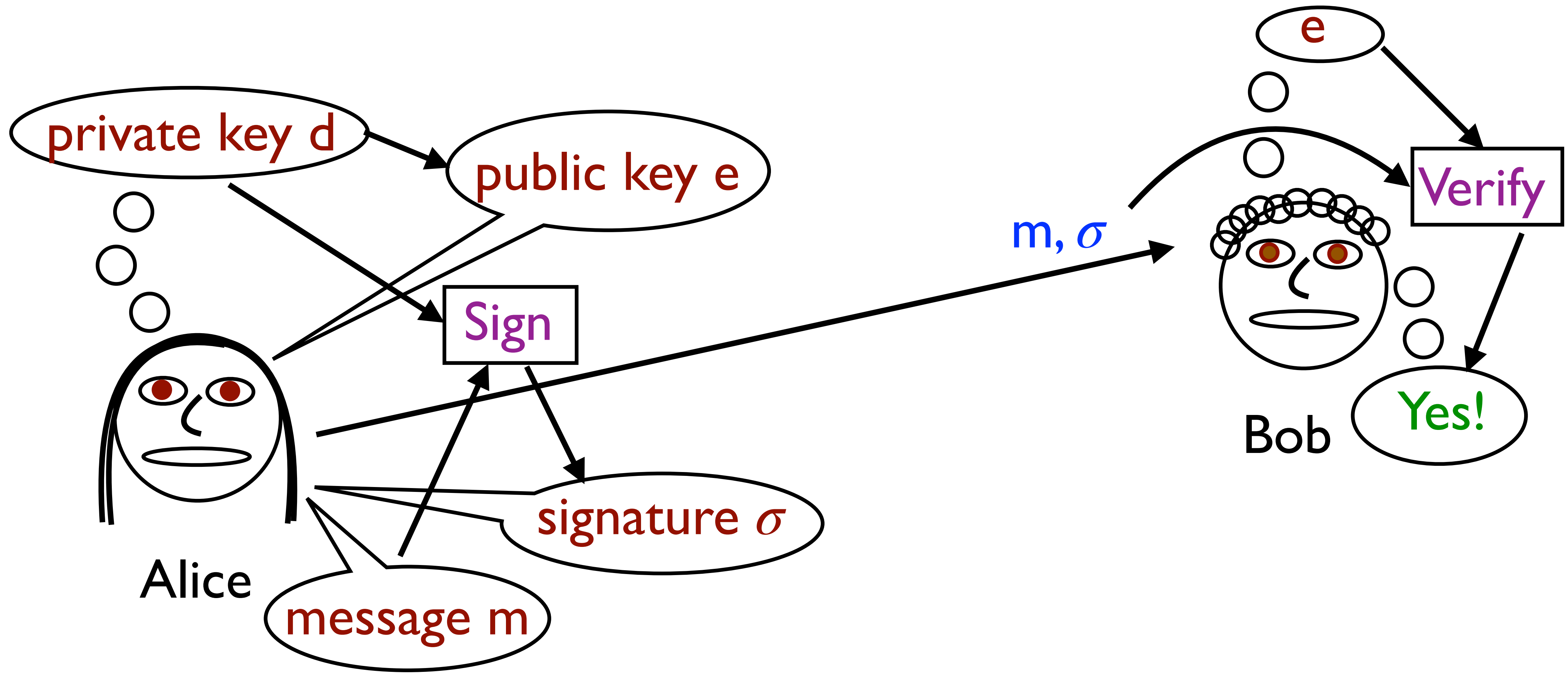
This class is being recorded

Transferability and Non-Repudiation



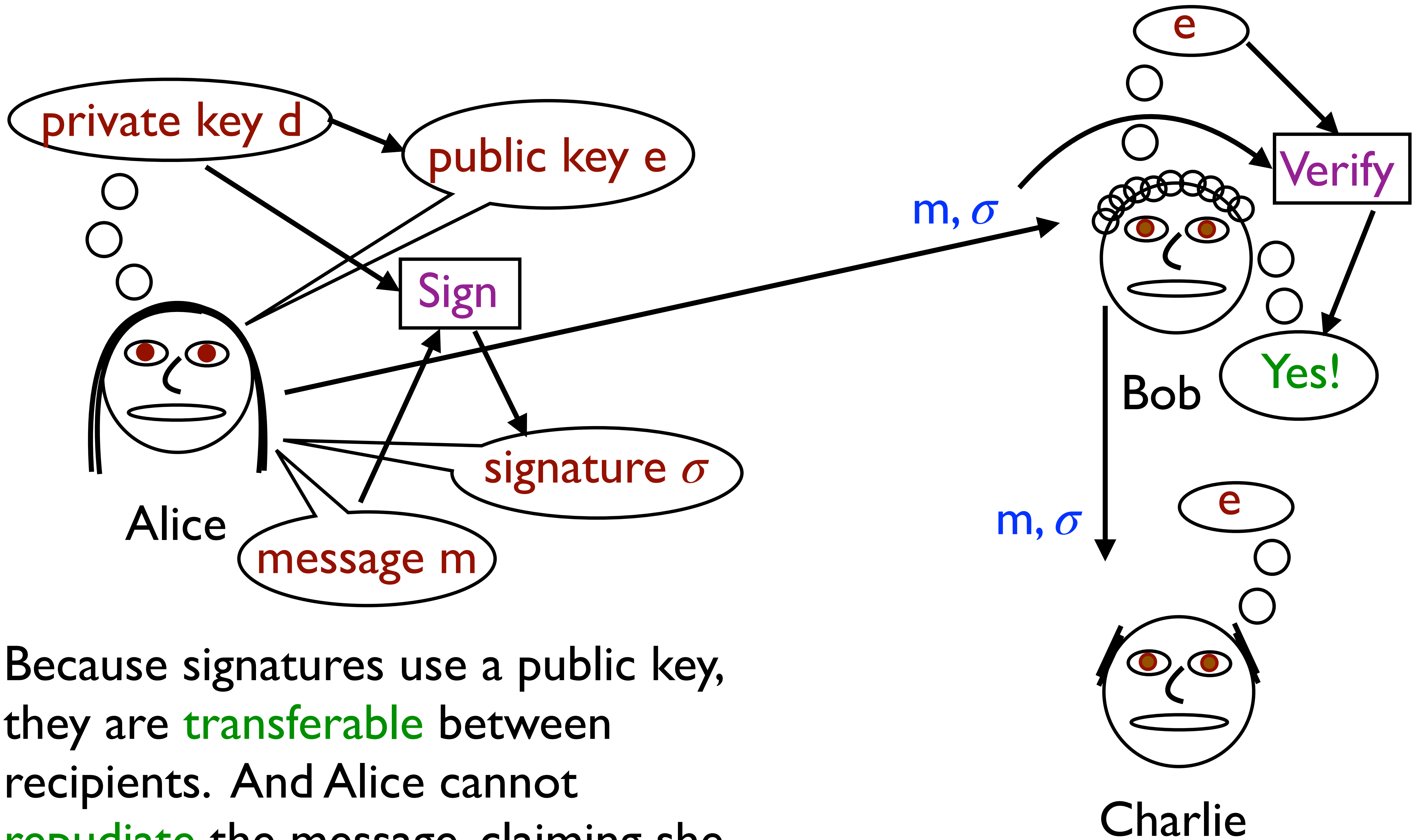
Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

