Cryptography

Lecture 13: Digital Signatures

January 21, 2025

Overview

Contents

1. Classical cryptography
   (Shift & Vigenère cipher, one-time pad, perfect secrecy)
2. Security definitions & threat models
   (Computational security, CPA & CCA)
3. Private-key cryptography
   (Message authentication, hash functions, primitives, relevant ciphers)
4. Public-key cryptography
   (Assumptions, key management, digital signatures, relevant ciphers)

Key Exchange

Motivation

- Alice & Bob want to establish shared private key
  (to utilize private-key encryption, message authentication, etc.)
- Secure channel not available
- Eve will eavesdrop on whole communication

Key Exchange

Key exchange protocol $\mathcal{E}$

$k_0 \leftarrow \mathcal{E}(1^n)$

Alice

Alice & Bob run key exchange protocol $\mathcal{E}(1^n)$

Listen

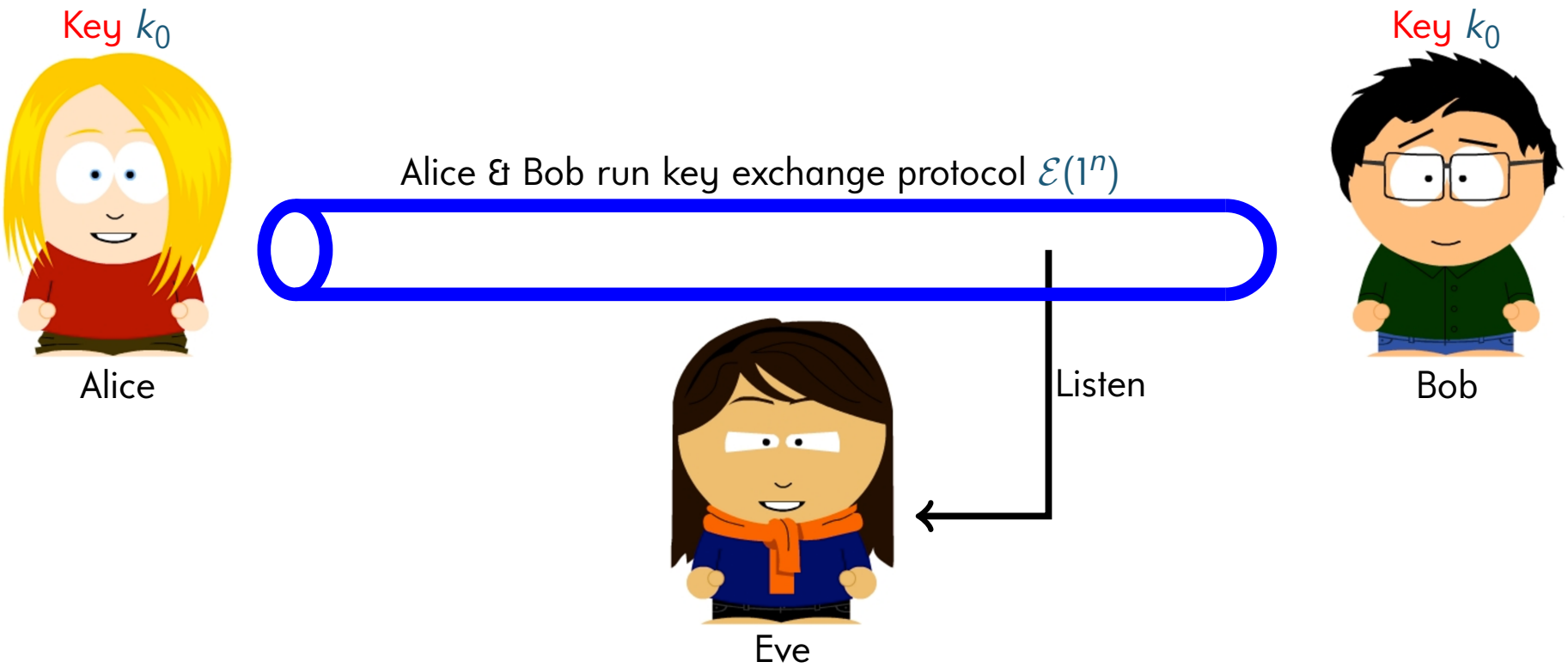
Eve

$k_0 \leftarrow \mathcal{E}(1^n)$

Bob

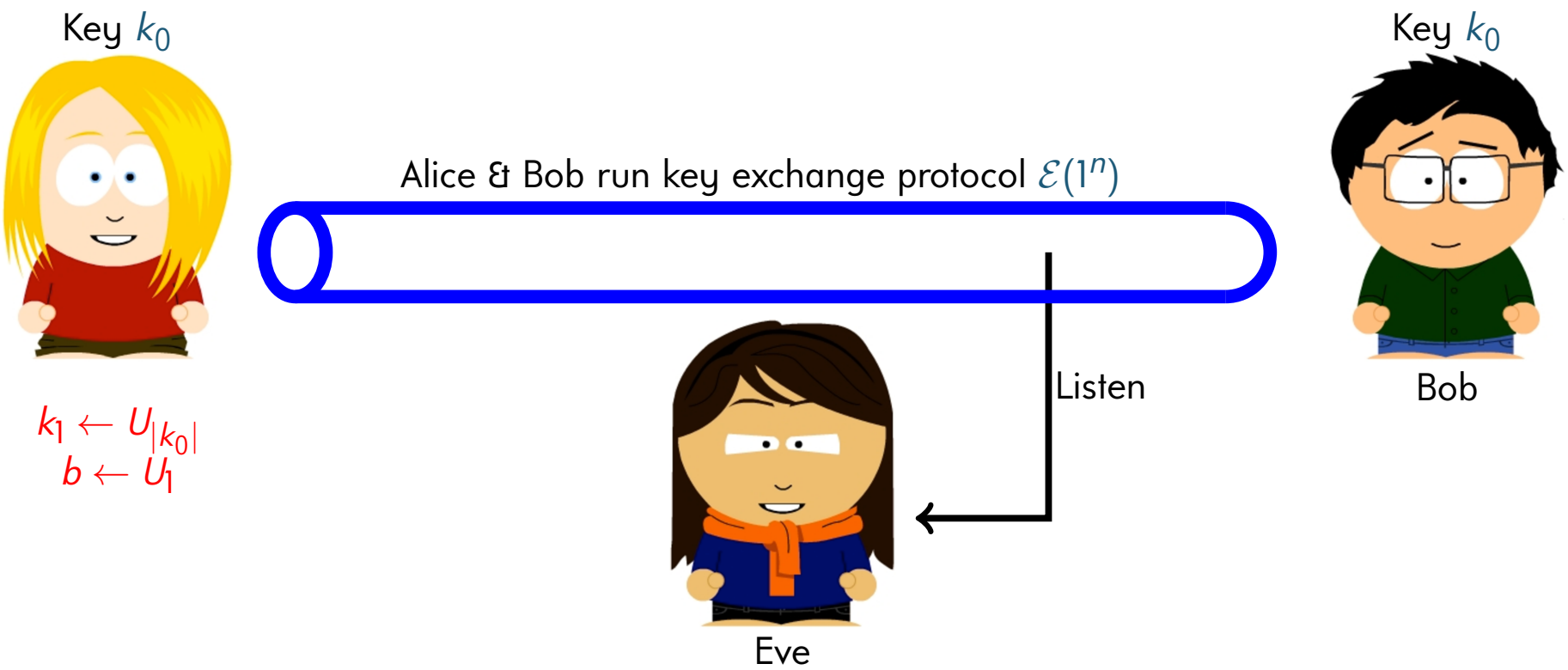
Key Exchange

Key exchange protocol $\mathcal{E}$



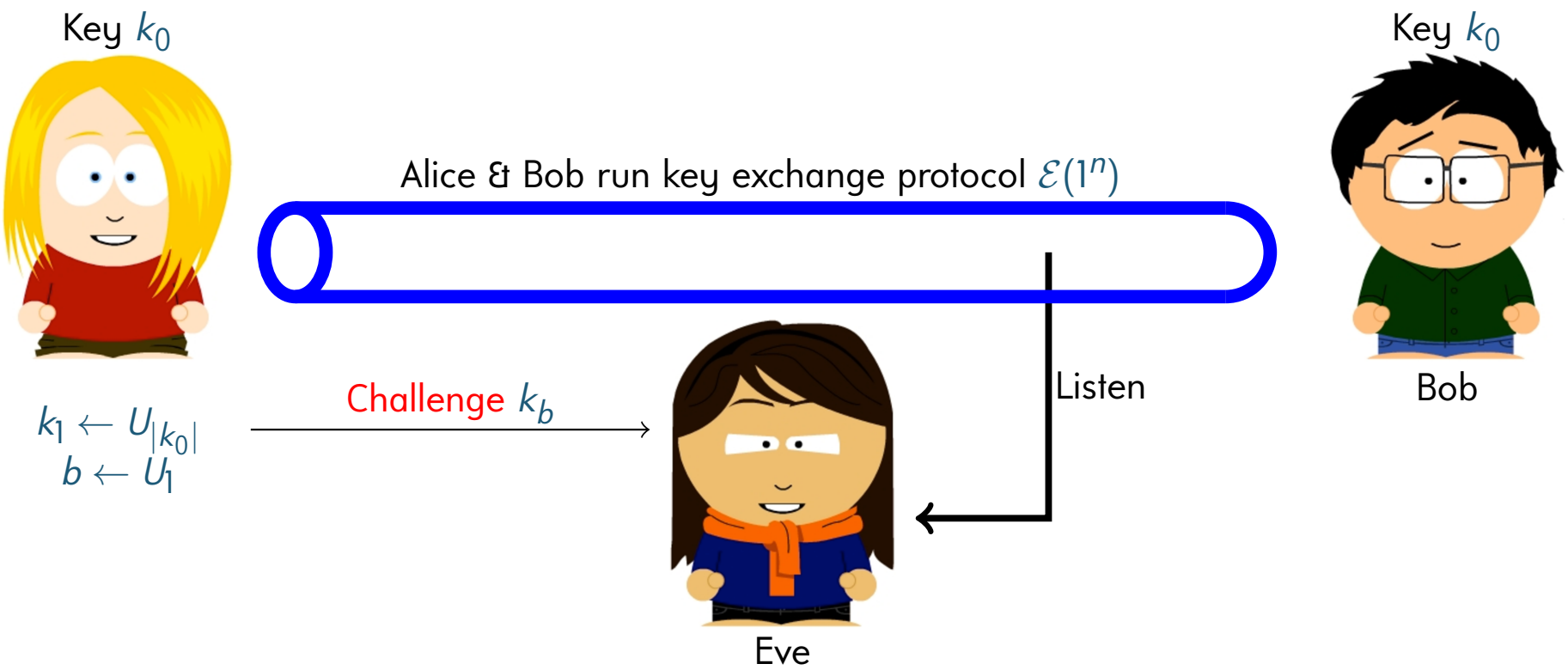
Key Exchange

Key exchange protocol $\mathcal{E}$



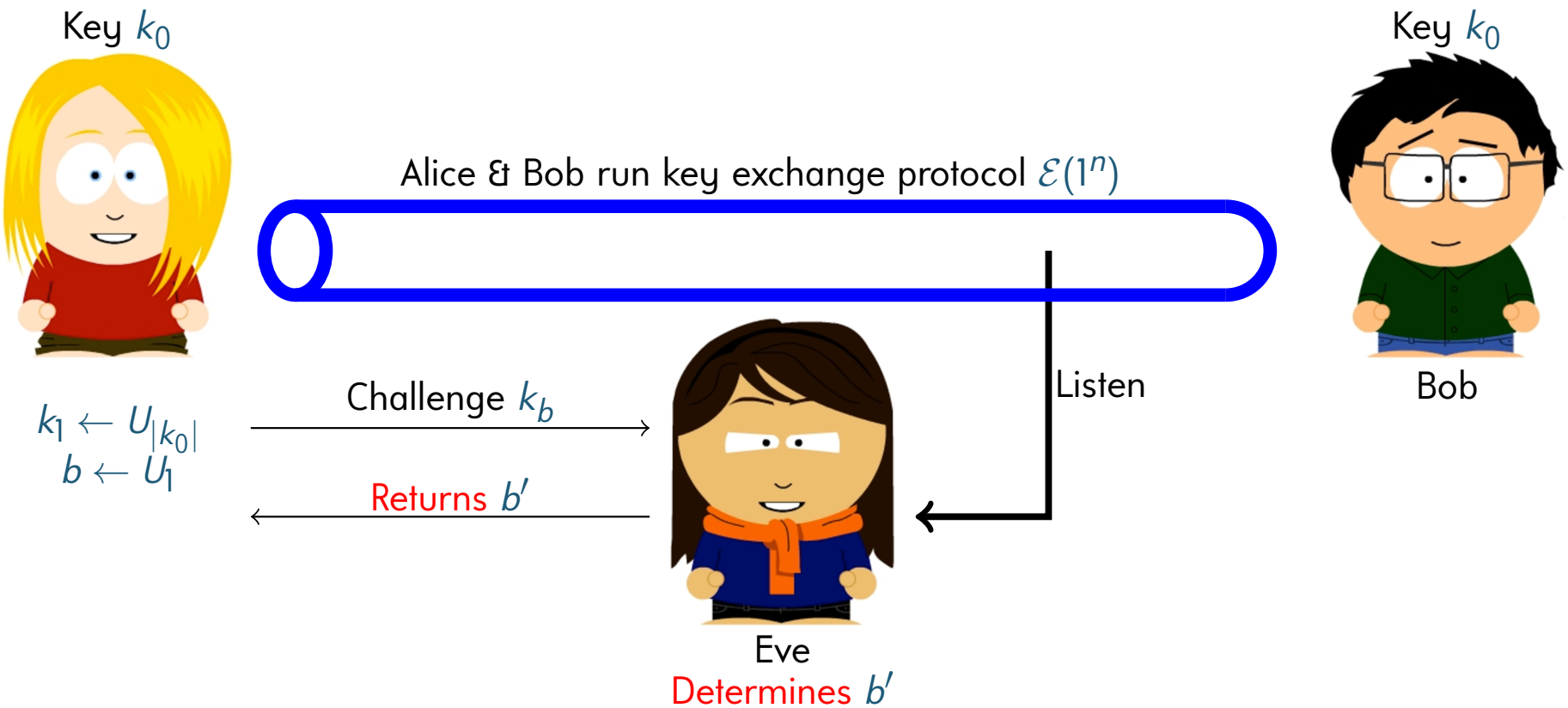
Key Exchange

Key exchange protocol $\mathcal{E}$



