Incompressible Encryption

Mahesh Sreekumar Rajasree
Post-Doctoral Fellow
Department of Computer Science & Engineering
IIT Delhi

(This is joint work with Rishab Goyal, Venkata Koppula and Aman Verma)

Contents

Contents

- Introduction

Contents

Contents

- Introduction
- Security Definitions
- Incompressible SKE & PKE

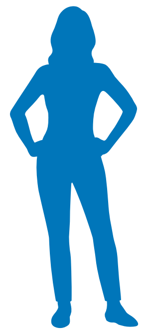
Contents

Contents

Introduction

Encryption Scheme

Encryption Scheme



ALICE

Encryption Scheme

ALICE

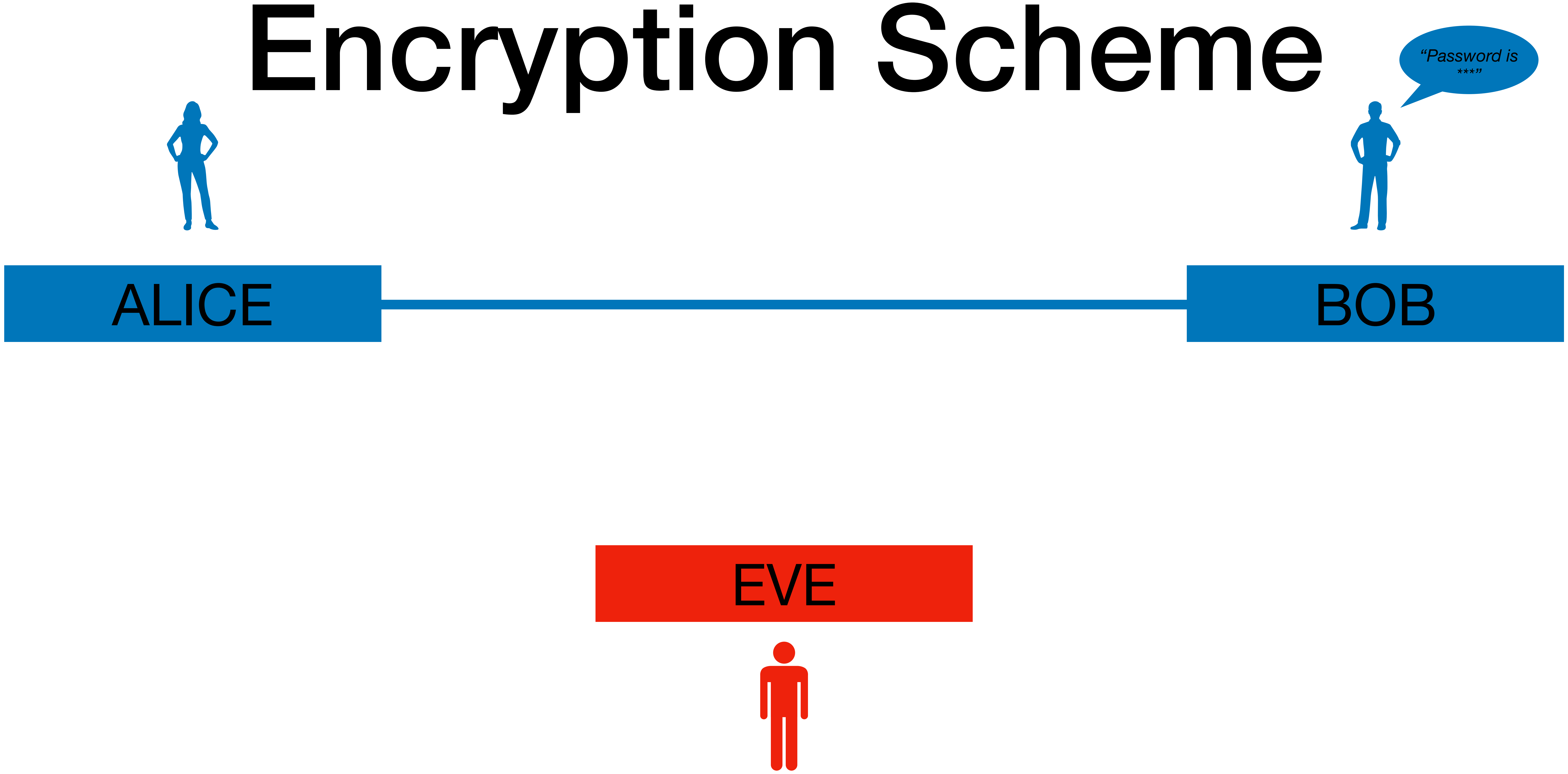
BOB

Encryption Scheme

ALICE

BOB

Encryption Scheme