Transferability and Non-Repudiation



Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

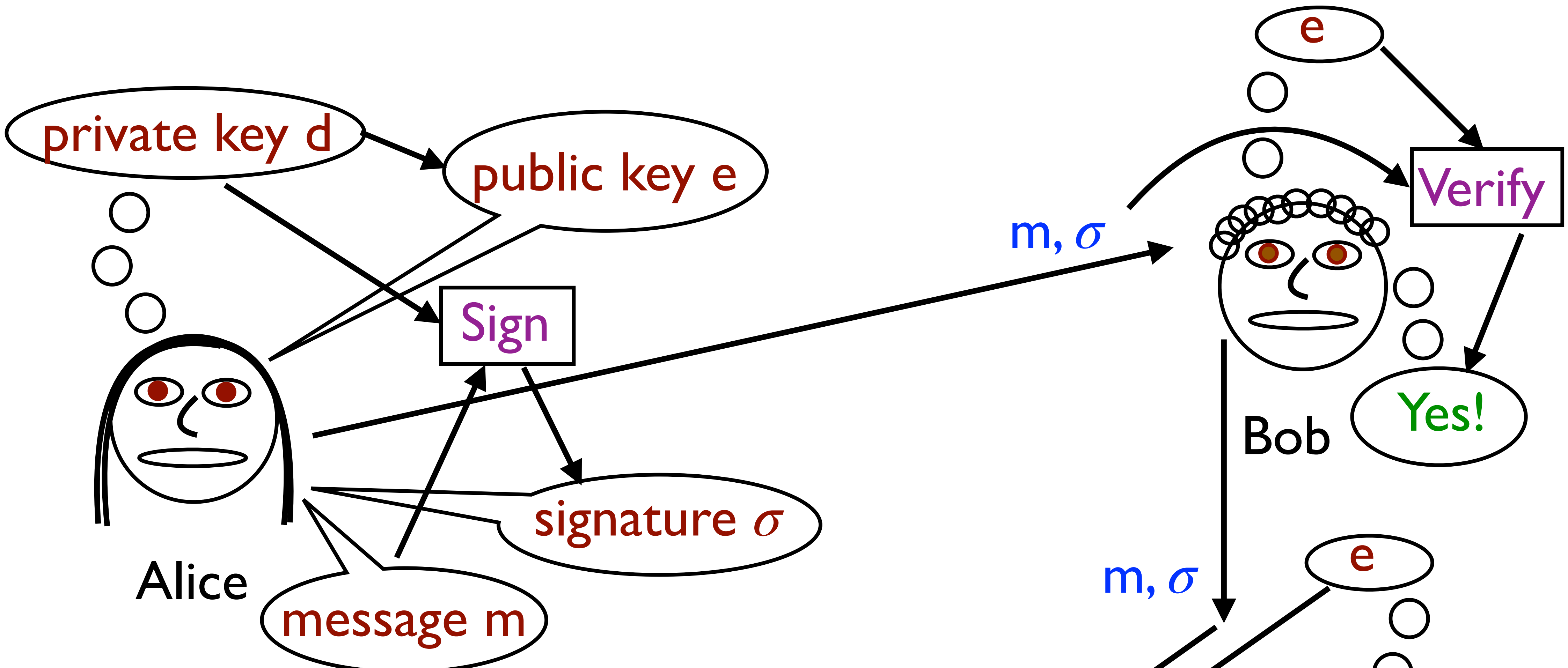
Transferability and Non-Repudiation

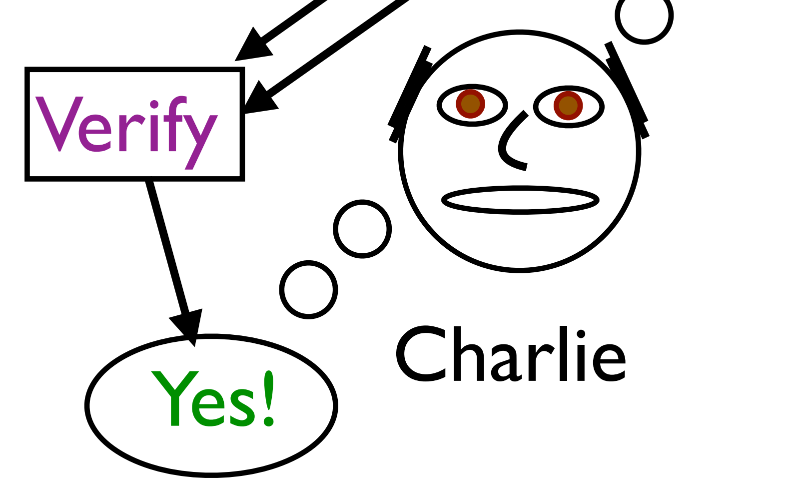


Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

This class is being recorded

Transferability and Non-Repudiation





Because signatures use a public key, they are transferable between recipients. And Alice cannot repudiate the message, claiming she did not send it, since it can only be created using her private key.

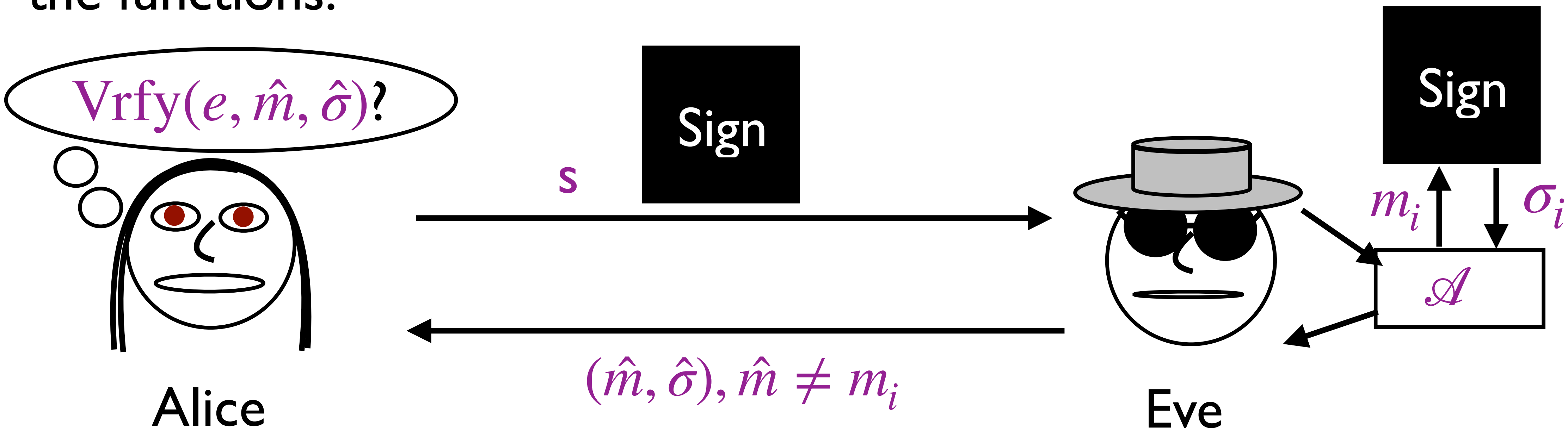
This class is being recorded

Digital Signature Security Definition

Definition: A digital signature (Gen, Sign, Vrfy) with security parameter s is secure (against an adaptive chosen-message attack) if, for any polynomial-time attack $\mathscr{A}$ with the public key e and oracle access to $\mathrm{Sign}(d, m)$, where $\mathscr{A}$ outputs $(\hat{m}, \hat{\sigma})$ such that $\mathscr{A}$ never queried the oracle for $m = \hat{m}$,

$$\Pr(\mathrm{Vrfy}(e, \hat{m}, \hat{\sigma}) = \mathrm{valid}) \leq \epsilon(s)$$

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



RSA Digital Signatures

Gen: Generate two random primes p and q which are s bits long. Let N = pq. Choose $e, d \in \mathbb{Z}_N^*$ such that $ed = 1 \bmod \varphi(N)$. The public key is (N, e) and the private key is (N, d).

Sign: Given message m and private key (N, d). The signature is $\sigma = m^d \bmod N$.

Vrfy: Given (message, signature) pair (m, σ) and public key (N, e). The message is accepted as valid if $m = \sigma^e \bmod N$.

Vote: Is this secure (yes/no/unknown)?

(If RSA assumption is true.)

RSA Digital Signatures

Gen: Generate two random primes p and q which are s bits long. Let N = pq. Choose $e, d \in \mathbb{Z}_N^*$ such that $ed = 1 \bmod \varphi(N)$. The public key is (N, e) and the private key is (N, d).

Sign: Given message m and private key (N, d). The signature is $\sigma = m^d \bmod N$.

Vrfy: Given (message, signature) pair (m, σ) and public key (N, e). The message is accepted as valid if $m = \sigma^e \bmod N$.

Vote: Is this secure (yes/no/unknown)?

(If RSA assumption is true.)

Answer: No.

This class is being recorded

Attack on Plain RSA Signatures

What if we come up with the signature σ *first*?

We then need to find a message m such that $\sigma = m^d \bmod N$.

How can we do this?

This class is being recorded

Attack on Plain RSA Signatures

What if we come up with the signature σ *first*?