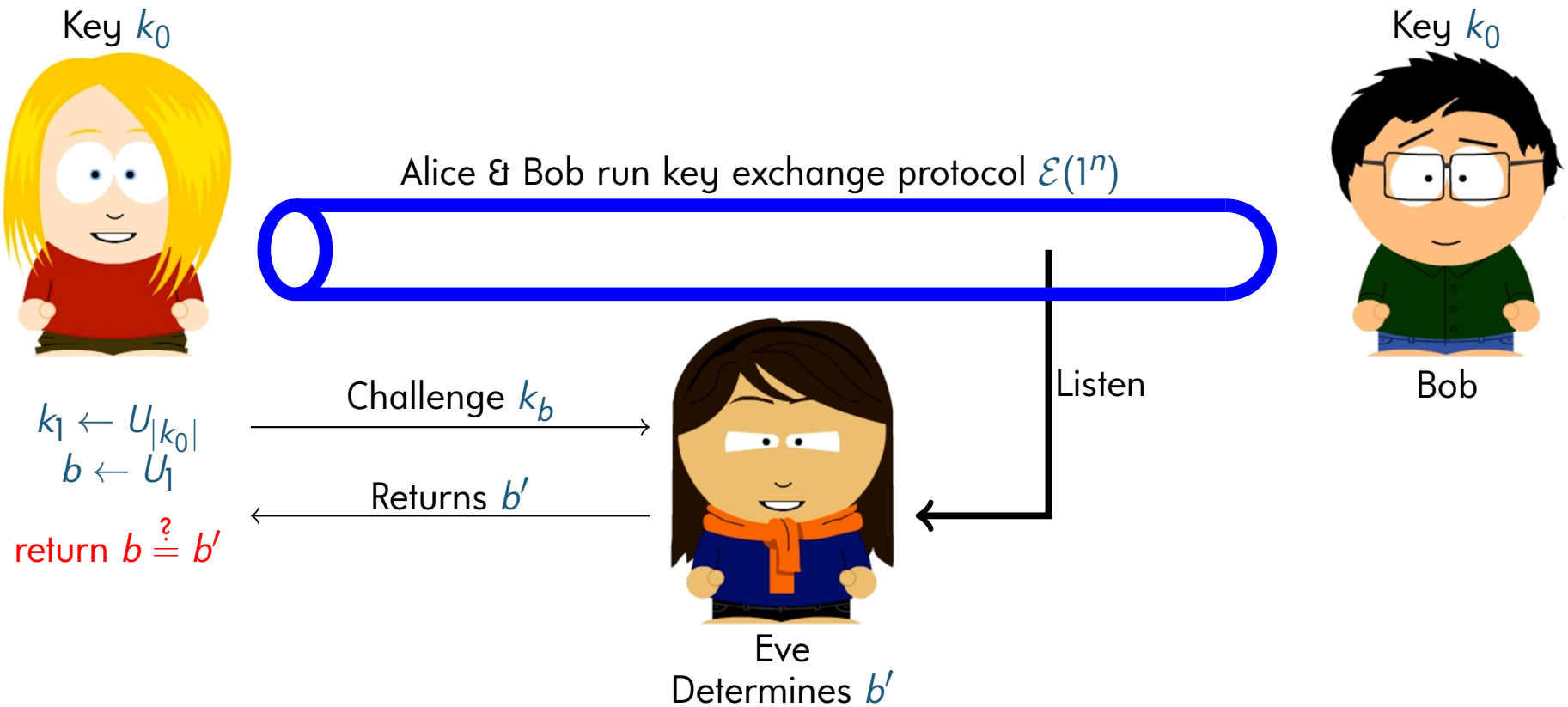
Key Exchange

Key exchange protocol $\mathcal{E}$



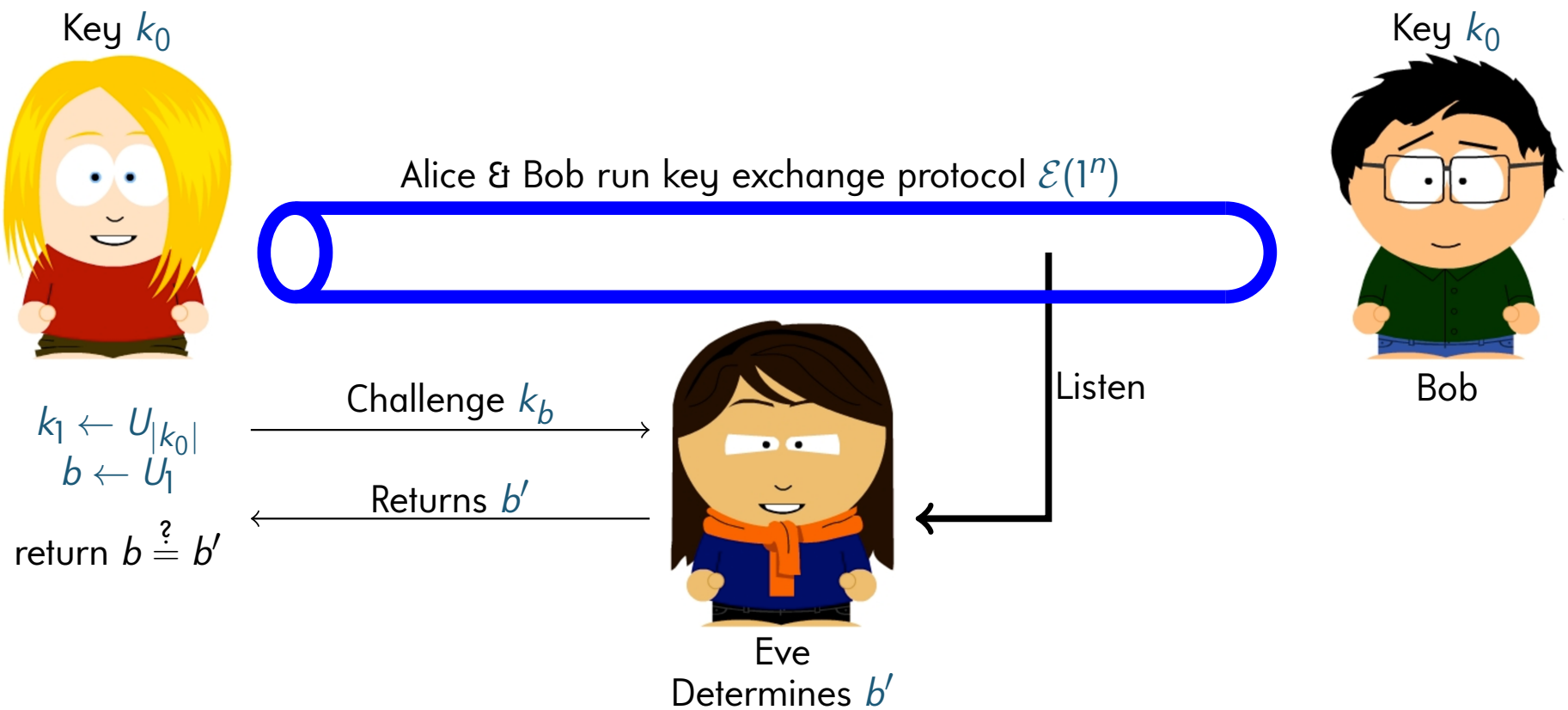
Key Exchange

Key exchange protocol $\mathcal{E}$



Key Exchange

Key exchange protocol $\mathcal{E}$



Key exchange protocol EAV-secure
if for all PPT implementations of Eve success chance is almost equal to $\frac{1}{2}$

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Bob

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Security parameter 1^n and element h

Bob

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Security parameter 1^n and element h

Element h'

Bob

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y \in \mathbb{Z}_q$

$h' = g^y$

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Key $k = (h')^x = g^{xy}$

Security parameter 1^n and element h

Element h'

Bob

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y \in \mathbb{Z}_q$

$h' = g^y$

Key $k = h^y = g^{xy}$

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Security parameter 1^n and element h

Element h'

Bob

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y \in \mathbb{Z}_q$

$h' = g^y$

Key $k = (h')^x = g^{xy}$

Key $k = h^y = g^{xy}$

Diffie-Hellman key exchange is EAV-secure if

a. Never
b. CDH problem hard for $\mathcal{G}$ (computational DH)
c. DDH problem hard for $\mathcal{G}$ (decisional DH)
d. DDH problem hard for $\mathcal{G}$ & additional assumptions needed

Diffie-Hellman Key Exchange

Cyclic group generator $\mathcal{G}$

Security parameter 1^n and element h

Element h'

Alice
$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$
uniform $x \in \mathbb{Z}_q$
$h = g^x$

Key $k = (h')^x = g^{xy}$

Bob
$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$
uniform $y \in \mathbb{Z}_q$
$h' = g^y$