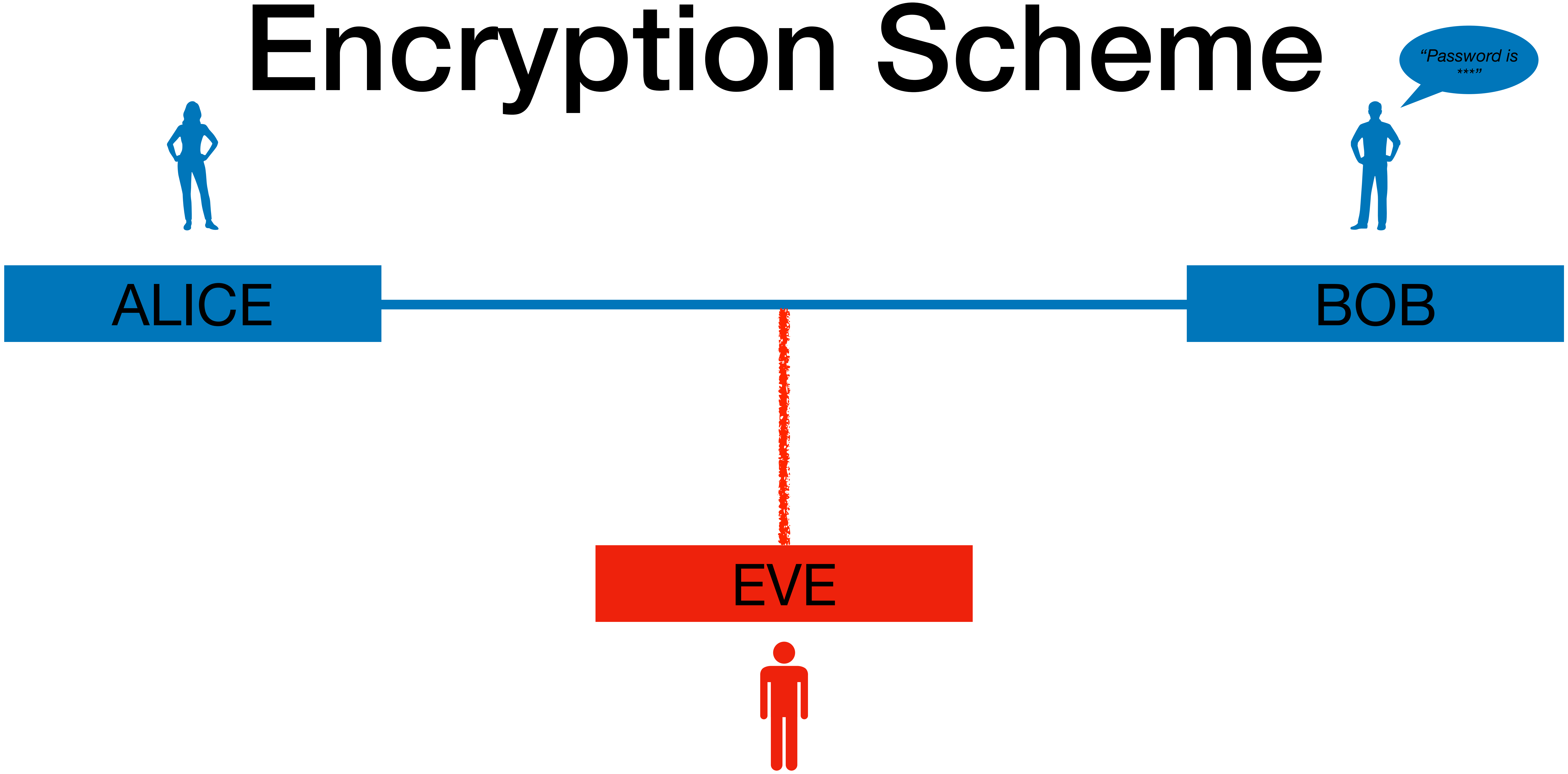


ALICE

BOB

EVE

Encryption Scheme

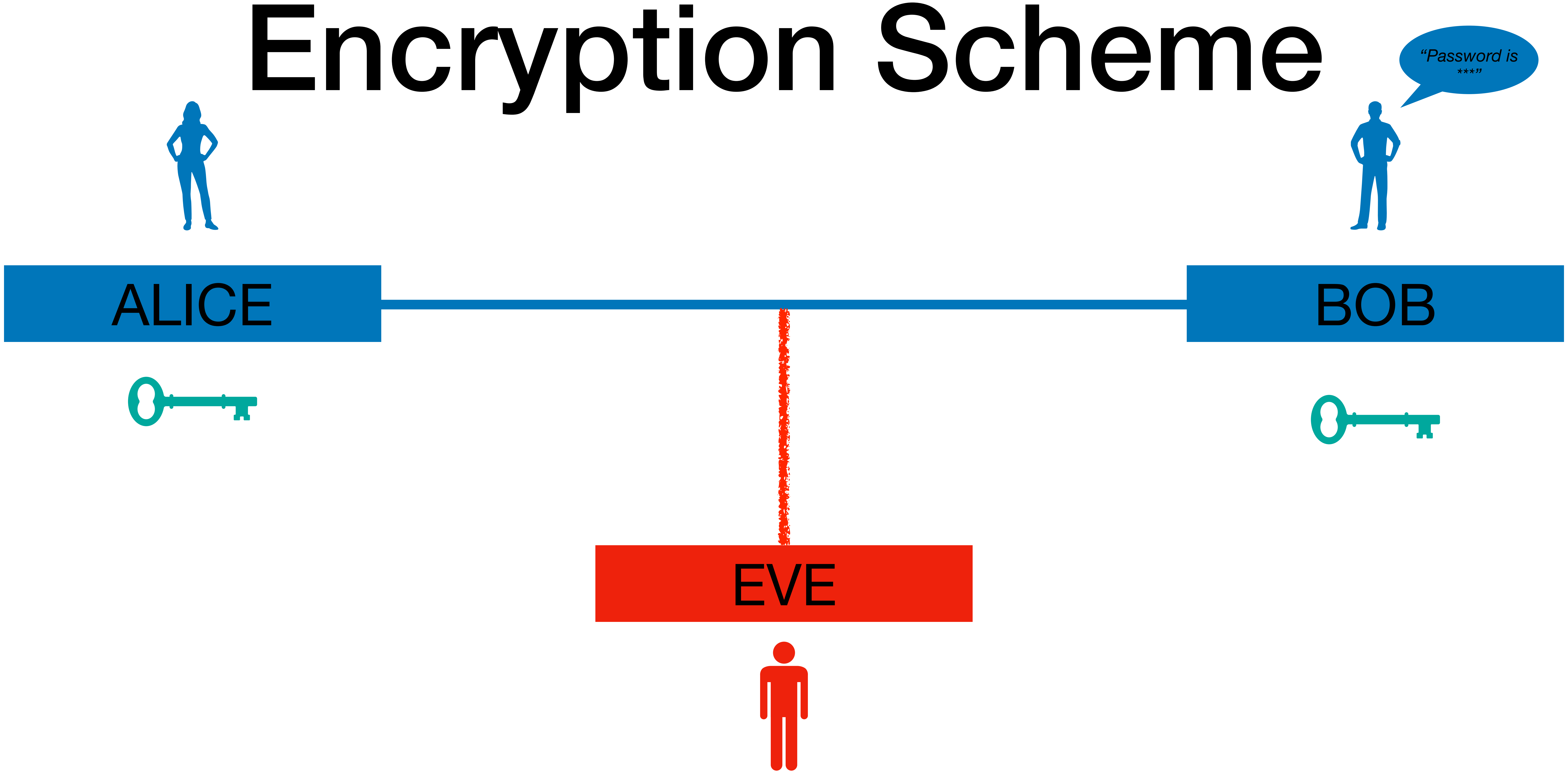


ALICE

BOB

EVE

Encryption Scheme

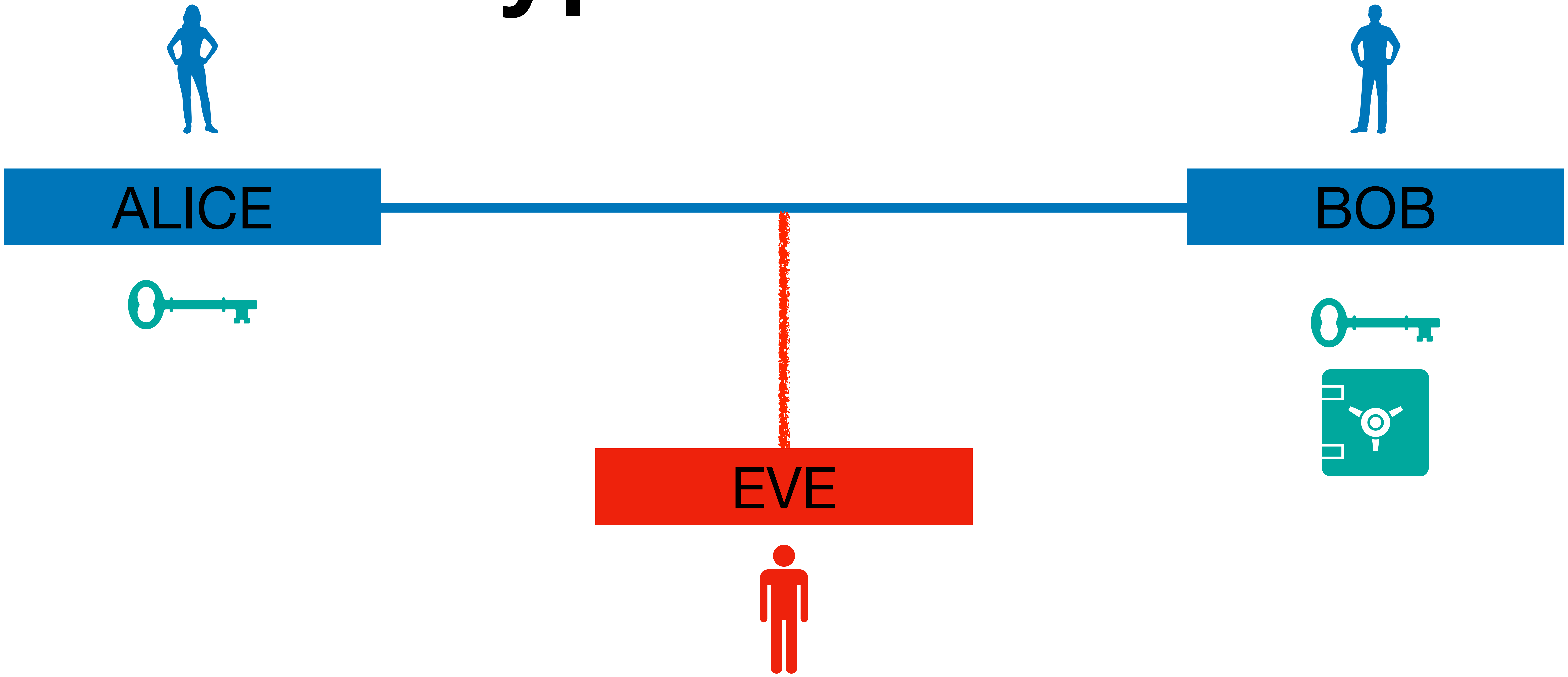


ALICE

BOB

EVE

Encryption Scheme

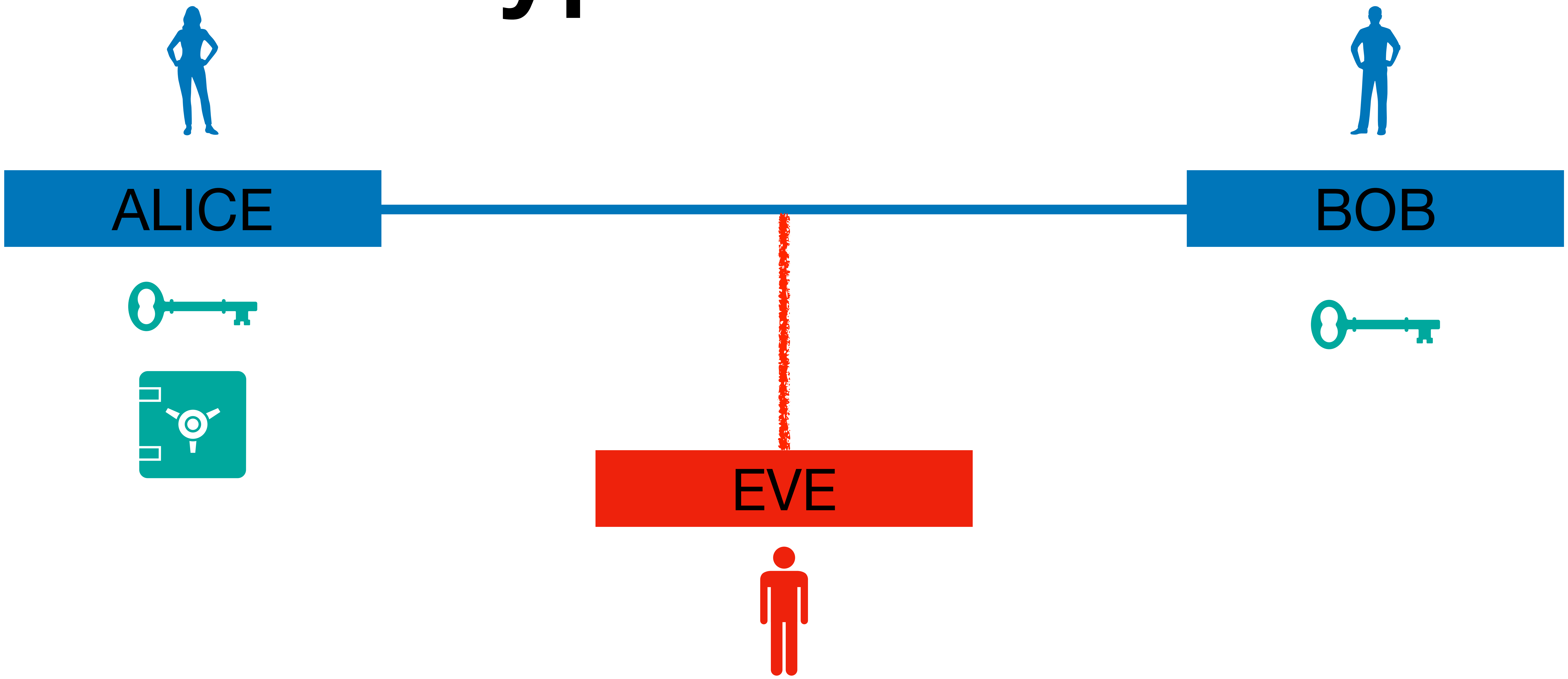


ALICE

BOB

EVE

Encryption Scheme

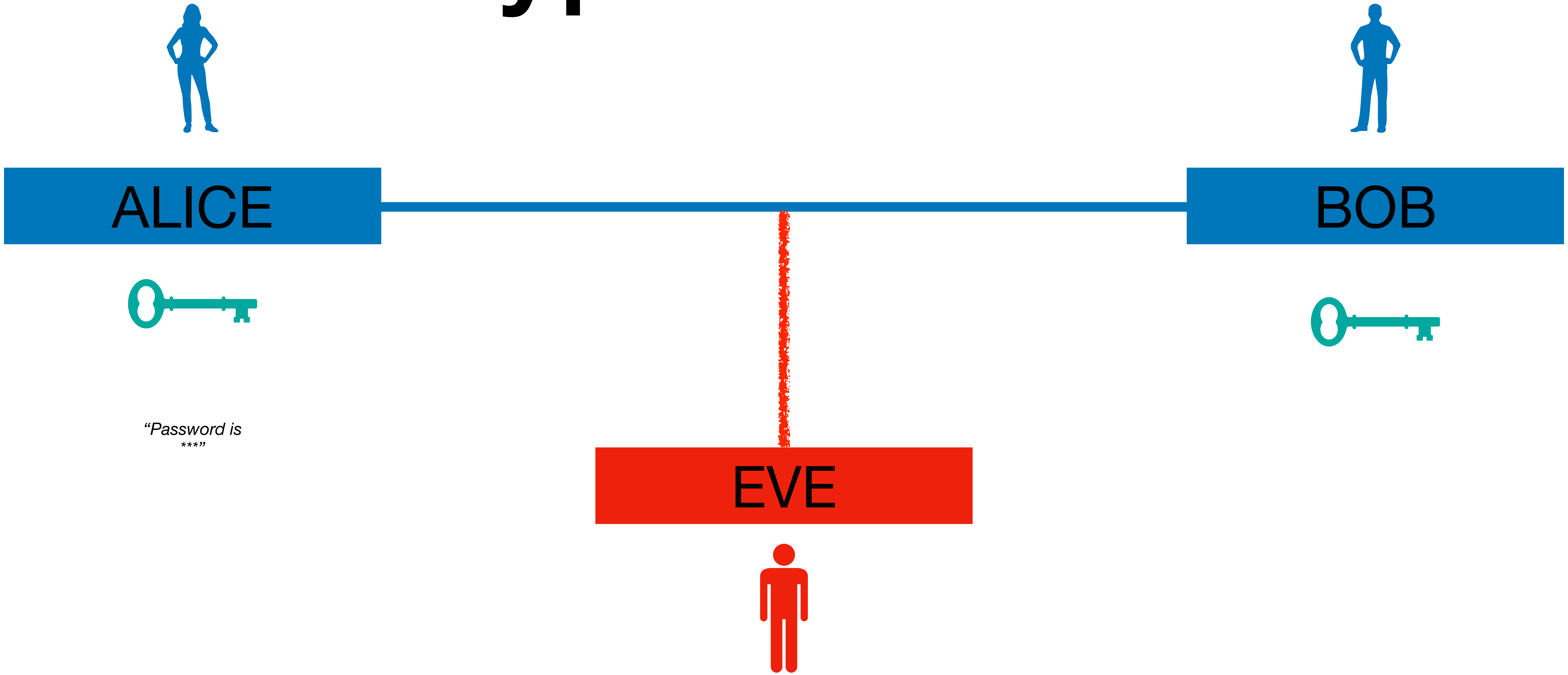


ALICE

BOB

EVE

Encryption Scheme



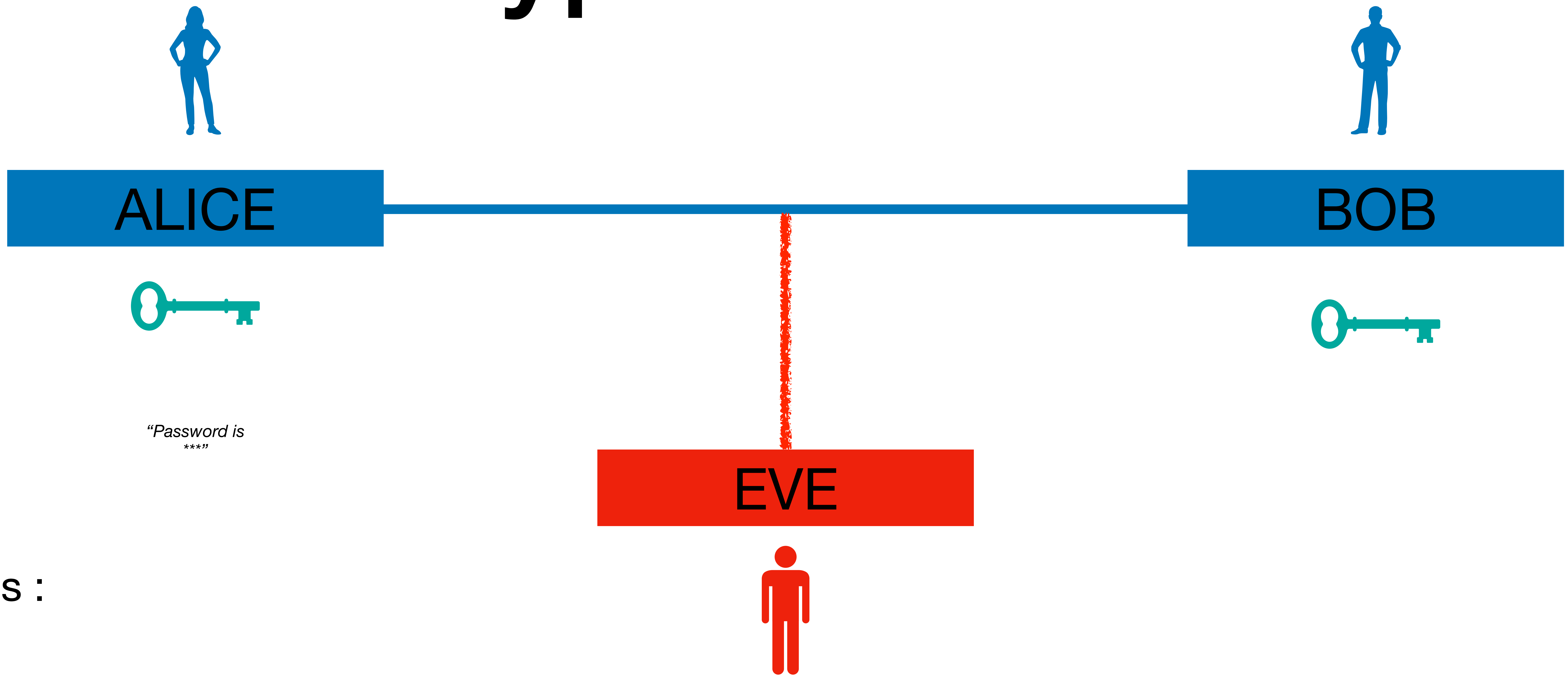
ALICE

BOB

EVE

*“Password is ***”*

Encryption Scheme



ALICE

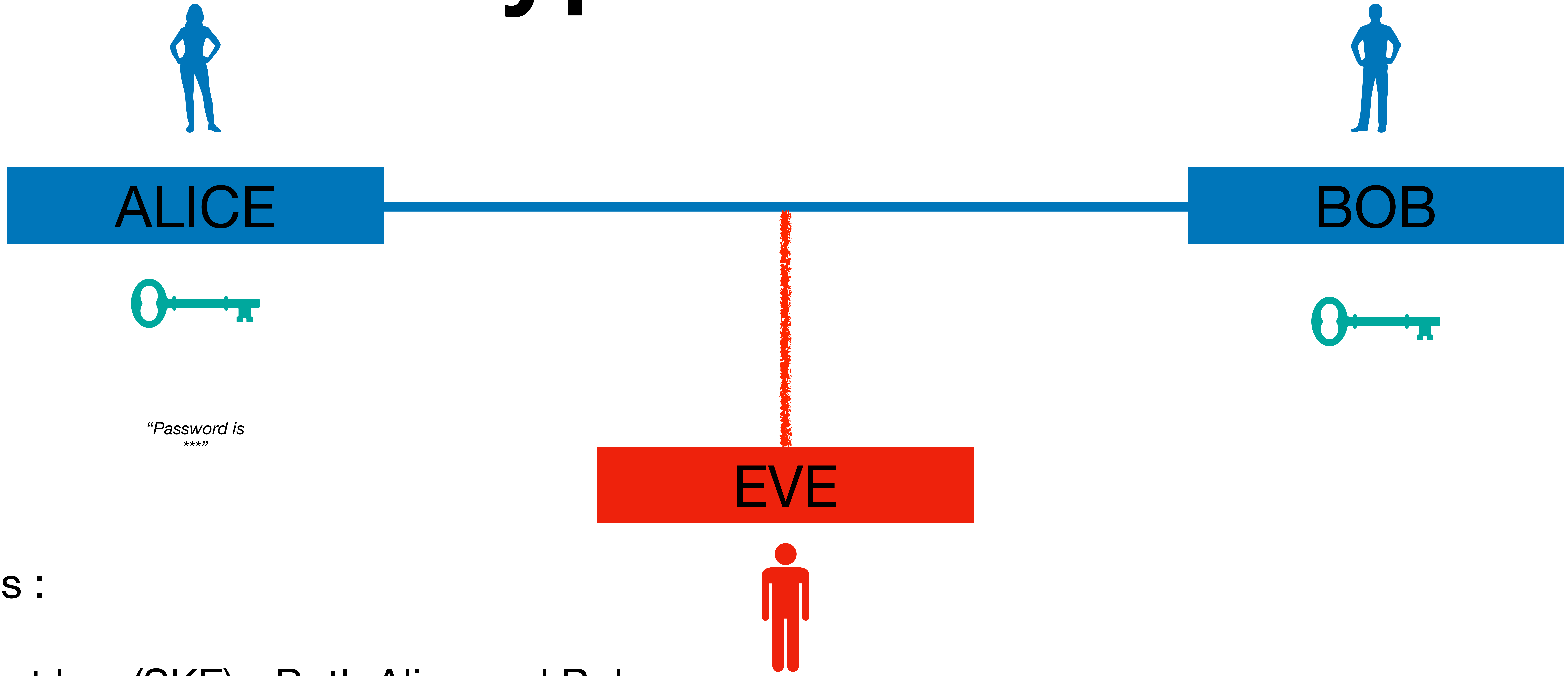
BOB

EVE

*“Password is ***”*

2 types :