We then need to find a message m such that $\sigma = m^d \bmod N$.

How can we do this?

Let $m = \sigma^e \bmod N$!
m decrypts to σ, so σ encrypts to m.

In this case, Eve picks a random σ and therefore gets a random m. This is maybe of limited practical use, but it is definitely a violation of the security definition.

Attack on Plain RSA Signatures

What if we come up with the signature σ *first*?

We then need to find a message m such that $\sigma = m^d \bmod N$.

How can we do this?

Let $m = \sigma^e \bmod N$!
m decrypts to σ, so σ encrypts to m.

In this case, Eve picks a random σ and therefore gets a random m. This is maybe of limited practical use, but it is definitely a violation of the security definition.

And notice that this attack doesn't even require Eve to see any valid signatures.

Forging a Specific Message

Suppose we want to forge a specific message m.

Forging a Specific Message

Suppose we want to forge a specific message m.

Notice: Given signatures for two messages m_1 and m_2,

$$\sigma_1 = m_1^d \bmod N$$
$$\sigma_2 = m_2^d \bmod N$$

Then

$$\sigma_1\sigma_2 = m_1^d m_2^d = (m_1 m_2)^d \bmod N$$

Forging a Specific Message

Suppose we want to forge a specific message m.

Notice: Given signatures for two messages m_1 and m_2,

$$\sigma_1 = m_1^d \bmod N$$
$$\sigma_2 = m_2^d \bmod N$$

Then

$$\sigma_1\sigma_2 = m_1^d m_2^d = (m_1 m_2)^d \bmod N$$

That is, the signature of the message $m_1 m_2$ is $\sigma_1\sigma_2$.

Forging a Specific Message

Suppose we want to forge a specific message m.

Notice: Given signatures for two messages m_1 and m_2,

$$\sigma_1 = m_1^d \bmod N$$
$$\sigma_2 = m_2^d \bmod N$$

Then

$$\sigma_1\sigma_2 = m_1^d m_2^d = (m_1 m_2)^d \bmod N$$

That is, the signature of the message $m_1 m_2$ is $\sigma_1\sigma_2$.

Therefore, to forge the message $m_1 m_2$, all we need are the signatures of the two messages m_1 and m_2.

Revised RSA Signatures

New Idea: Put m through a hash function H first.

Gen: Generate two random primes p and q which are s bits long. Let N = pq. Choose $e, d \in \mathbb{Z}_N^*$ such that $ed = 1 \bmod \varphi(N)$. The public key is (N, e) and the private key is (N, d).

Sign: Given message m and private key (N, d). The signature is $\sigma = H(m)^d \bmod N$.

Vrfy: Given (message, signature) pair (m, σ) and public key (N, e). The message is accepted as valid if $H(m) = \sigma^e \bmod N$.

Vote: Is this secure (yes/no/unknown)?

(If RSA assumption is true.)

Revised RSA Signatures

New Idea: Put m through a hash function H first.

Gen: Generate two random primes p and q which are s bits long. Let N = pq. Choose $e, d \in \mathbb{Z}_N^*$ such that $ed = 1 \bmod \varphi(N)$. The public key is (N, e) and the private key is (N, d).

Sign: Given message m and private key (N, d). The signature is $\sigma = H(m)^d \bmod N$.

Vrfy: Given (message, signature) pair (m, σ) and public key (N, e). The message is accepted as valid if $H(m) = \sigma^e \bmod N$.

Vote: Is this secure (yes/no/unknown)?

(If RSA assumption is true.)

Answer: Yes, if H is a random oracle. The standards pad the output of a standard hash function instead, in which case the answer is unknown.

Security Discussion

With the revised definition, the first attack fails because finding m given σ requires inverting H(m).

But it is hard to invert a random oracle.

Since $H(m_1)$, $H(m_2)$, and $H(m_1 m_2)$ are random, they have no relationship to each other and in particular, $H(m_1 m_2) \neq H(m_1)H(m_2)$.
This foils the second attack.

However, we do need to be careful that we can't find multiplicative relations among the outputs of the $H(m_i)$.

Padding H helps to achieve this in practice, but there is no proof that the particular schemes which are widely used actually work — but no attacks are known despite having been used for many years.

Same Key for Encryption & Signature

What if Bob decides to use the same N and the same (private key, public key) pair (e,d) for encryption and signatures?

Suppose Bob is using plain RSA encryption and signatures. They are already insecure, but this makes an even more powerful attack possible:

Alice sends Bob a ciphertext $c = m^e \bmod N$. Eve intercepts it and convinces Bob to sign the "message" c. Bob then produces the signature $\sigma = c^d = m^{ed} = m \bmod N$! Bob's signature is a decryption of Alice's message, revealing the message to Eve.

When correctly using the hash function with the signature, this particular attack doesn't work since c won't be correctly padded and it will be hard to find a message x to sign that gives H(x) = c, but this still seems like a vulnerability.

There are also good key management reasons not to do this.

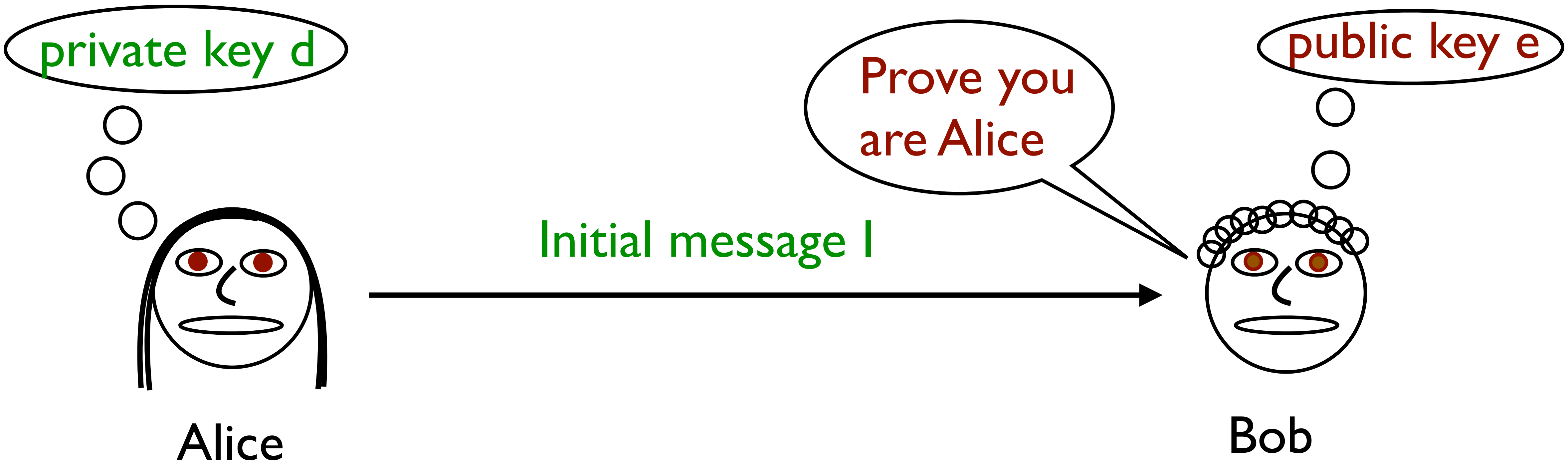
Identification Schemes

Suppose Bob has Alice's public key. Bob is talking to someone who claims to be Alice and wants proof. Alice doesn't want to reveal information that would let Bob impersonate her.

This is a "proof of knowledge."

Identification Schemes

Suppose Bob has Alice's public key. Bob is talking to someone who claims to be Alice and wants proof. Alice doesn't want to reveal information that would let Bob impersonate her.