Key $k = h^y = g^{xy}$

Diffie-Hellman key exchange is EAV-secure if

a. Never
b. CDH problem hard for $\mathcal{G}$ (computational DH)
c. DDH problem hard for $\mathcal{G}$ (decisional DH)
d. DDH problem hard for $\mathcal{G}$ & additional assumptions needed

Diffie-Hellman Key Exchange

§11.8 Theorem (Security of Diffie-Hellman key exchange)

Diffie-Hellman key exchange is EAV-secure if DDH problem hard for $\mathcal{G}$

Notes

- Only secure against eavesdropping
- Extremely insecure against any active attacker
  (e.g. Man-in-the-Middle attack)

Diffie-Hellman Key Exchange

Man-in-the-Middle attack

(Eve disrupts communication between Alice & Bob
and runs 2 separate Diffie-Hellman key exchanges with Alice & with Bob)

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Eve

Bob

Diffie-Hellman Key Exchange

Man-in-the-Middle attack

(Eve disrupts communication between Alice & Bob
and runs 2 separate Diffie-Hellman key exchanges with Alice & with Bob)

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

$(1^n, h)$

Eve

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y, \underline{x} \in \mathbb{Z}_q$

$h' = g^y$, $\underline{h} = g^{\underline{x}}$

Bob

Diffie-Hellman Key Exchange

Man-in-the-Middle attack

(Eve disrupts communication between Alice & Bob
and runs 2 separate Diffie-Hellman key exchanges with Alice & with Bob)

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

$(1^n, h)$

h'

$(1^n, \underline{h})$

Eve

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y, \underline{x} \in \mathbb{Z}_q$

$h' = g^y$, $\underline{h} = g^{\underline{x}}$

Bob

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $\underline{y} \in \mathbb{Z}_q$

$\underline{h'} = g^{\underline{y}}$

Diffie-Hellman Key Exchange

Man-in-the-Middle attack

(Eve disrupts communication between Alice & Bob
and runs 2 separate Diffie-Hellman key exchanges with Alice & with Bob)

Alice

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $x \in \mathbb{Z}_q$

$h = g^x$

Key $k = (h')^x = g^{xy}$

$(1^n, h)$

h'

$(1^n, \underline{h})$

$\underline{h'}$

Eve

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $y, \underline{x} \in \mathbb{Z}_q$

$h' = g^y$, $\underline{h} = g^{\underline{x}}$

Keys $k = h^y = g^{xy}$ and $\underline{k} = (\underline{h'})^{\underline{x}} = g^{\underline{xy}}$

Bob

$(\mathbb{G}, q, g) = \mathcal{G}(1^n)$

uniform $\underline{y} \in \mathbb{Z}_q$

$\underline{h'} = g^{\underline{y}}$

Key $\underline{k} = (\underline{h})^{\underline{y}} = g^{\underline{xy}}$

El Gamal Encryption

§11.9 Lemma

Let $(\mathbb{G}, \cdot, 1)$ be finite cyclic group and $m \in \mathbb{G}$.
Then $h_m \colon \mathbb{G} \to \mathbb{G}$ given by $h_m(a) = a \cdot m$ for all $a \in \mathbb{G}$ is bijection

Proof

It suffices to show injectivity. Let $a, b \in \mathbb{G}$ be such that $h_m(a) = h_m(b)$.
Then $a \cdot m = h_m(a) = h_m(b) = b \cdot m$ and thus $a = b$ by multiplying m^{-1} from the right. □

Note

- uniform $a \in \mathbb{G}$; return $h_m(a)$ again yields uniform distribution for any fixed (message) $m \in \mathbb{G}$

El Gamal Encryption

Taher E. Elgamal (* 1955)

- Egyptian cryptographer
- PhD student of Martin Hellman
  now CTO of Security at `Salesforce.com`
- Marconi prize 2019; father of SSL

© Alexander Klink

El Gamal Encryption

§11.10 Definition (El Gamal encryption)

Let $\mathcal{G}$ be cyclic group generator.
Public-key encryption scheme $\mathrm{ElG}(\mathcal{G}) = (\mathrm{gen}, \mathrm{enc}, \mathrm{dec})$ given by

- $\mathrm{gen}(1^n)$ runs $(\mathbb{G}, q, g) = \mathcal{G}(1^n)$; uniform $x \in \mathbb{Z}_q$; $h = g^x$ and returns $\langle (1^n, h), (1^n, x) \rangle$ (public key = h; private key = x)
- $\mathrm{enc}_{(1^n,h)}(m)$ performs uniform $y \in \mathbb{Z}_q$; return $\langle g^y, h^y \cdot m \rangle$ for all $m \in \mathbb{G}$ (formally: suitable binary encoding of group elements)
- $\mathrm{dec}_{(1^n,x)}(\langle c_1, c_2 \rangle) = c_2 \cdot c_1^{-x}$ for all $c_1, c_2 \in \mathbb{G}$