Encryption Scheme



ALICE

BOB

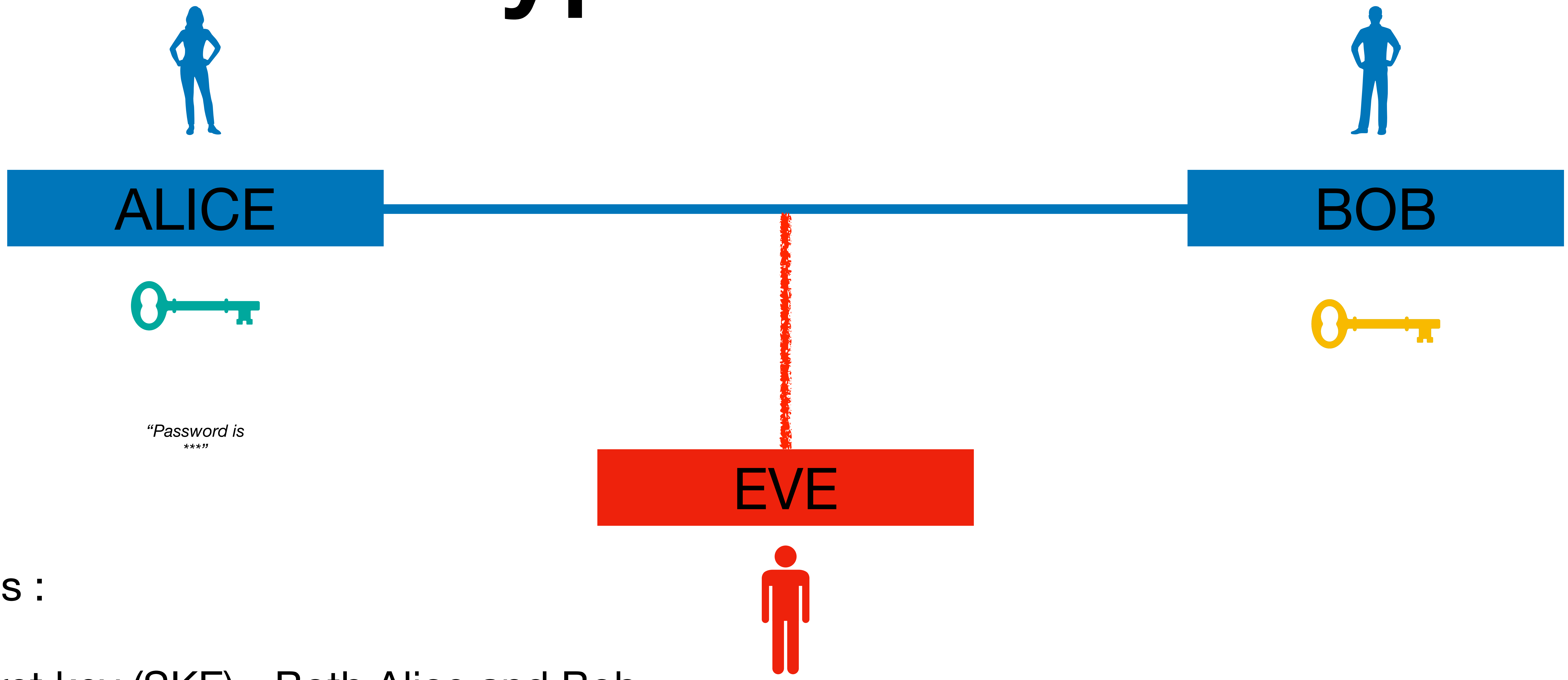
“Password is ***”

EVE

2 types :

- secret key (SKE) - Both Alice and Bob have the same key.

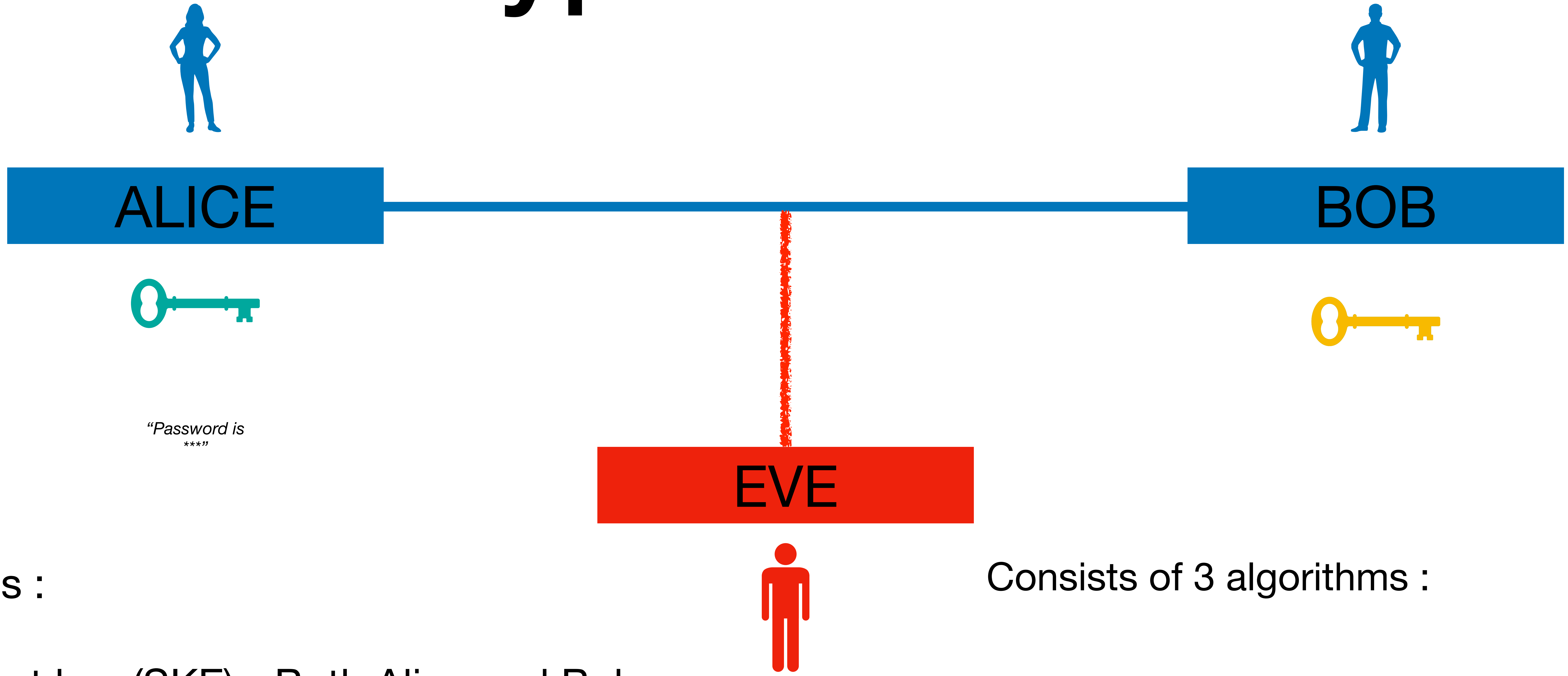
Encryption Scheme



2 types :

- secret key (SKE) - Both Alice and Bob have the same key.
- public key (PKE) - Encryptor has public key and decrypt has secret key.

Encryption Scheme

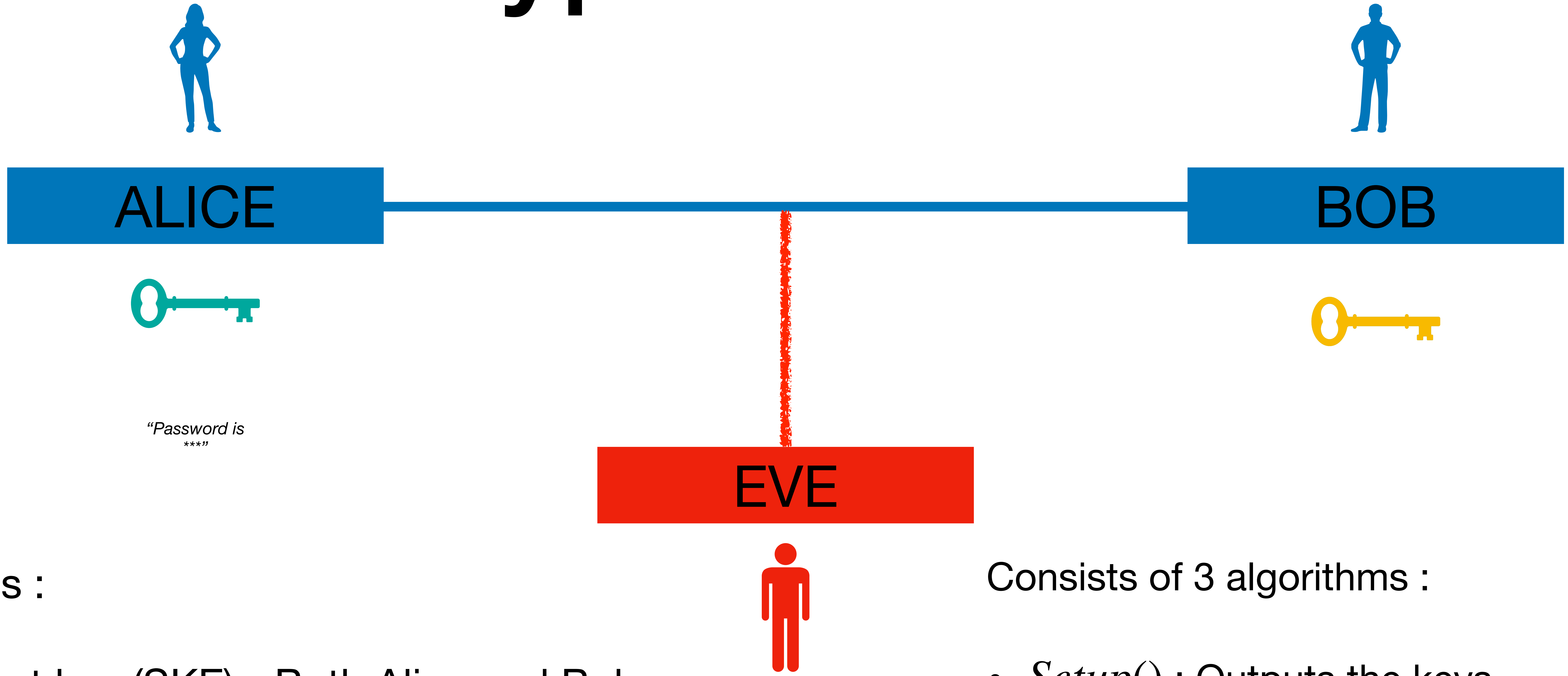


2 types :

- secret key (SKE) - Both Alice and Bob have the same key.
- public key (PKE) - Encryptor has public key and decrypt has secret key.