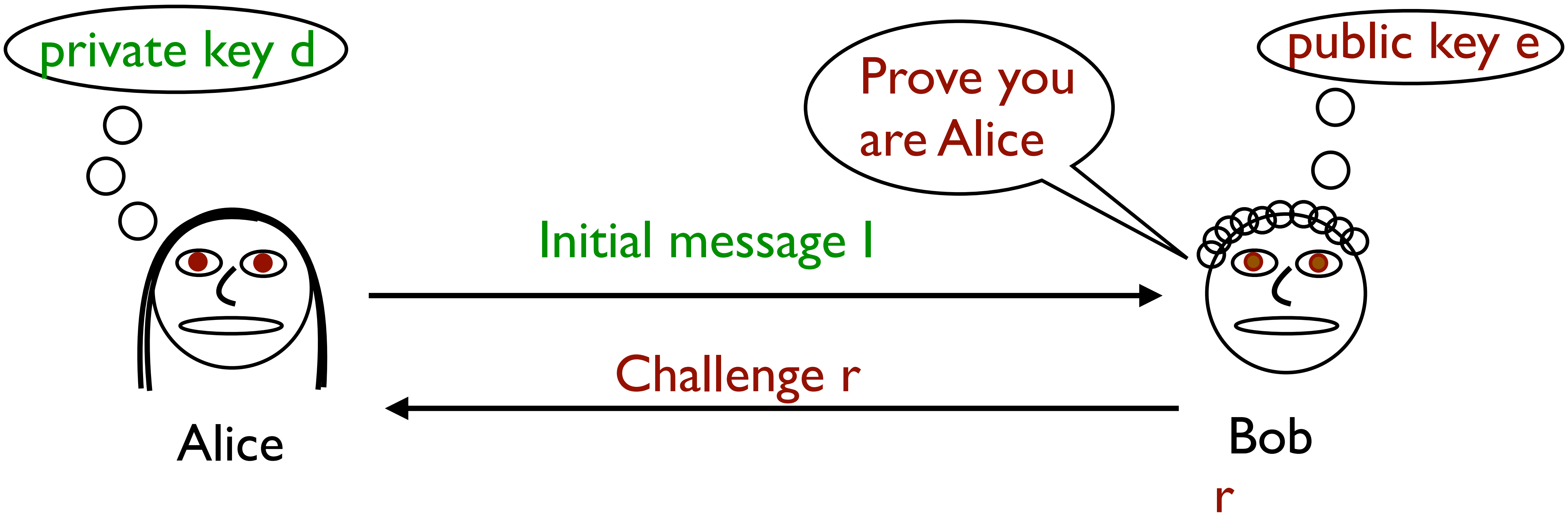


This is a "proof of knowledge."

This class is being recorded

Identification Schemes

Suppose Bob has Alice's public key. Bob is talking to someone who claims to be Alice and wants proof. Alice doesn't want to reveal information that would let Bob impersonate her.

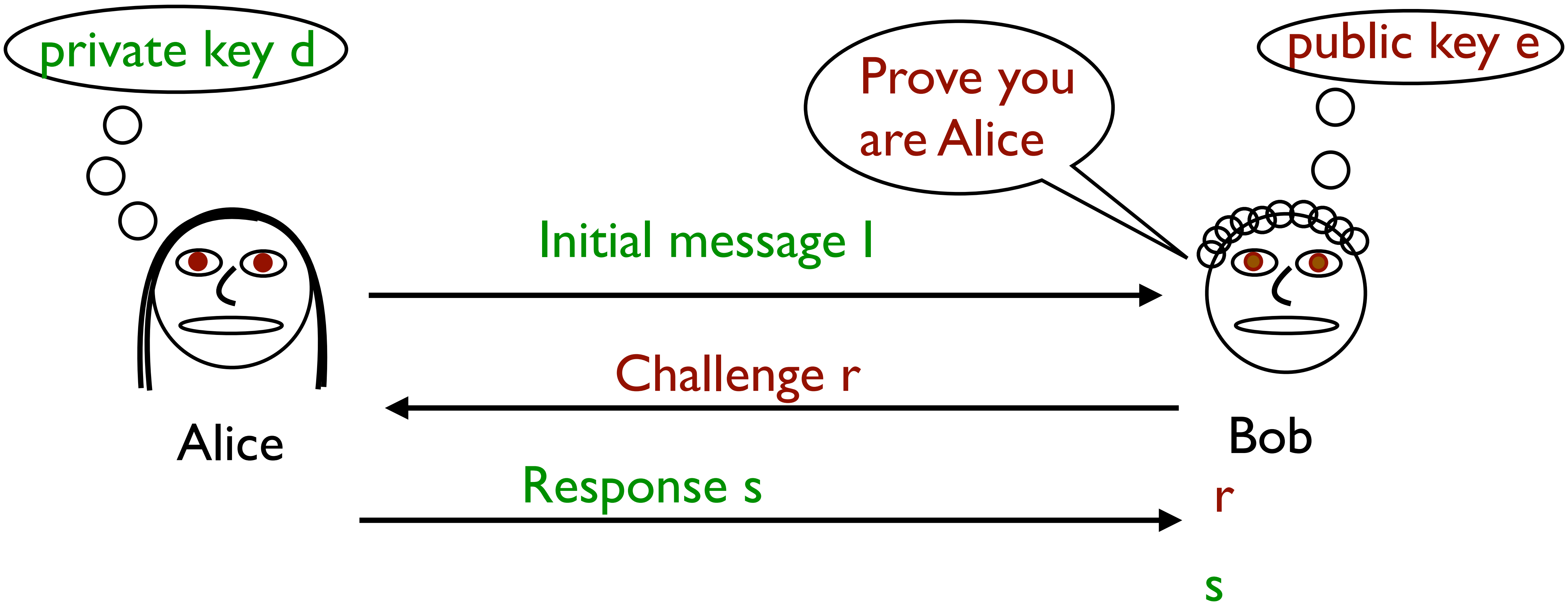


This is a "proof of knowledge."

This class is being recorded

Identification Schemes

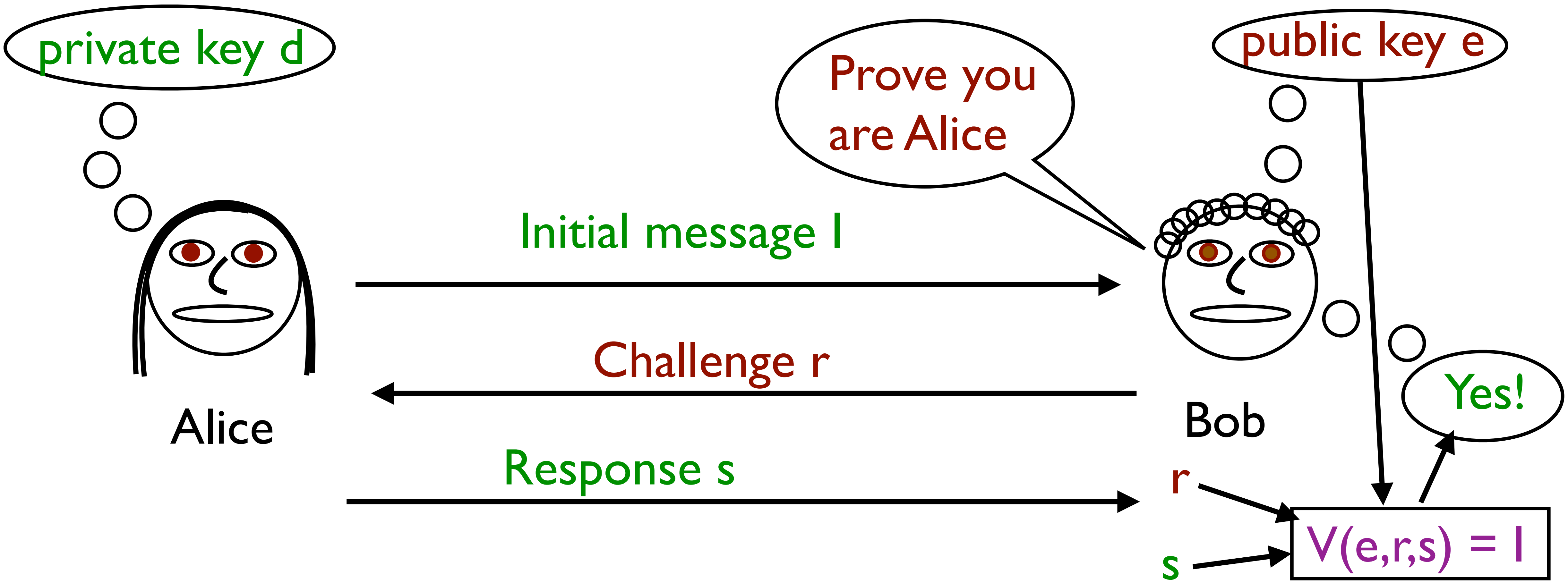
Suppose Bob has Alice's public key. Bob is talking to someone who claims to be Alice and wants proof. Alice doesn't want to reveal information that would let Bob impersonate her.