Note

- $\mathrm{dec}_{(1^n,x)}(\langle g^y, h^y \cdot m \rangle) = (\underbrace{h^y}_{=(g^x)^y} \cdot m) \cdot (g^y)^{-x} = m \cdot g^{xy} \cdot g^{-xy} = m$

  shows perfect correctness

El Gamal Encryption

§11.11 Theorem (Security of El Gamal encryption)

ElG($\mathcal{G}$) is CPA-secure if DDH problem is hard for $\mathcal{G}$

Proof

You should be able to prove this by now
(consider encryption with result $\langle g^y, g^z \cdot m \rangle$ for uniform y, z,
show that this encryption leaks no information about m,
construct DDH distinguisher that reduces to indistinguishability game
against ElG($\mathcal{A}$) or its modified version) □

El Gamal Encryption

Question: Is $\mathrm{ElG}(\mathcal{G})$ CCA-secure provided that DDH problem is hard for $\mathcal{G}$?

No!

Illustration

- Receive ciphertext $\langle c_1, c_2 \rangle$ for message m
- Select arbitrary non-unit group element $a \in \mathbb{G} \setminus \{1\}$
- Query decryption oracle for $\langle c_1, a \cdot c_2 \rangle$ and receive message m'
  (permitted since $a \cdot c_2 \neq c_2$ by $a \neq 1$)
- Message $m = a^{-1} \cdot m'$ because

$$m' = \mathrm{dec}_{(1^n,x)}(\langle c_1, a \cdot c_2 \rangle) = (a \cdot c_2) \cdot c_1^{-x} = a \cdot \mathrm{dec}_{(1^n,x)}(\langle c_1, c_2 \rangle)$$
$$= a \cdot m$$

Excursion: One-Way Permutations

Excursion: One-Way Permutations

Motivation

- Private-key cryptography had primitives
  (like pseudorandom functions, etc.)
  & generic secure constructions
- Such primitives also exist for public-key cryptography
- Seen easy to compute, but difficult to invert bijective functions
  (exponentiation vs. root & exponentiation vs. logarithm)

Excursion: One-Way Permutations

§12.1 Definition (Permutation family)

(Efficient) permutation family is pair (gen, f) of PPT functions

- Each call of $\mathsf{gen}(1^n)$ outputs parameter $\ell \in \mathbb{N}$ with $\ell \geq n$
- $f\colon \{0,1\}^* \to \{0,1\}^*$ is deterministic and $\{(x,y) \in \{0,1\}^\ell \times \{0,1\}^\ell \mid (x,y) \in f\}$ is bijective for all $\ell \in \mathsf{ran}(\mathsf{gen})$

Excursion: One-Way Permutations

Permutation family (gen, f)

Alice

$\ell \leftarrow \text{gen}(1^n)$

$x \leftarrow U_\ell$

$y = f(x)$

Eve

Excursion: One-Way Permutations

Permutation family (gen, f)

Alice

$\ell \leftarrow \mathsf{gen}(1^n)$

$x \leftarrow U_\ell$

$y = f(x)$

Challenge y

Eve

Excursion: One-Way Permutations

Permutation family (gen, f)

Alice

$\ell \leftarrow \text{gen}(1^n)$

$x \leftarrow U_\ell$

$y = f(x)$

Challenge y

Candidate x'

Eve

Determines x'

Excursion: One-Way Permutations

Permutation family (gen, f)

Alice

$\ell \leftarrow \mathsf{gen}(1^n)$

$x \leftarrow U_\ell$

$y = f(x)$

Return $x' \stackrel{?}{=} x$

Challenge y

Candidate x'

Eve

Determines x'

Excursion: One-Way Permutations

§12.2 Definition (Inversion game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\mathsf{gen}, f)$ permutation family and $\mathcal{A}$ stateful algorithm.
Inversion game $\mathsf{Inv}_{\mathcal{E},\mathcal{A}}(n)$ is

1. $\ell \leftarrow \mathsf{gen}(1^n)$, $x \leftarrow U_\ell$, and $y = f(x)$
2. Send y to adversary $\mathcal{A}$
3. $\mathcal{A}$ selects & returns element $x' \in \{0,1\}^\ell$
4. Return 1 if $x = x'$ ($\mathcal{A}$ wins) and return 0 otherwise

Note

- $\mathcal{A}$ receives image y and is expected to compute pre-image $f^{-1}(y)$

Excursion: One-Way Permutations

§12.3 Definition (One-way permutation)