Consists of 3 algorithms :

Encryption Scheme



ALICE

BOB

EVE

“Password is ***”

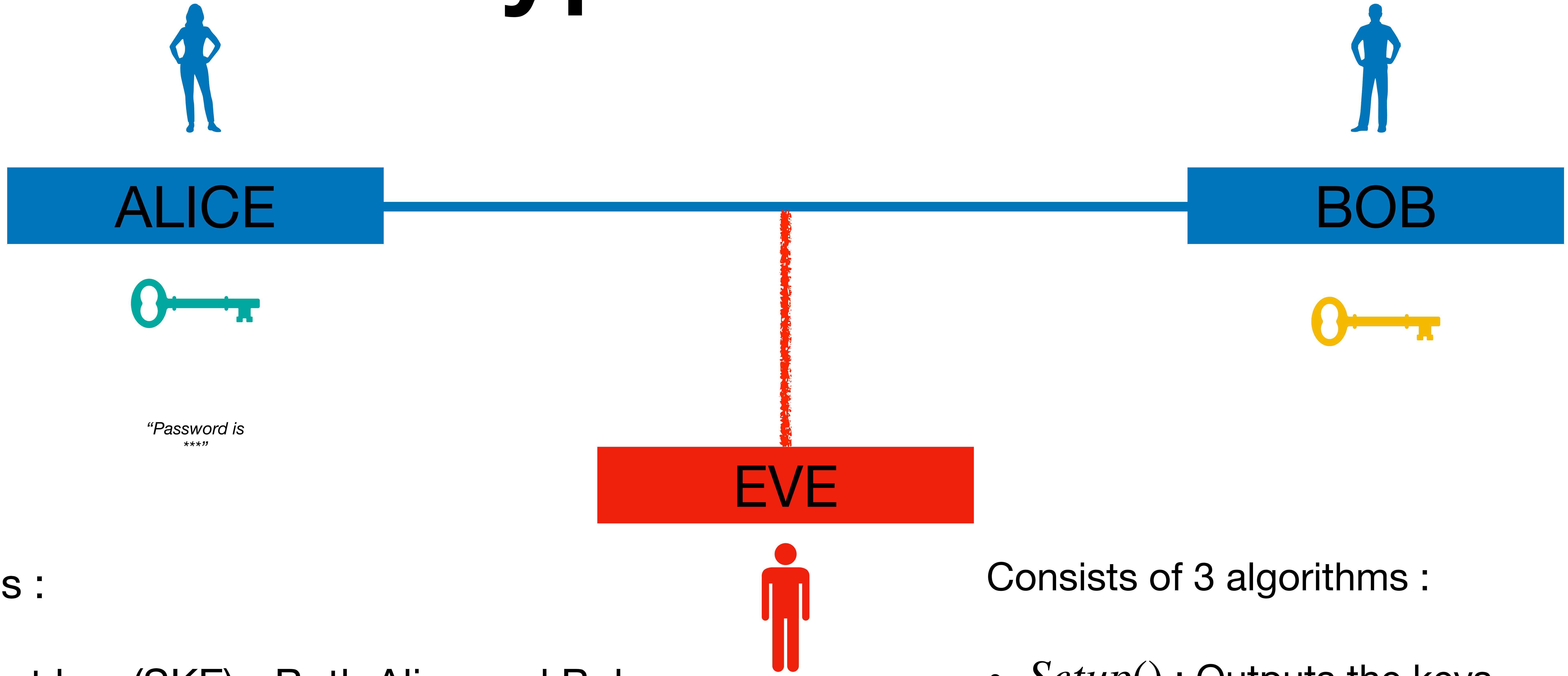
2 types :

- secret key (SKE) - Both Alice and Bob have the same key.
- public key (PKE) - Encryptor has public key and decrypt has secret key.

Consists of 3 algorithms :

- $Setup()$: Outputs the keys

Encryption Scheme



ALICE

BOB

“Password is ***”

EVE

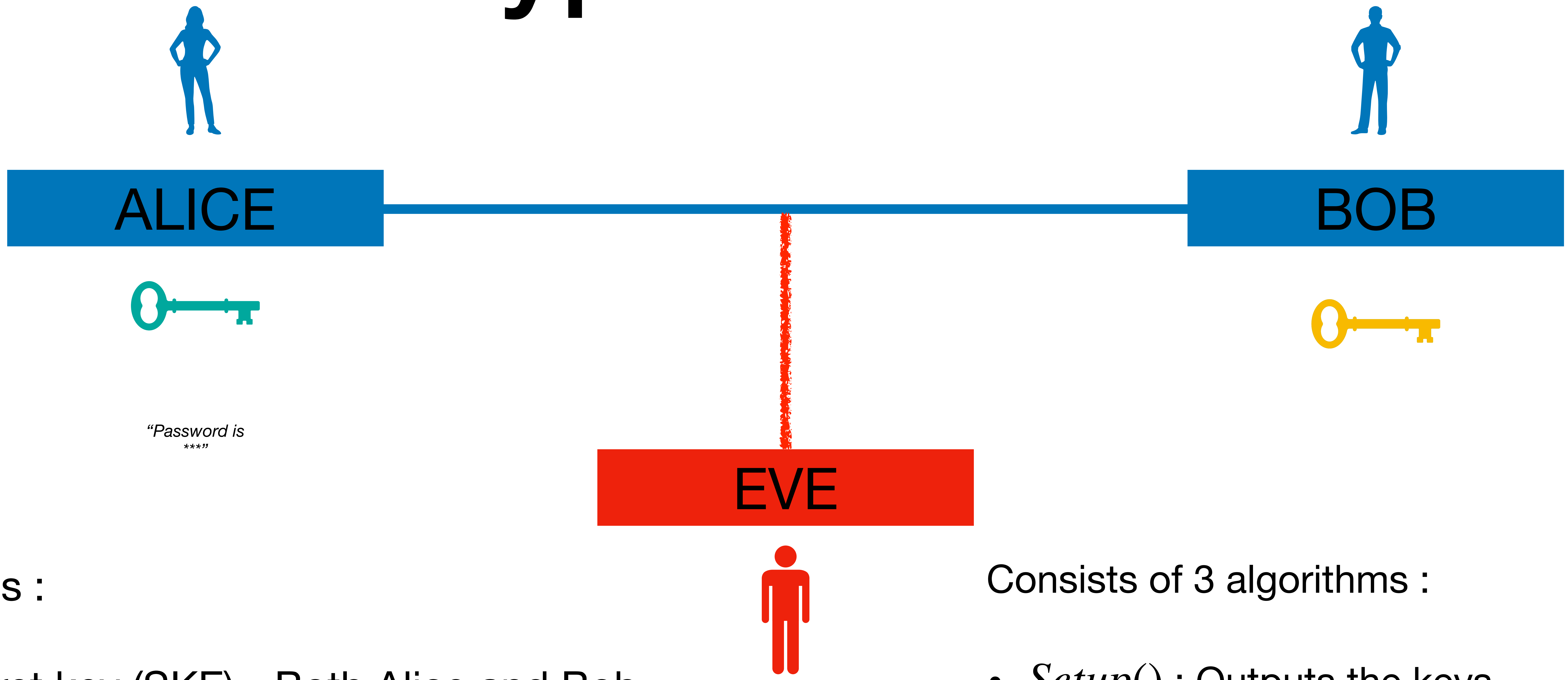
2 types :

- secret key (SKE) - Both Alice and Bob have the same key.
- public key (PKE) - Encryptor has public key and decrypt has secret key.

Consists of 3 algorithms :

- $Setup()$: Outputs the keys
- $Enc(pk/sk, m)$: Outputs ciphertext

Encryption Scheme



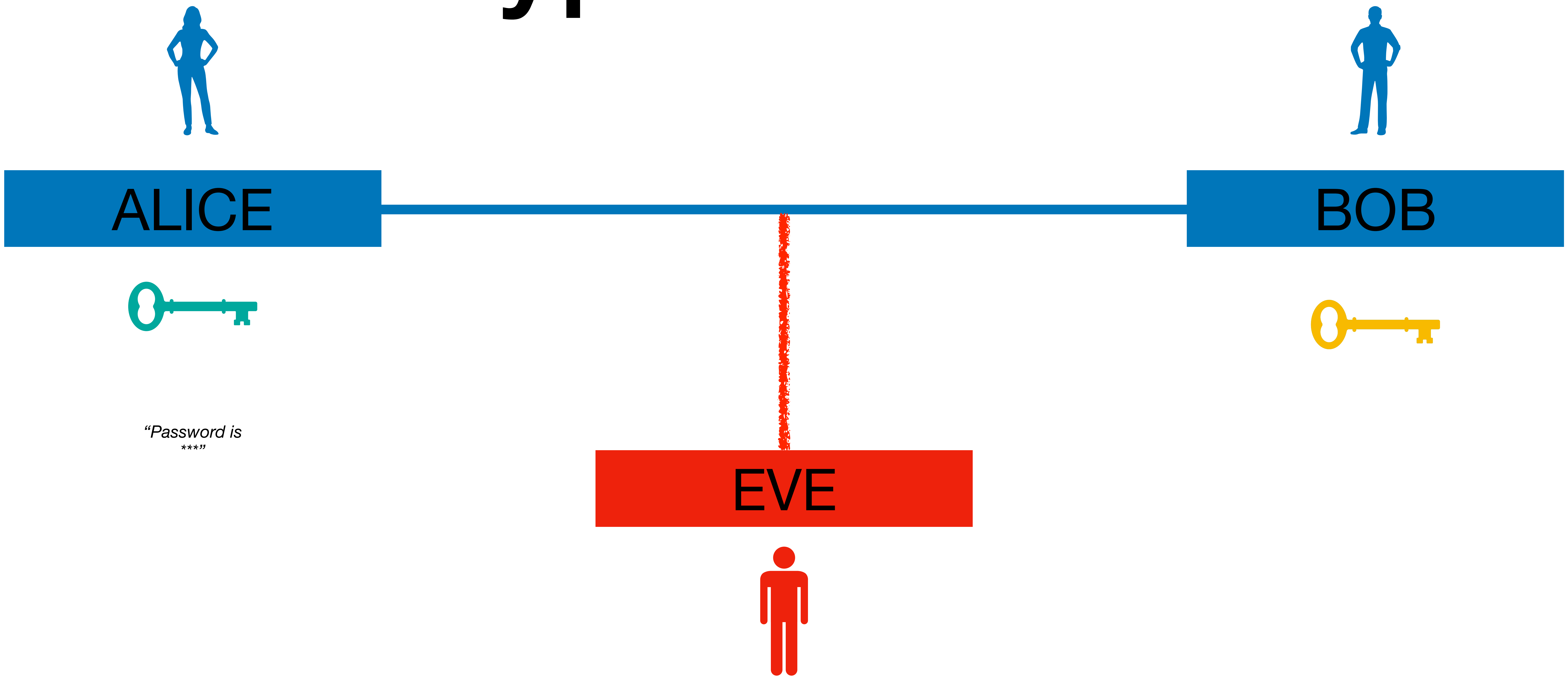
2 types :

- secret key (SKE) - Both Alice and Bob have the same key.
- public key (PKE) - Encryptor has public key and decrypt has secret key.