This is a "proof of knowledge."

Identification Schemes

Suppose Bob has Alice's public key. Bob is talking to someone who claims to be Alice and wants proof. Alice doesn't want to reveal information that would let Bob impersonate her.



This is a "proof of knowledge."

Discrete-Log Identification

Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.

This class is being recorded

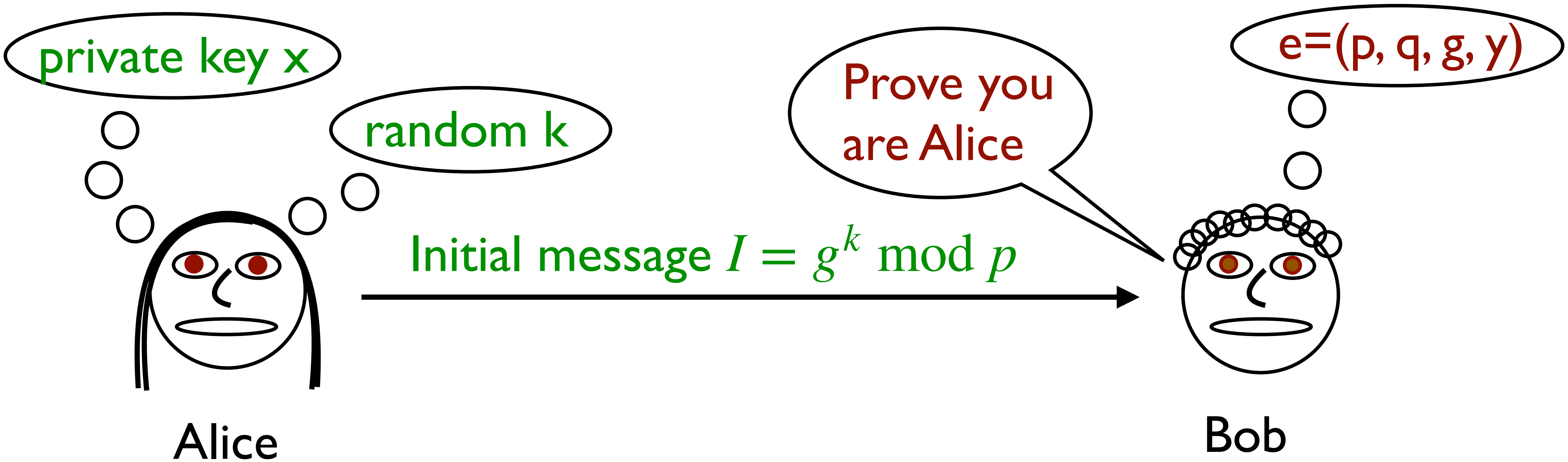
Discrete-Log Identification

Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.

This class is being recorded

Discrete-Log Identification

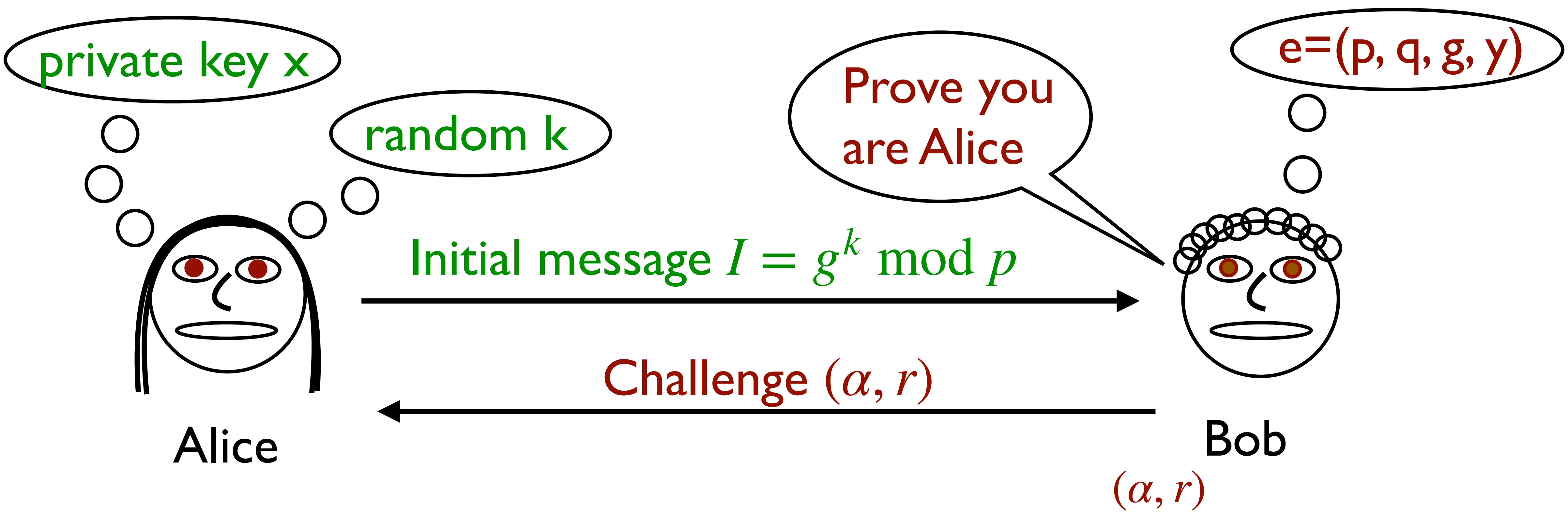
Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.



This class is being recorded

Discrete-Log Identification

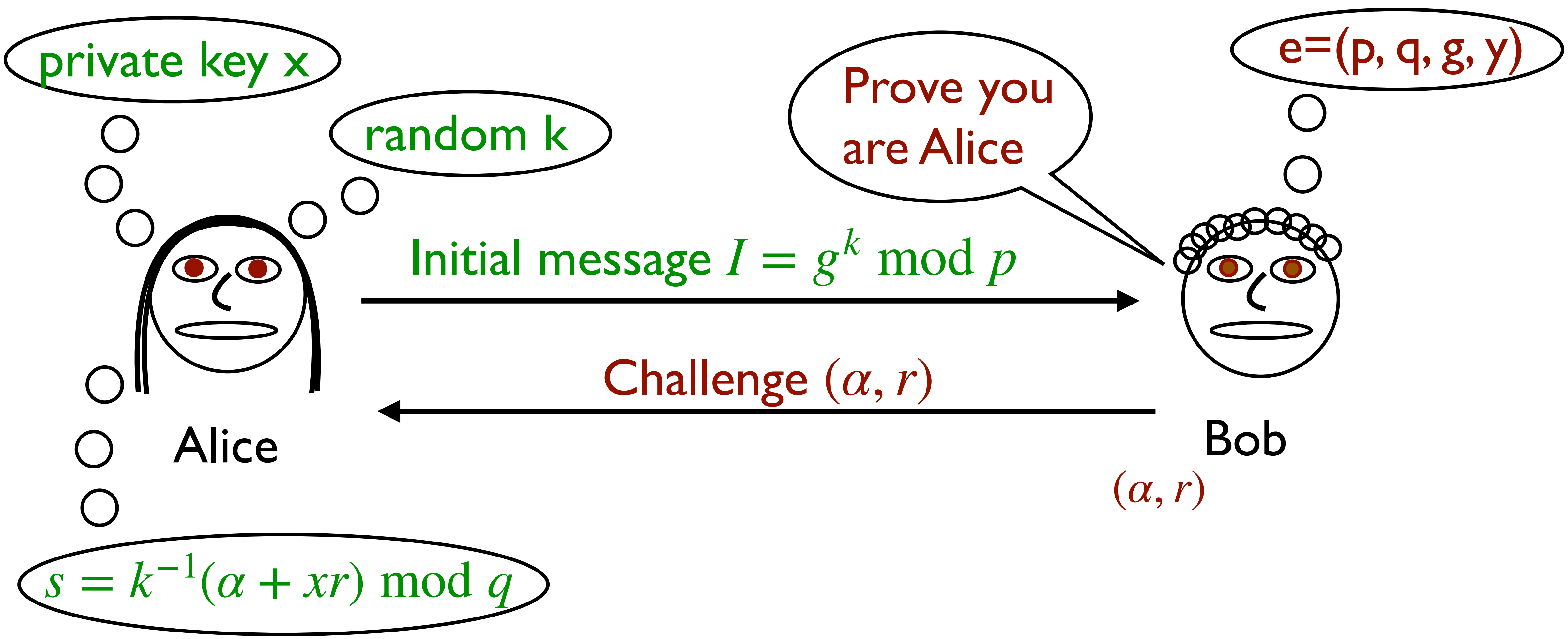
Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.