Permutation family $\mathcal{E}$ is one-way if for every PPT algorithm $\mathcal{A}$

$$\mathsf{P}\left[\mathsf{Inv}_{\mathcal{E},\mathcal{A}}(n)\right] \simeq 0$$

Question: If one-way permutation families exist, then

- A Collision-resistant hash functions exist
- B Pseudorandom generators exist
- C Pseudorandom permutations exist
- D CPA-secure public-key encryption schemes exist

Excursion: One-Way Permutations

§12.4 Definition (Hardcore predicate)

Deterministic polynomial-time function $h\colon \{0,1\}^* \to \{0,1\}$ is hardcore predicate for permutation family $\mathcal{E} = (\mathrm{gen}, f)$ if for every PPT algorithm $\mathcal{A}$

$$\mathrm{E}\Big[y \stackrel{?}{=} h(x)\Big]_{\substack{\ell \leftarrow \mathrm{gen}(1^n) \\ x \leftarrow U_\ell \\ y \leftarrow \mathcal{A}(f(x))}} \simeq \frac{1}{2}$$

Notes

- Hardcore predicate easy to compute on x (polynomial-time function)
- But infeasible to compute if just given $f(x)$ (can only guess output)

Excursion: One-Way Permutations

§12.5 Theorem (Goldreich-Levin theorem)

For every one-way permutation family $\mathcal{E} = (\mathrm{gen}, f)$ there is one-way permutation family $\mathcal{E}' = (\mathrm{gen}', f')$ and hardcore predicate h for $\mathcal{E}'$

Rough idea

- $\mathrm{gen}'(1^n)$ runs $\ell \leftarrow \mathrm{gen}(1^n)$; return 2ℓ
- $f'(\langle x, r\rangle) = \langle f(x), r\rangle$ for all $x, r \in \{0,1\}^\ell$ and $\ell \in \mathrm{ran}(\mathrm{gen})$
- $h(\langle x, r\rangle) = \bigoplus_{i=1}^{\ell} x_i \cdot r_i$ for all $x, r \in \{0,1\}^\ell$ and $\ell \in \mathrm{ran}(\mathrm{gen})$
  (XOR of bits of x selected by r)

Excursion: One-Way Permutations

§12.6 Theorem

Let h be hardcore predicate for one-way permutation family $\mathcal{E} = (\mathrm{gen}, f)$.
For every $n \in \mathbb{N}$, let $\ell(n) = \max\{i \in \mathrm{ran}(\mathrm{gen}) \mid i \leq n\}$ and
$G\colon \{0,1\}^* \to \{0,1\}^*$ is given for all $s \in \{0,1\}^n$ by

$$G(s's'') = f(s')\, h(s')\, s''$$

where $s = s's''$ with $|s'| = \ell(n)$.

Then G is pseudorandom generator with expansion $p(n) = n + 1$.

Notes

- Longer expansions possible by Exercise 4.2
- Pseudorandom function from pseudorandom generator (Theorem §6.2)
- Pseudorandom permutation from pseudorandom function by Feistel network (Theorem §9.2)

Excursion: One-Way Permutations

Primitive for public-key encryption (Section 13.1 of [Katz & Lindell])

- Need to add way to easily invert given additional knowledge
- Add additional secret component, called trapdoor, to generator
- Add efficient inversion algorithm that reliably inverts provided that correct trapdoor is supplied
- Trivially adjust hardcore predicate
- Construct CPA-secure public-key encryption scheme

Excursion: One-Way Permutations

Final remarks

- Collision-resistant compression function via hard discrete logarithm
  (Hash function via Merkle-Damgård transform; Theorem §9.7)
  1. Take cyclic group generator $\mathcal{G}$ with hard discrete logarithm
  2. On security parameter n run $\mathcal{G}(1^n) = (\mathbb{G}, q, g)$ and uniform $h \in \mathbb{G}$
  3. On input $x, y \in \mathbb{Z}_q$ return hash $f(x, y) = g^x h^y$
- $f : \mathbb{Z}_q^2 \to \mathbb{G}$ with $|\mathbb{Z}_q| = |\mathbb{G}|$ so compression given suitable encoding
- If discrete logarithm easy, then determine $z \in \mathbb{Z}_p$ such that $h = g^z$.
  Construct collision $(x + z, y - 1)$

$$f(x + z, y - 1) = g^{x+z}(g^z)^{y-1} = g^{x+z+z(y-1)} = g^{x+zy} = f(x, y)$$

- Given collision $g^x h^y = f(x, y) = f(x', y') = g^{x'} h^{y'}$ we have
  $g^{x-x'} = h^\Delta$ with $\Delta = y' - y$. Also $(g^{x-x'})^{\Delta^{-1}} = (h^\Delta)^{\Delta^{-1}} = h$,
  so $\log_g(h) = (x - x')(y' - y)^{-1}$ is logarithm of h for base g

Summary

- Diffie-Hellman key exchange
- El Gamal encryption
- One-way permutations