Consists of 3 algorithms :

- $Setup()$: Outputs the keys
- $Enc(pk/sk, m)$: Outputs ciphertext
- $Dec(sk, c)$: Outputs message or error

Encryption Scheme



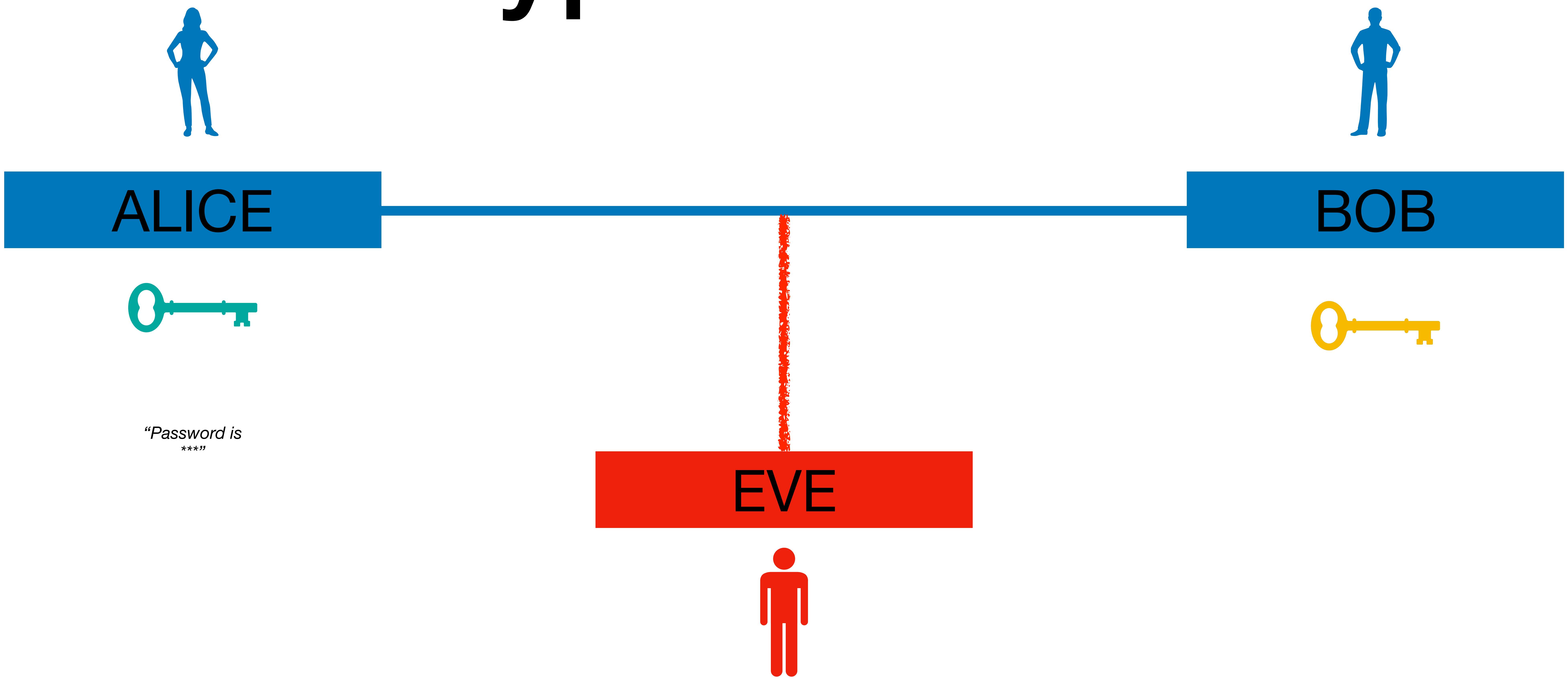
ALICE

BOB

“Password is ***”

EVE

Encryption Scheme



ALICE

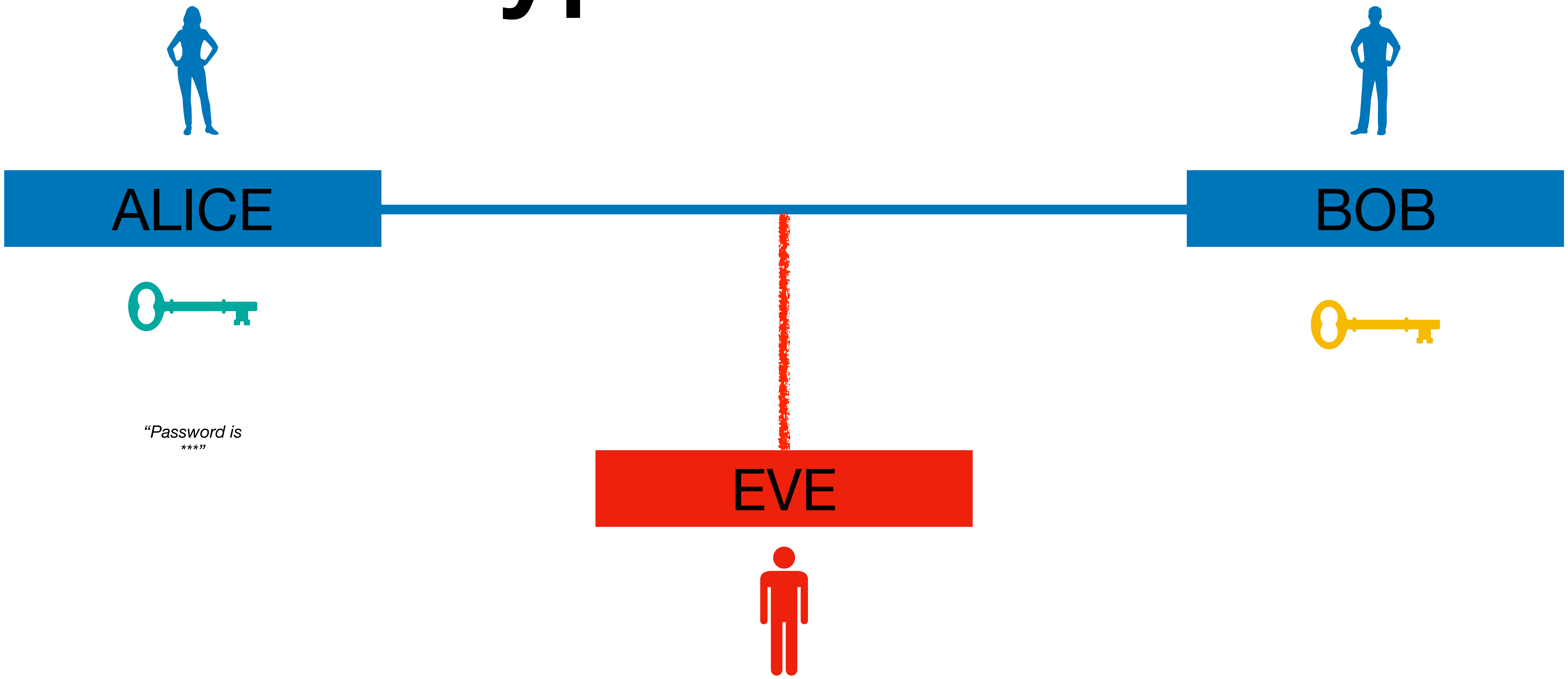
BOB

EVE

*"Password is ***"*

- Correctness - $Dec(sk, Enc(pk, m)) = m$

Encryption Scheme



- Correctness - $Dec(sk, Enc(pk, m)) = m$

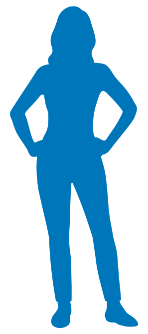
- Security - …….

Security Definitions

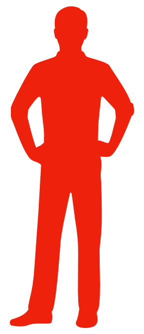
Standard Security [Goldwaser,Micali84]

Standard Security [Goldwaser,Micali84]

Standard Security [Goldwaser,Micali84]

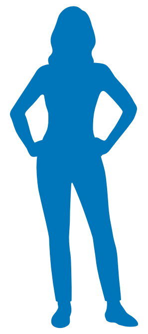


Challenger

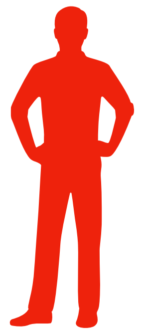


Adversary

Standard Security [Goldwaser,Micali84]



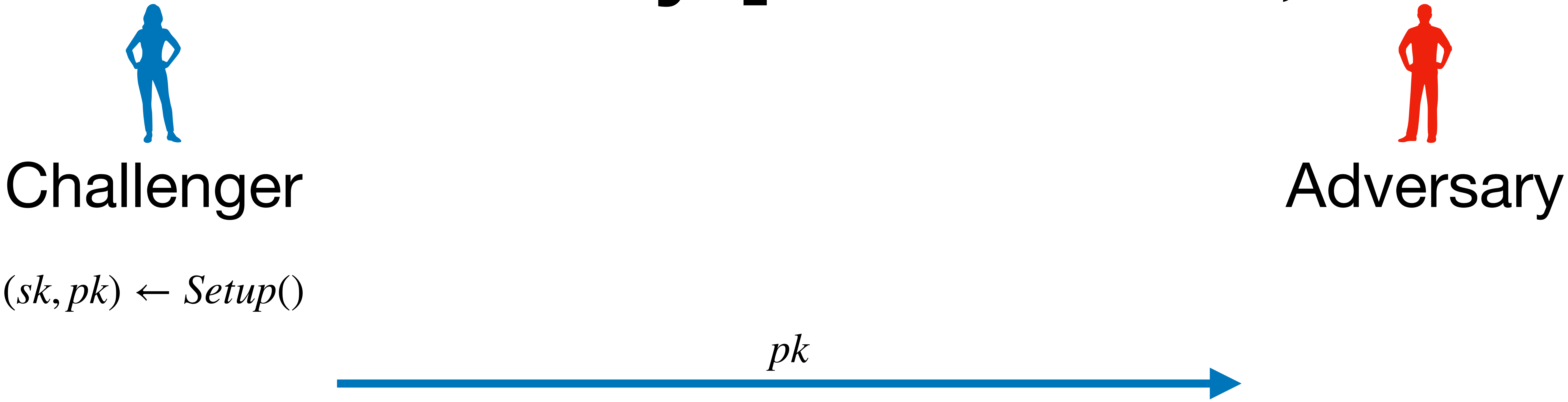
Challenger



Adversary

$(sk, pk) \leftarrow Setup()$

Standard Security [Goldwaser,Micali84]



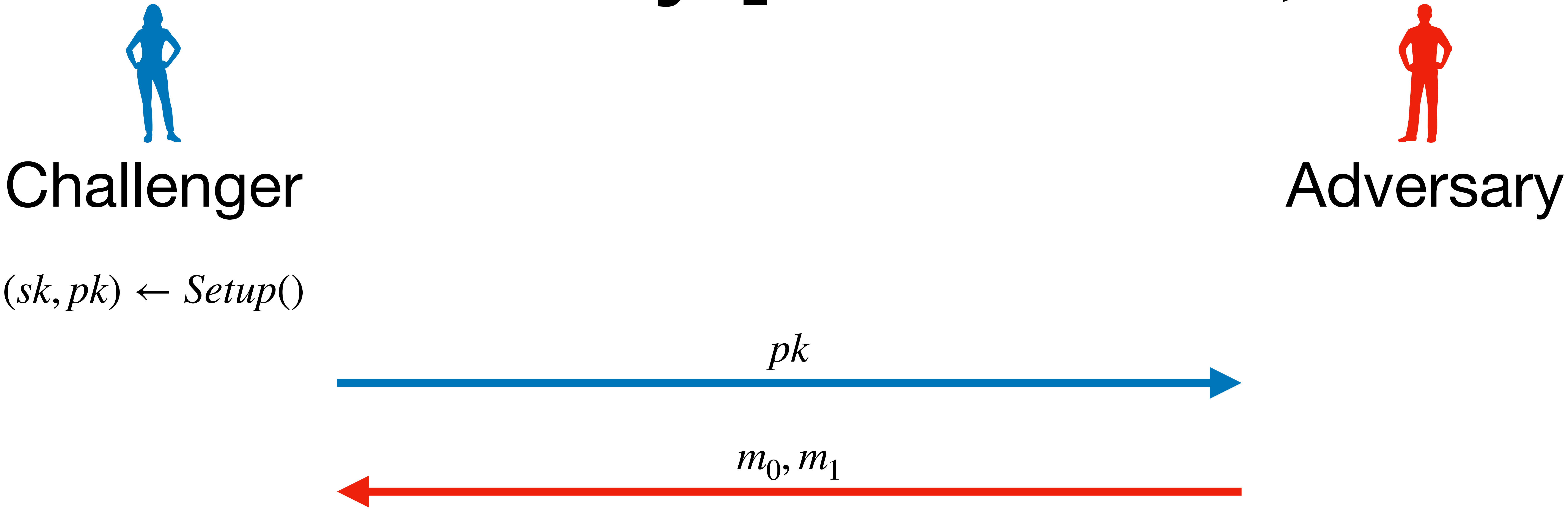
Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

$$\xrightarrow{pk}$$

Standard Security [Goldwaser,Micali84]