This class is being recorded

Discrete-Log Identification

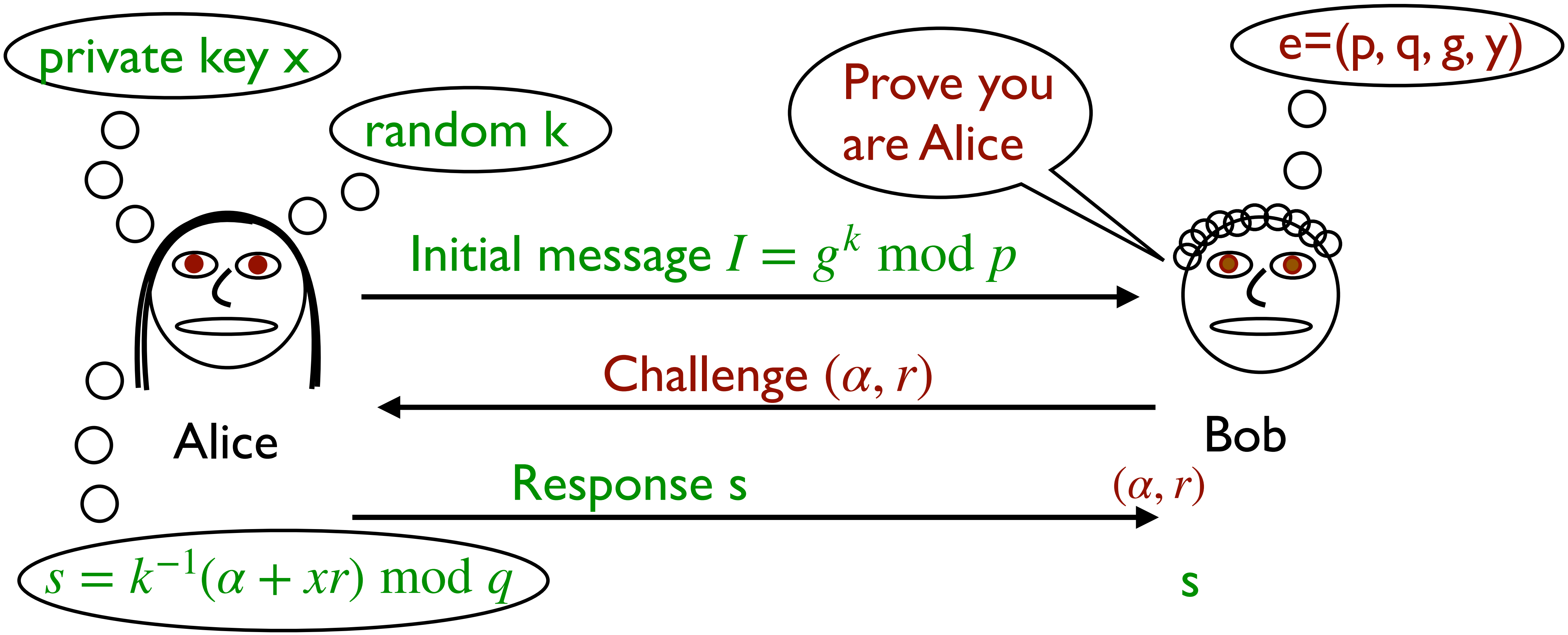
Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.



This class is being recorded

Discrete-Log Identification

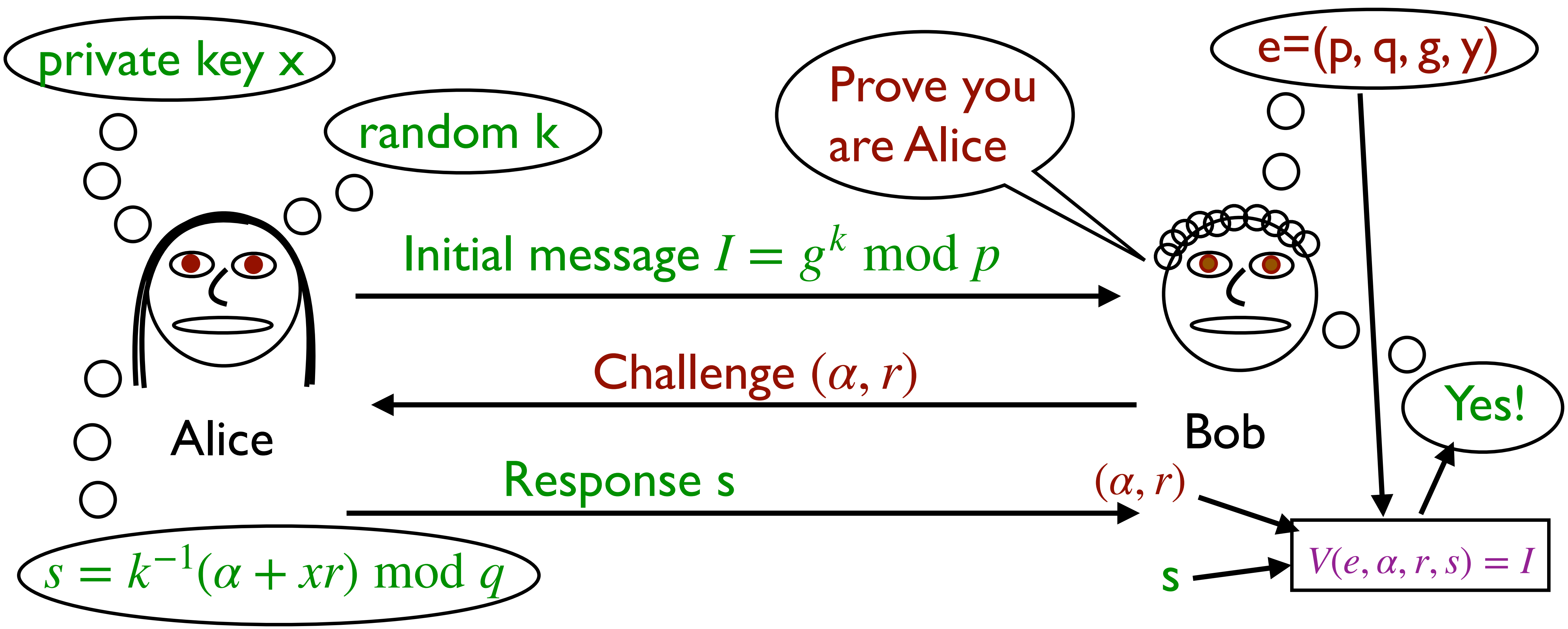
Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.



This class is being recorded

Discrete-Log Identification

Suppose Alice uses a El Gamal-like public key: Prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup), and y, with $y = g^x \bmod p$. Here x is the private key.



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

This class is being recorded

Discrete-Log Identification

Setup:

This class is being recorded

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Identification Protocol:

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Identification Protocol:

1. Alice chooses random $k \in \mathbb{Z}_q^*$, computes $I = g^k \bmod p$, and sends I to Bob. Alice keeps k and does not send it to Bob.

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Identification Protocol:

1. Alice chooses random $k \in \mathbb{Z}_q^*$, computes $I = g^k \bmod p$, and sends I to Bob. Alice keeps k and does not send it to Bob.
2. Bob chooses random α and r in $\mathbb{Z}_q$ and sends (α, r) to Alice.

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Identification Protocol:

1. Alice chooses random $k \in \mathbb{Z}_q^*$, computes $I = g^k \bmod p$, and sends I to Bob. Alice keeps k and does not send it to Bob.
2. Bob chooses random α and r in $\mathbb{Z}_q$ and sends (α, r) to Alice.
3. Alice computes $s = k^{-1}(\alpha + xr) \bmod q$ using her two secrets k and x (along with public (α, r, q)) and sends s to Bob.

Discrete-Log Identification

Setup:

1. Alice chooses (p, q, g) for prime p with large prime order q subgroup of $\mathbb{Z}_p^*$, base g (in the order q subgroup).
2. Alice chooses random x in $\mathbb{Z}_q$ and computes $y = g^x \bmod p$.
3. Alice distributes (p, q, g, y) as her public key and keeps x as her private key.

Identification Protocol:

1. Alice chooses random $k \in \mathbb{Z}_q^*$, computes $I = g^k \bmod p$, and sends I to Bob. Alice keeps k and does not send it to Bob.
2. Bob chooses random α and r in $\mathbb{Z}_q$ and sends (α, r) to Alice.
3. Alice computes $s = k^{-1}(\alpha + xr) \bmod q$ using her two secrets k and x (along with public (α, r, q)) and sends s to Bob.
4. Bob computes $V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{rs^{-1}} \bmod p$ and accepts if $V(e, \alpha, r, s) = I$ and $s \neq 0$.

This class is being recorded

Correctness of Identification

When Alice is actually present she can pass this test:

Correctness of Identification

When Alice is actually present she can pass this test:

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

Correctness of Identification

When Alice is actually present she can pass this test:

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

Correctness of Identification

When Alice is actually present she can pass this test:

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

⟵ $y = g^{x} \bmod p$

Correctness of Identification

When Alice is actually present she can pass this test:

$$\begin{aligned} V(e, \alpha, r, s) &= g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p \\ &= (g^{\alpha} y^{r})^{s^{-1}} \bmod p \\ &= (g^{\alpha + r x})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p \end{aligned}$$

This class is being recorded

Correctness of Identification

When Alice is actually present she can pass this test:

$$
\begin{aligned}
V(e, \alpha, r, s) &= g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p \\
&= (g^{\alpha} y^{r})^{s^{-1}} \bmod p \\
&= (g^{\alpha + rx})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p \\
&= g^{(\alpha + rx) s^{-1}} \bmod p
\end{aligned}
$$

Correctness of Identification

When Alice is actually present she can pass this test:

$$V(e,\alpha,r,s) = g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p$$
$$= (g^{\alpha} y^{r})^{s^{-1}} \bmod p$$
$$= (g^{\alpha + rx})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p$$
$$= g^{(\alpha + rx) s^{-1}} \bmod p$$

$$s = k^{-1}(\alpha + xr) \bmod q$$

Correctness of Identification

When Alice is actually present she can pass this test:

$$
\begin{aligned}
V(e, \alpha, r, s) &= g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p \\
&= (g^{\alpha} y^{r})^{s^{-1}} \bmod p \\
&= (g^{\alpha + rx})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p \\
&= g^{(\alpha + rx)s^{-1}} \bmod p \\
&= g^{k} \bmod p
\end{aligned}
$$

$$s = k^{-1}(\alpha + xr) \bmod q$$

This class is being recorded

Correctness of Identification

When Alice is actually present she can pass this test:

$$\begin{aligned} V(e, \alpha, r, s) &= g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p \\ &= (g^{\alpha} y^{r})^{s^{-1}} \bmod p \\ &= (g^{\alpha + rx})^{s^{-1}} \bmod p \\ &= g^{(\alpha + rx) s^{-1}} \bmod p \\ &= g^{k} \bmod p \\ &= I \end{aligned}$$

$y = g^{x} \bmod p$

$s = k^{-1}(\alpha + xr) \bmod q$

This class is being recorded

Correctness of Identification

When Alice is actually present she can pass this test:

$$
\begin{aligned}
V(e, \alpha, r, s) &= g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p \\
&= (g^{\alpha} y^{r})^{s^{-1}} \bmod p \\
&= (g^{\alpha + rx})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p \\
&= g^{(\alpha + rx) s^{-1}} \bmod p \\
&= g^{k} \bmod p \\
&= I
\end{aligned}
$$

$$s = k^{-1}(\alpha + xr) \bmod q$$

This is the condition which causes Bob to accept.

Correctness of Identification

When Alice is actually present she can pass this test:

$$V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p$$
$$= (g^{\alpha} y^{r})^{s^{-1}} \bmod p$$
$$= (g^{\alpha + rx})^{s^{-1}} \bmod p \quad \longleftarrow \quad y = g^{x} \bmod p$$
$$= g^{(\alpha + rx) s^{-1}} \bmod p$$
$$= g^{k} \bmod p$$
$$= I$$

$$s = k^{-1}(\alpha + xr) \bmod q$$

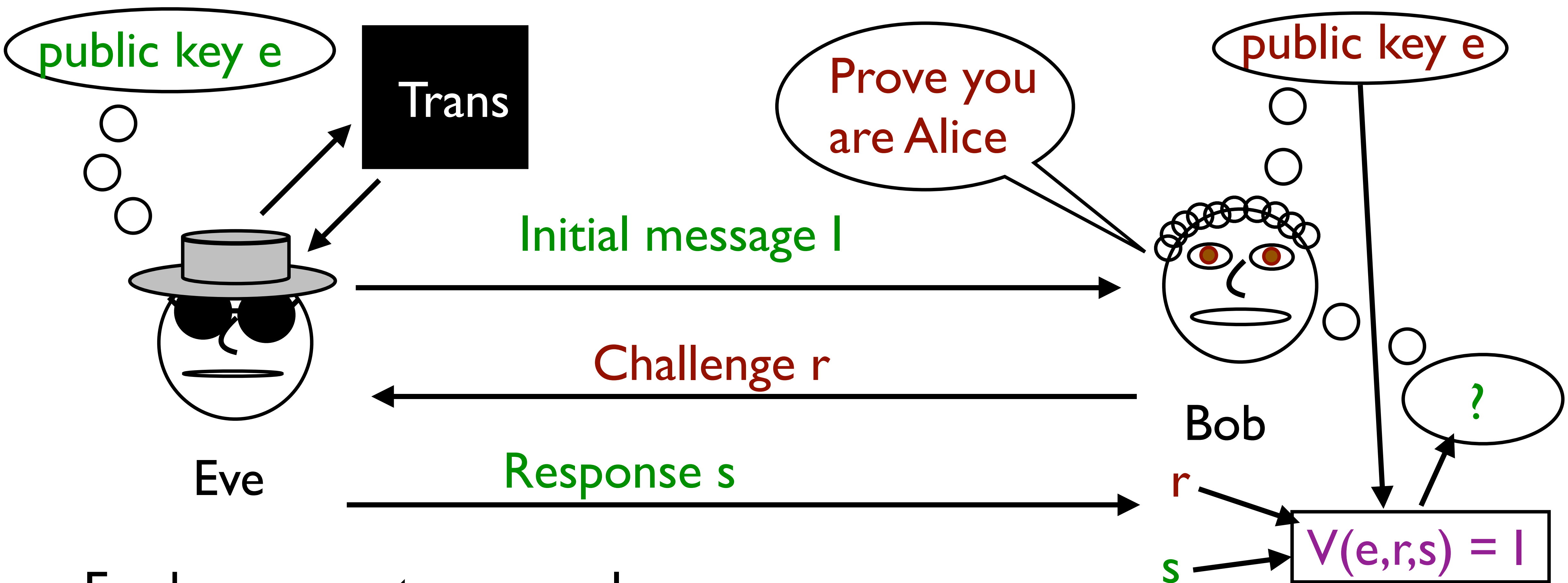
This is the condition which causes Bob to accept.

Bob checks for 1 more thing: that $s \neq 0$. But this happens only if $\alpha + xr = 0 \bmod q$, which happens with probability only 1/q.

Security of Identification

An identification protocol is secure if Eve can't masquerade as Alice even after seeing transcripts of previous identification sessions (all public information of the session).

Identification security game:



Eve has access to an oracle to generate transcripts.

Security of Identification

Why is the identification protocol we saw before secure?

Security of Identification

Why is the identification protocol we saw before secure?

Claim: It is possible to generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

Security of Identification

Why is the identification protocol we saw before secure?

Claim: It is possible to generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

This means that anything Eve can do with access to the transcript oracle, she can also do without access to the oracle.

Security of Identification

Why is the identification protocol we saw before secure?