Challenger

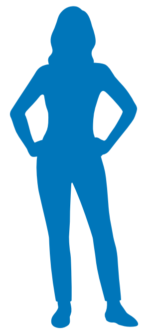
Adversary

$(sk, pk) \leftarrow Setup()$

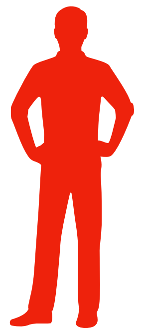
pk

m_0, m_1

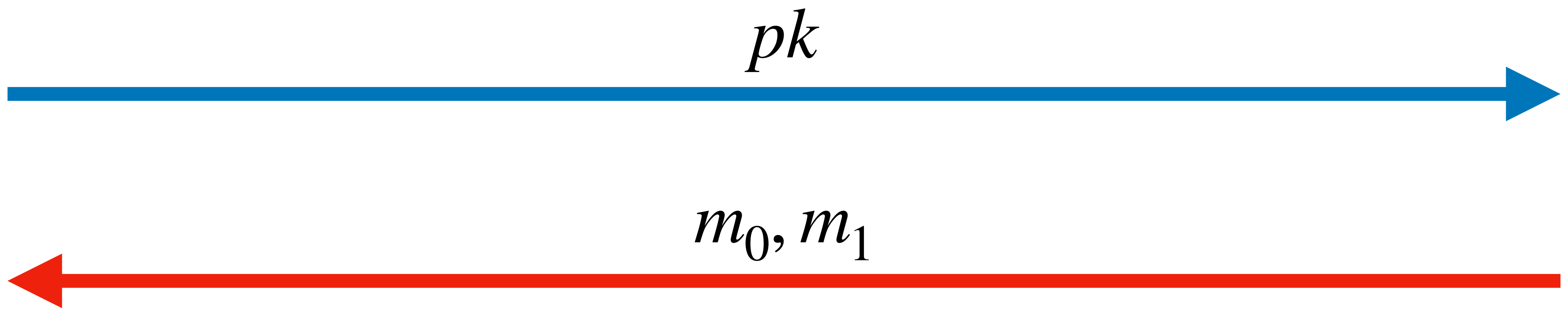
Standard Security [Goldwaser,Micali84]



Challenger



Adversary



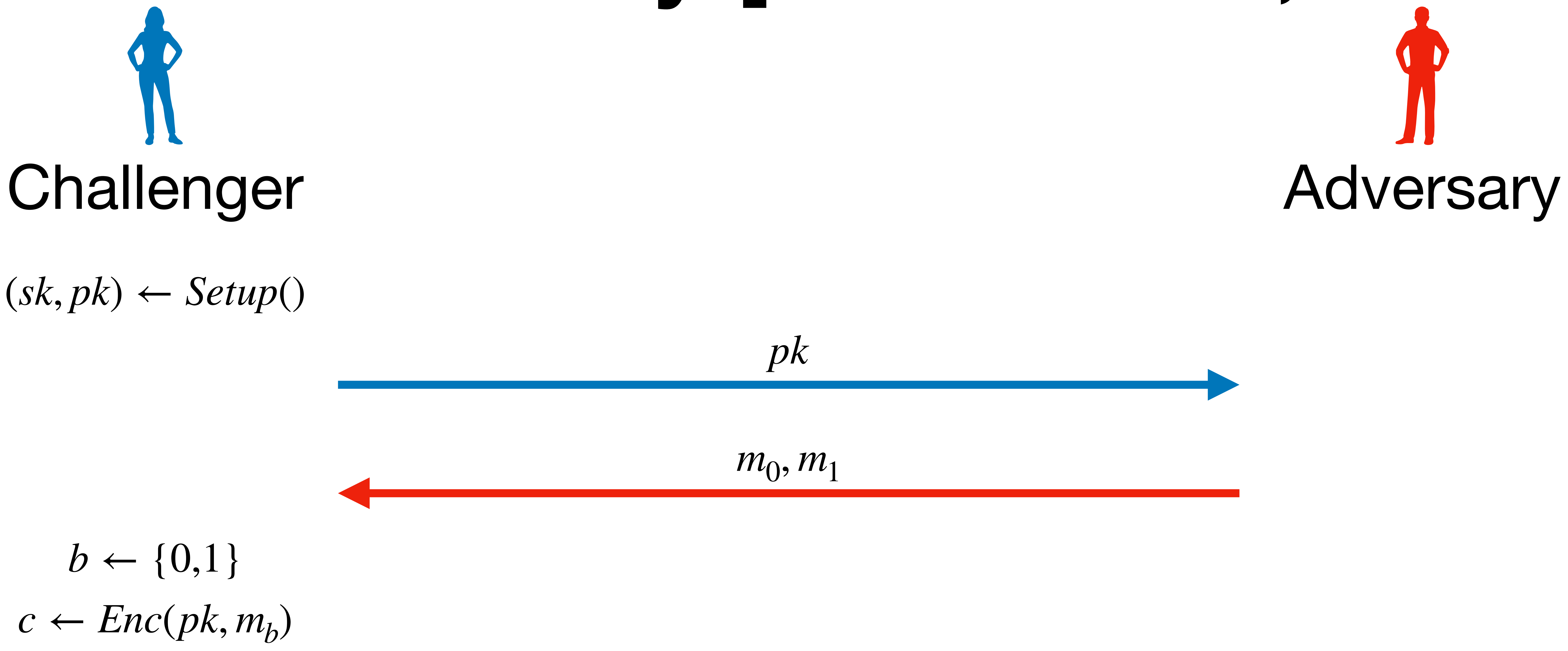
$(sk, pk) \leftarrow Setup()$

pk →

← m_0, m_1

$b \leftarrow \{0,1\}$

Standard Security [Goldwaser,Micali84]



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

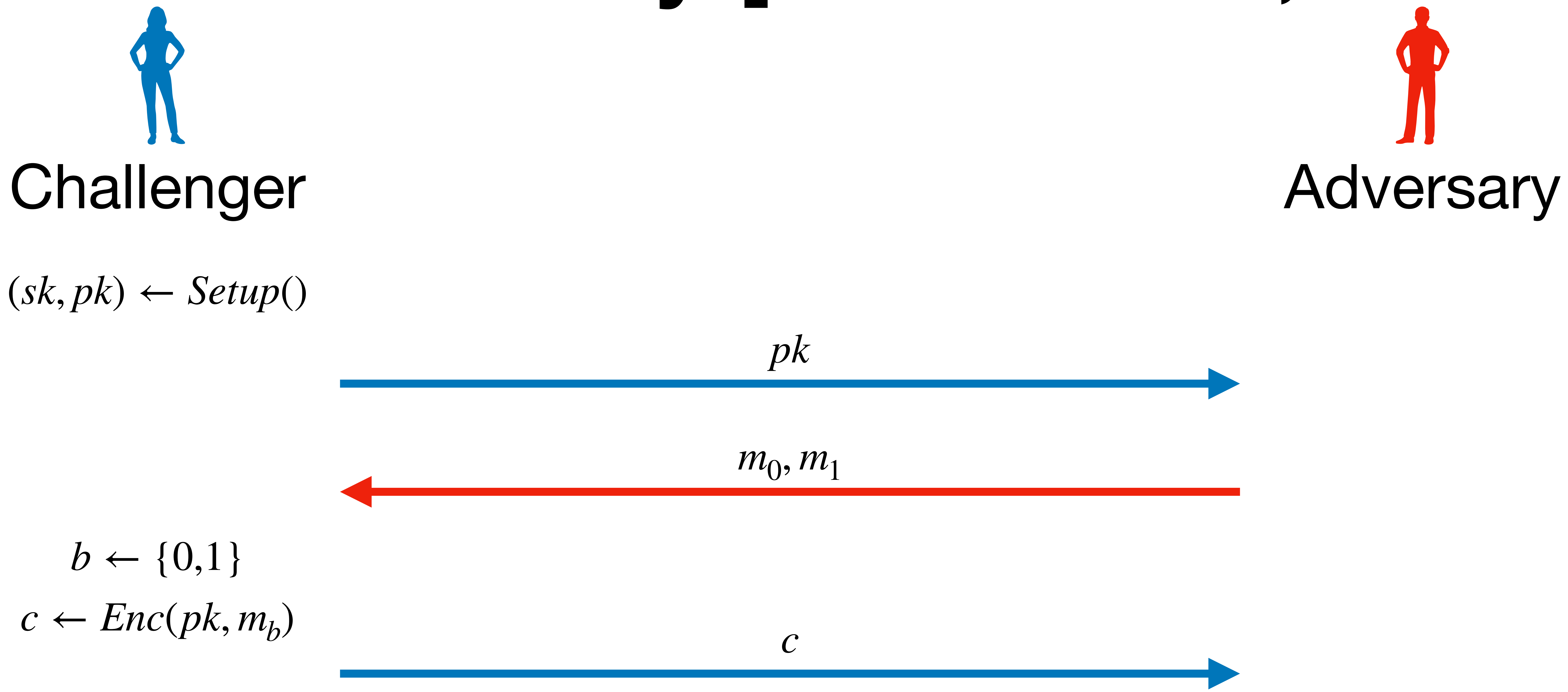
pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

Standard Security [Goldwaser,Micali84]



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

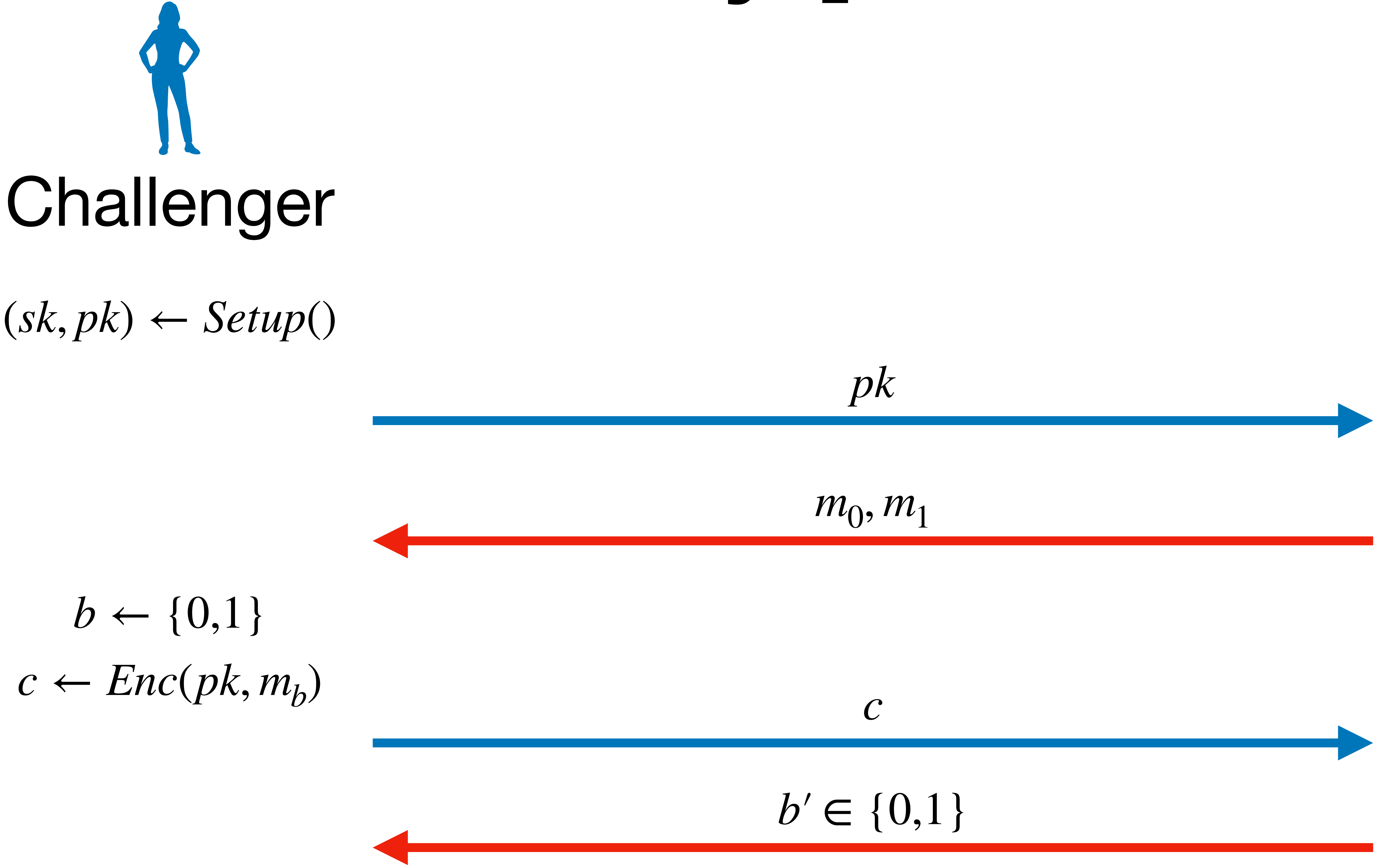
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

Standard Security [Goldwaser,Micali84]