> Claim: It is possible to generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

This means that anything Eve can do with access to the transcript oracle, she can also do without access to the oracle.

Suppose Eve can break the identification protocol. Then she can find the correct s to match a random challenge pair (α, r). She can do this in response to Bob, but she can also generate (α, r) herself and find s given I. (Note: generating I doesn't use x.)

Security of Identification

Why is the identification protocol we saw before secure?

> Claim: It is possible to generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

This means that anything Eve can do with access to the transcript oracle, she can also do without access to the oracle.

Suppose Eve can break the identification protocol. Then she can find the correct s to match a random challenge pair (α, r). She can do this in response to Bob, but she can also generate (α, r) herself and find s given I. (Note: generating I doesn't use x.)

She can repeat this for the same I and α but different s and r.

Security of Identification

Why is the identification protocol we saw before secure?

> Claim: It is possible to generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

This means that anything Eve can do with access to the transcript oracle, she can also do without access to the oracle.

Suppose Eve can break the identification protocol. Then she can find the correct s to match a random challenge pair (α, r). She can do this in response to Bob, but she can also generate (α, r) herself and find s given I. (Note: generating I doesn't use x.)

She can repeat this for the same I and α but different s and r.

> Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

This class is being recorded

Fake Transcript

Claim: Eve can generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

How?

Fake Transcript

Claim: Eve can generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

How?

Generate the transcript out of order:

Fake Transcript

Claim: Eve can generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

How?

Generate the transcript out of order:

1. Choose random α and **r** in $\mathbb{Z}_q$ and random **s** in $\mathbb{Z}_q^*$.
2. Let $I = g^{\alpha s^{-1}} y^{r s^{-1}}$.

Fake Transcript

Claim: Eve can generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

How?

Generate the transcript out of order:

1. Choose random α and r in $\mathbb{Z}_q$ and random s in $\mathbb{Z}_q^*$.
2. Let $I = g^{\alpha s^{-1}} y^{r s^{-1}}$.

This choice of I corresponds to a uniform choice of k with $I = g^k \bmod p$ (but the distribution of s is not quite the same since it is guaranteed that $s \neq 0$).

Fake Transcript

Claim: Eve can generate fake transcripts which don't use any knowledge of x but look the same as real transcripts.

How?

Generate the transcript out of order:

1. Choose random α and r in $\mathbb{Z}_q$ and random s in $\mathbb{Z}_q^*$.
2. Let $I = g^{\alpha s^{-1}} y^{r s^{-1}}$.

This choice of I corresponds to a uniform choice of k with $I = g^k \bmod p$ (but the distribution of s is not quite the same since it is guaranteed that $s \neq 0$).

I will pass Bob's test:

$V(e, \alpha, r, s) = g^{\alpha s^{-1}} y^{r s^{-1}} \bmod p$ automatically equals I.

Reduction from Discrete Log

This class is being recorded

Reduction from Discrete Log

Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

This class is being recorded

Reduction from Discrete Log

Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

Suppose the responses are s_1 and s_2.

Reduction from Discrete Log

Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

Suppose the responses are s_1 and s_2.

Then

$$g^{\alpha s_1^{-1}} y^{r_1 s_1^{-1}} = I = g^{\alpha s_2^{-1}} y^{r_2 s_2^{-1}} \bmod p$$

Reduction from Discrete Log

Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

Suppose the responses are s_1 and s_2.

Then

$$g^{\alpha s_1^{-1}} y^{r_1 s_1^{-1}} = I = g^{\alpha s_2^{-1}} y^{r_2 s_2^{-1}} \bmod p$$

Thus,

$$y^{r_2 s_2^{-1} - r_1 s_1^{-1}} = g^{\alpha (s_1^{-1} - s_2^{-1})} \bmod p$$

Reduction from Discrete Log

Claim: Eve can solve discrete log if she can give correct responses to two pairs (α, r_1) and (α, r_2) for the same I.

Suppose the responses are s_1 and s_2.

Then

$$g^{\alpha s_1^{-1}} y^{r_1 s_1^{-1}} = I = g^{\alpha s_2^{-1}} y^{r_2 s_2^{-1}} \bmod p$$

Thus,

$$y^{r_2 s_2^{-1} - r_1 s_1^{-1}} = g^{\alpha (s_1^{-1} - s_2^{-1})} \bmod p$$

Since g and y have order q, if we let

$$x = \alpha (s_1^{-1} - s_2^{-1})(r_2 s_2^{-1} - r_1 s_1^{-1})^{-1} \bmod q$$

then

$$y = g^x \bmod p$$

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

Gen: The primes p and q and base g have been pre-determined, along with hash function H. Gen chooses a random $x \in \mathbb{Z}_q$, which becomes the private key. I.e., d = x. The public key is $y = g^x \bmod p$.

This class is being recorded

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

Gen: The primes p and q and base g have been pre-determined, along with hash function H. Gen chooses a random $x \in \mathbb{Z}_q$, which becomes the private key. I.e., d = x. The public key is $y = g^x \bmod p$.

Sign: Given message m and private key x. Choose random $k \in \mathbb{Z}_q^*$ and let $r = g^k \bmod p$. The signature is $\sigma = (r, s)$, where $s = k^{-1}(H(m) + xr) \bmod q$.

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

Gen: The primes p and q and base g have been pre-determined, along with hash function H. Gen chooses a random $x \in \mathbb{Z}_q$, which becomes the private key. I.e., d = x. The public key is $y = g^x \bmod p$.

Sign: Given message m and private key x. Choose random $k \in \mathbb{Z}_q^*$ and let $r = g^k \bmod p$. The signature is $\sigma = (r, s)$, where $s = k^{-1}(H(m) + xr) \bmod q$.

Vrfy: Given m, σ, and y. Vrfy checks if $r = g^{H(m)s^{-1}} y^{rs^{-1}} \bmod p$.

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

Gen: The primes p and q and base g have been pre-determined, along with hash function H. Gen chooses a random $x \in \mathbb{Z}_q$, which becomes the private key. I.e., d = x. The public key is $y = g^x \bmod p$.

Sign: Given message m and private key x. Choose random $k \in \mathbb{Z}_q^*$ and let $r = g^k \bmod p$. The signature is $\sigma = (r, s)$, where $s = k^{-1}(H(m) + xr) \bmod q$.

Vrfy: Given m, σ, and y. Vrfy checks if $r = g^{H(m)s^{-1}} y^{rs^{-1}} \bmod p$.

Correctness: Same as identification scheme.

DSA (Discrete-Log Signatures)

DSA is like a transcript of the identification protocol where $\alpha = H(m)$ and $r = I$.

Gen: The primes p and q and base g have been pre-determined, along with hash function H. Gen chooses a random $x \in \mathbb{Z}_q$, which becomes the private key. I.e., d = x. The public key is $y = g^x \bmod p$.

Sign: Given message m and private key x. Choose random $k \in \mathbb{Z}_q^*$ and let $r = g^k \bmod p$. The signature is $\sigma = (r, s)$, where $s = k^{-1}(H(m) + xr) \bmod q$.

Vrfy: Given m, σ, and y. Vrfy checks if $r = g^{H(m)s^{-1}} y^{rs^{-1}} \bmod p$.

Correctness: Same as identification scheme.

Security: Unproven, even when H is a random oracle.

This class is being recorded

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

This class is being recorded

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

$$k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

$$k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

Now k is known. Then:

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

$$k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

Now k is known. Then:

$$s = k^{-1}(H(m) + xr) \bmod q$$

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

$$k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

Now k is known. Then:

$$s = k^{-1}(H(m) + xr) \bmod q$$

← Everything but r known

Repeated k

k must be new each time and kept hidden from Eve.

If k is repeated:

Then r repeats and (m_1, s_1) and (m_2, s_2) are known with:

$$s_1 = k^{-1}(H(m_1) + xr) \bmod q$$

$$s_2 = k^{-1}(H(m_2) + xr) \bmod q$$

$$s_1 - s_2 = k^{-1}(H(m_1) - H(m_2)) \bmod q$$

$$k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \bmod q$$

Now k is known. Then:

$$s = k^{-1}(H(m) + xr) \bmod q$$

← Everything but r known

$$x = r^{-1}(ks - H(m)) \bmod q$$

and Eve learns the private key x.