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

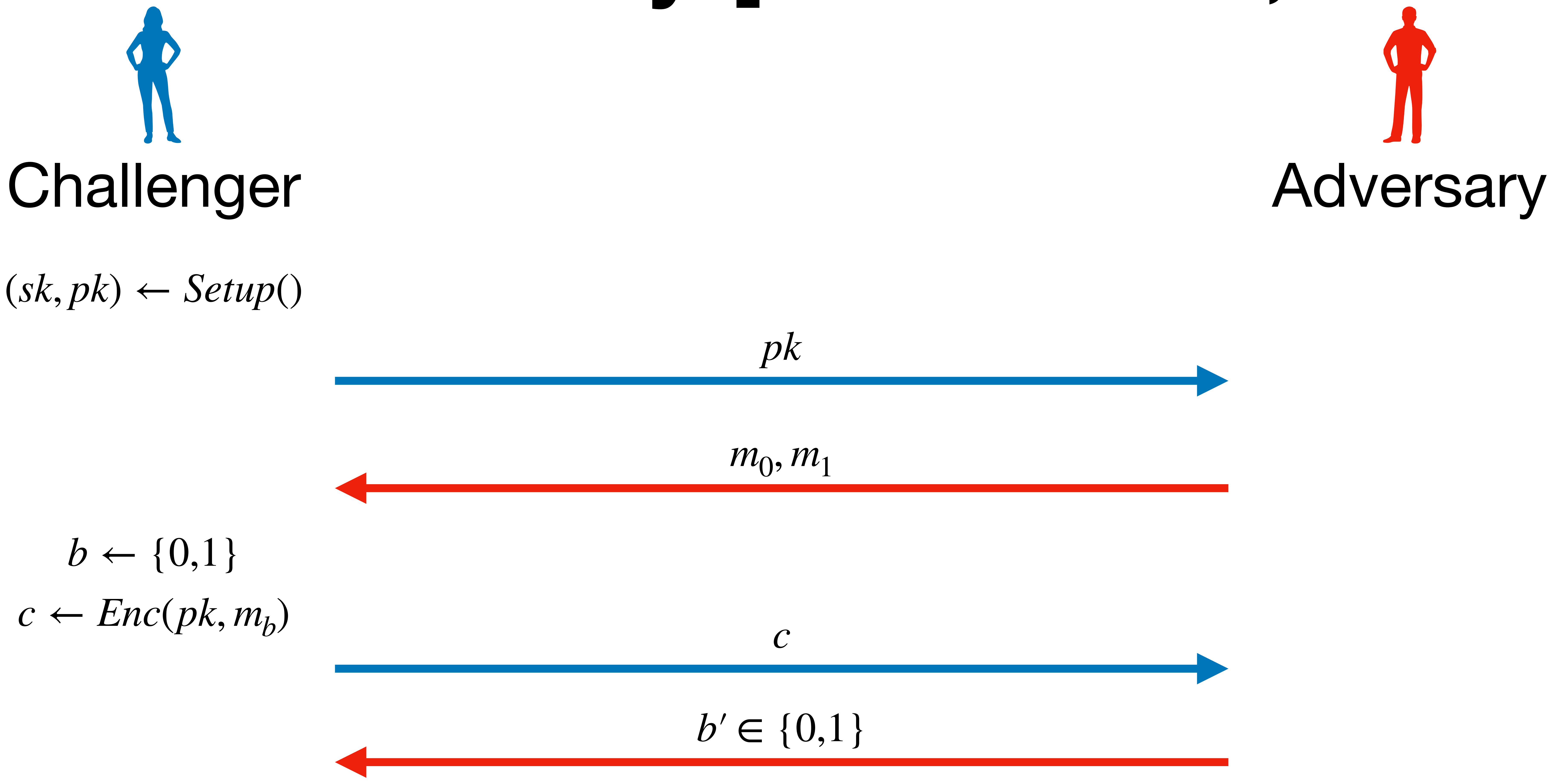
$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

$b' \in \{0,1\}$

Standard Security [Goldwaser,Micali84]



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

$b' \in \{0,1\}$

Adversary wins if $b = b'$

Can Secret Key be leaked?

Can Secret Key be leaked?

- Standard security says that adversary cannot distinguish between encryptions of two different message provided **no** information of secret key is leaked.

Can Secret Key be leaked?

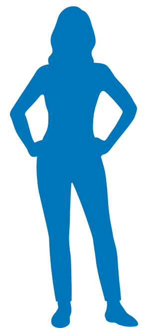
- Standard security says that adversary cannot distinguish between encryptions of two different message provided **no** information of secret key is leaked.

- In practice, secret key can be leaked using side-channel attacks.

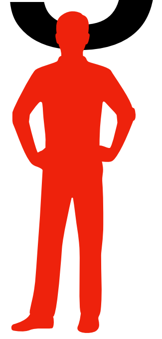
Security against Leakage

Security against Leakage

Security against Leakage

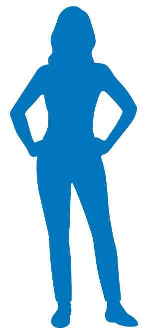


Challenger

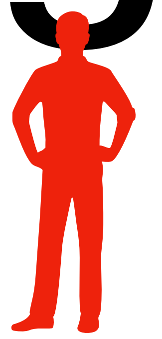


Adversary

Security against Leakage



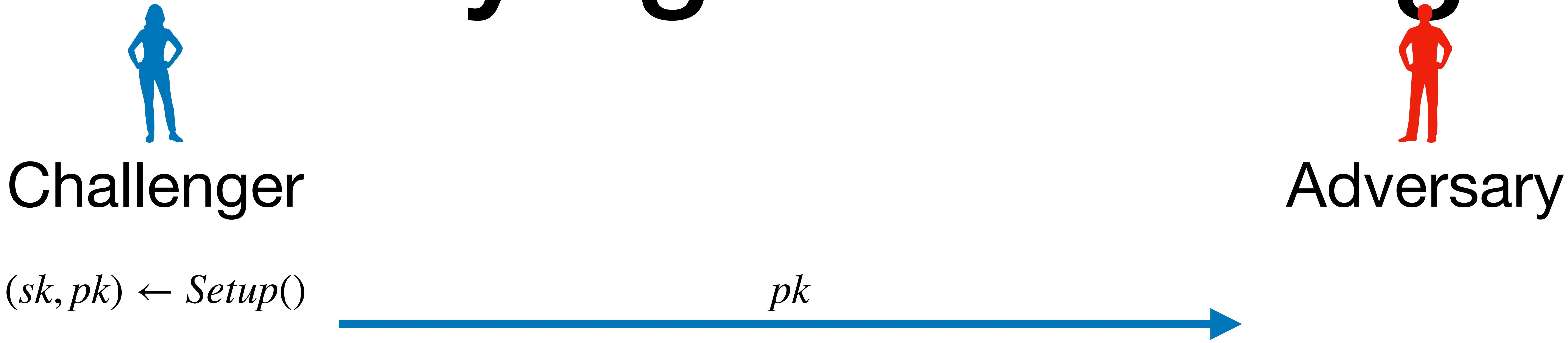
Challenger



Adversary

$(sk, pk) \leftarrow Setup()$

Security against Leakage



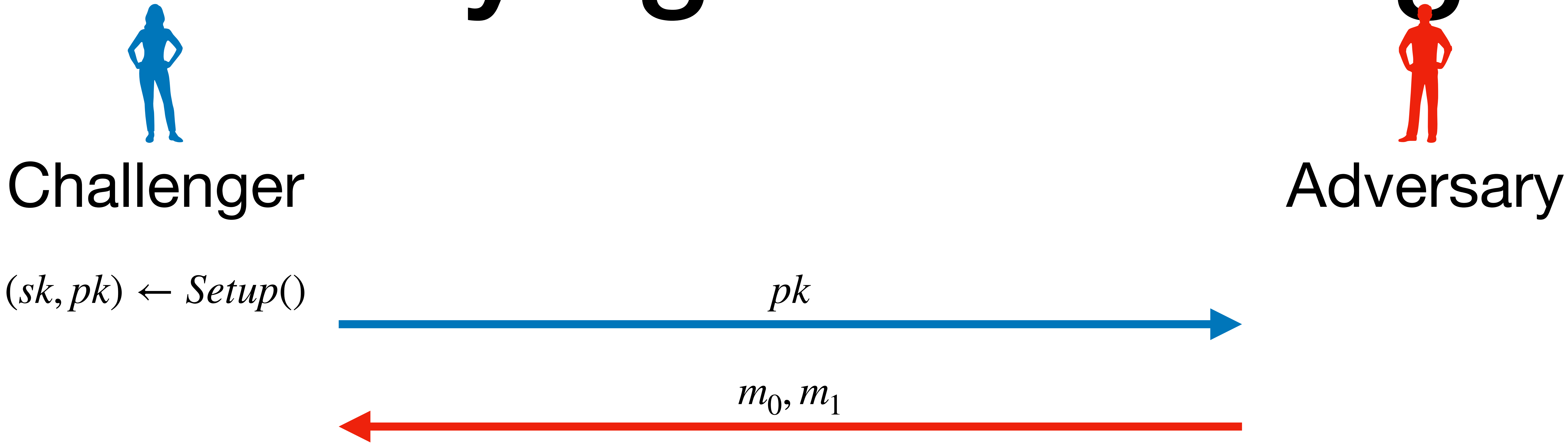
Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

$\xrightarrow{pk}$

Security against Leakage



Challenger

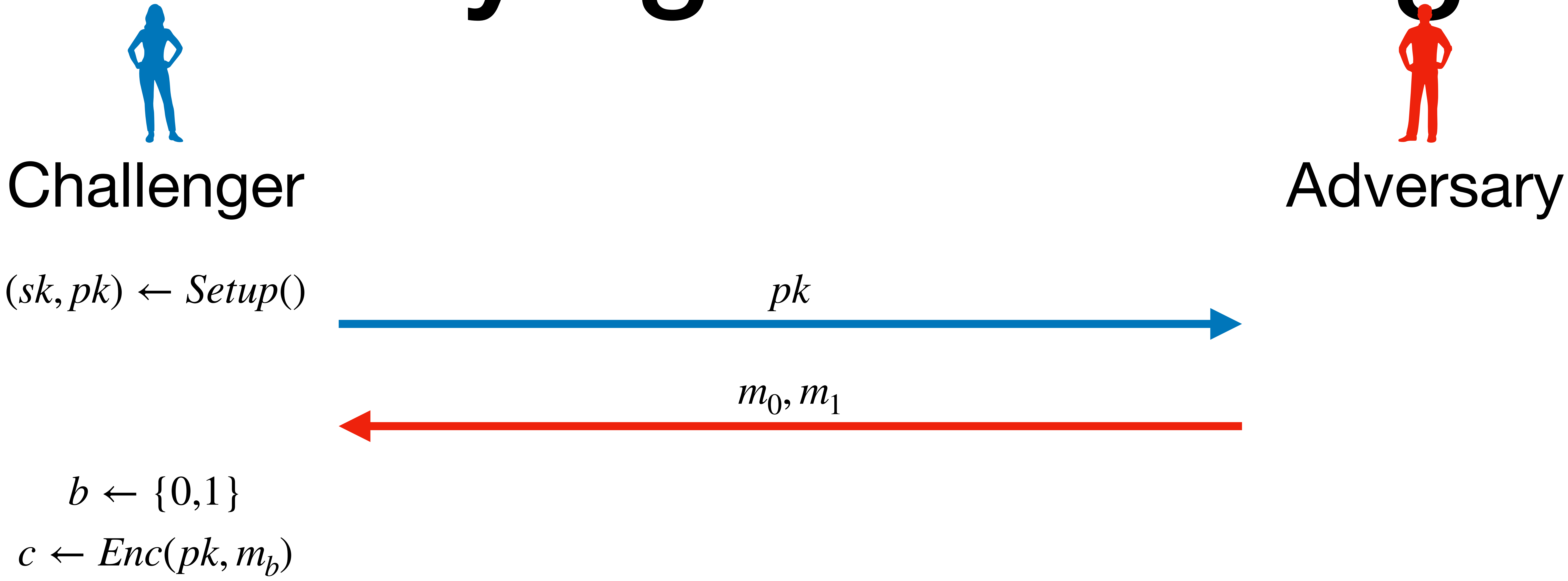
Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

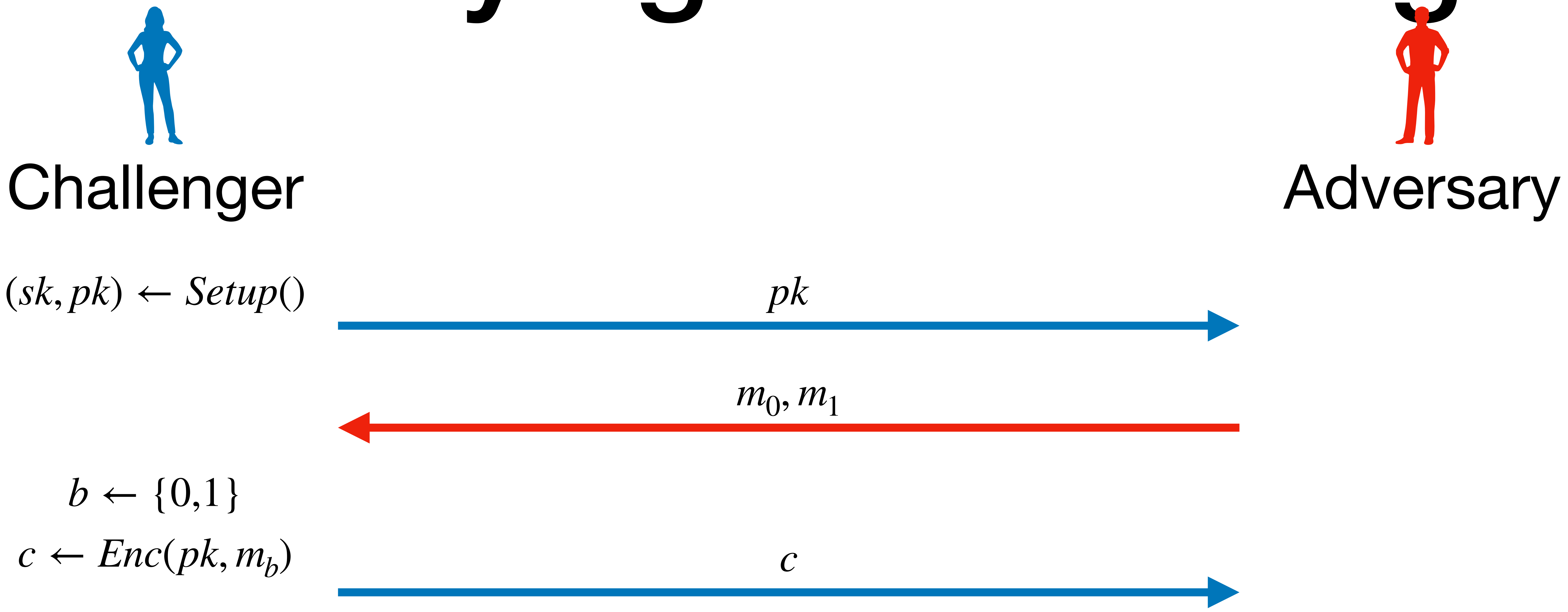
pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

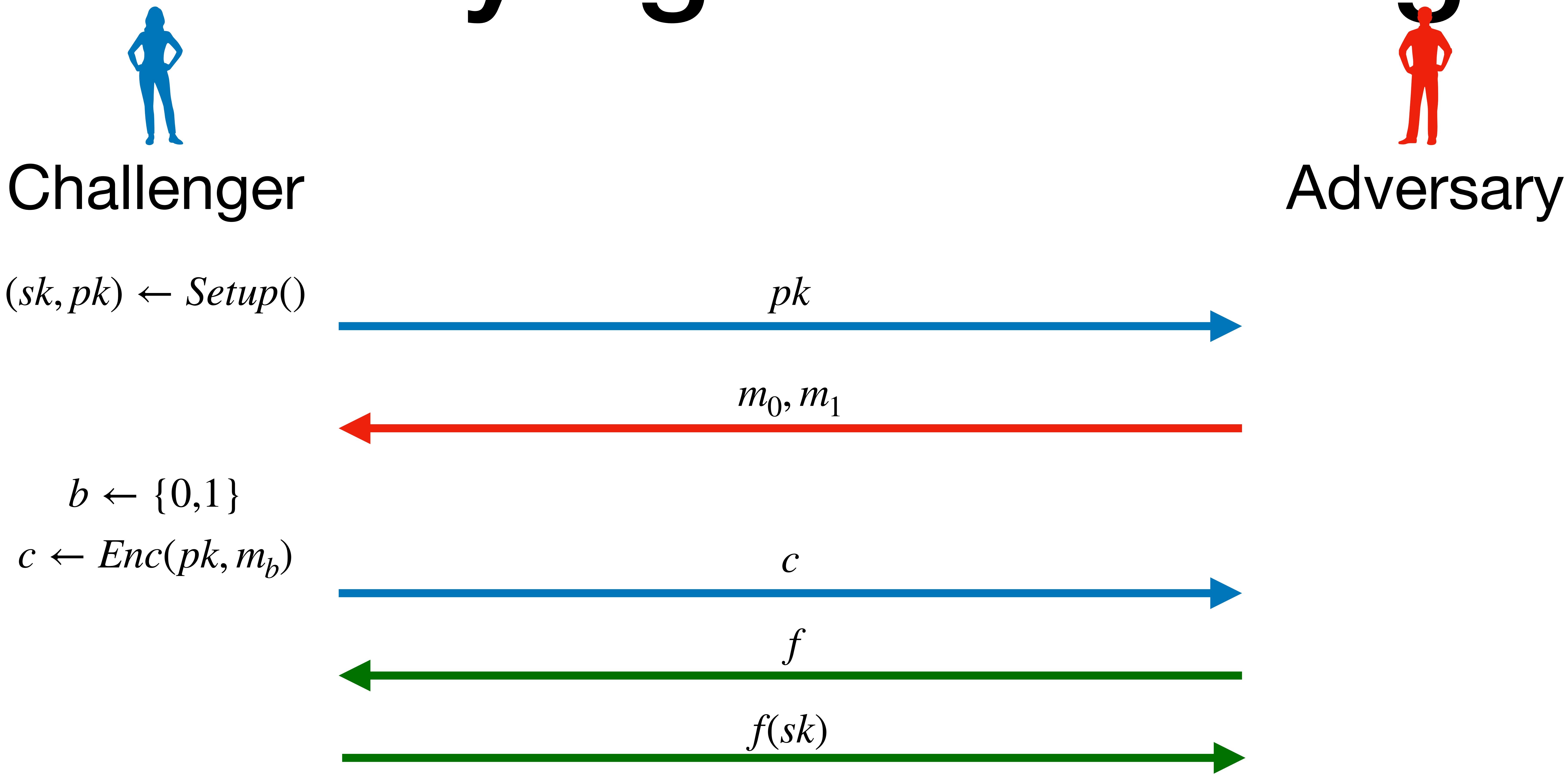
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

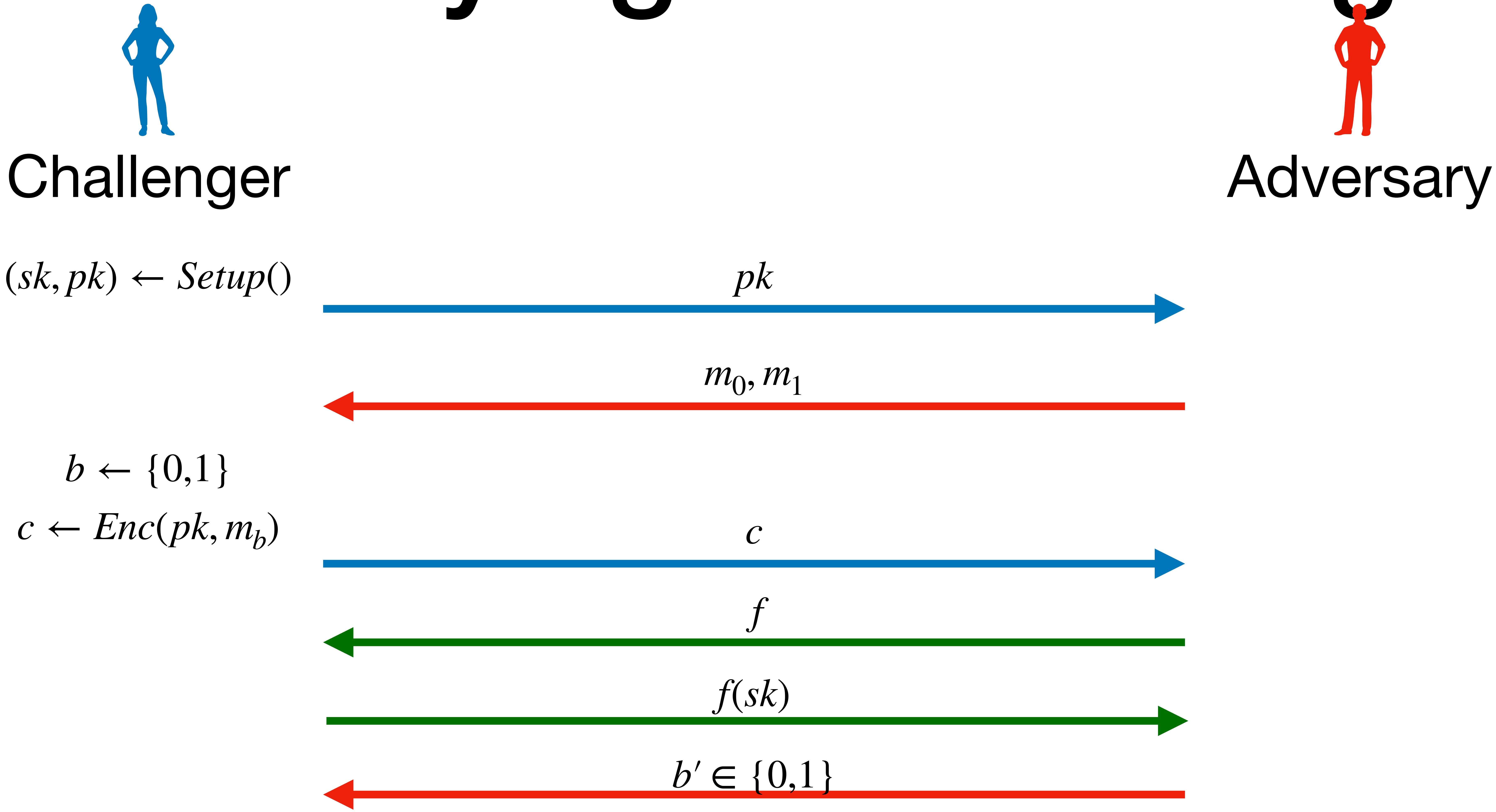
$c \leftarrow Enc(pk, m_b)$

c

f

$f(sk)$

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

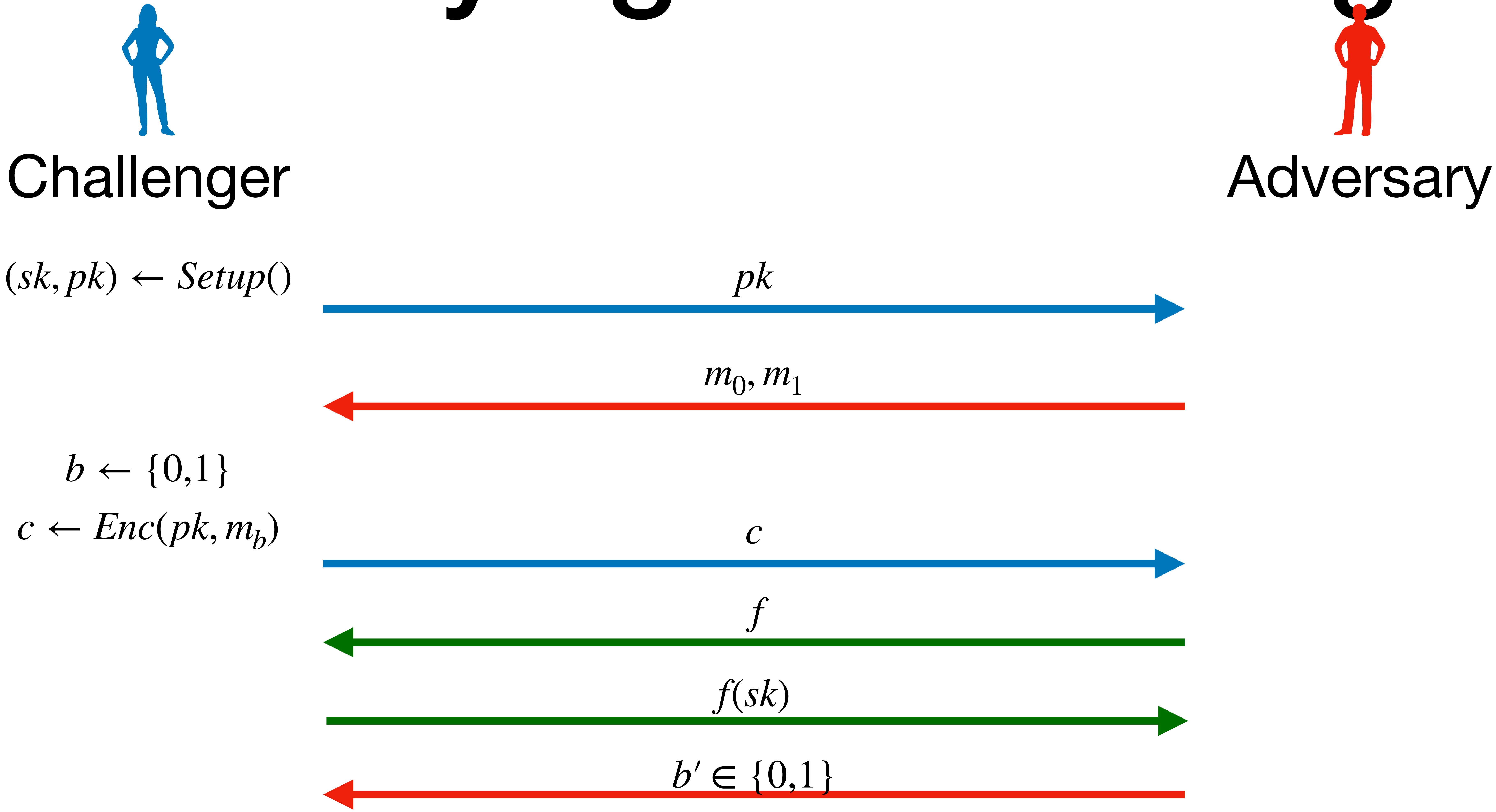
c

f

$f(sk)$

$b' \in \{0,1\}$

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

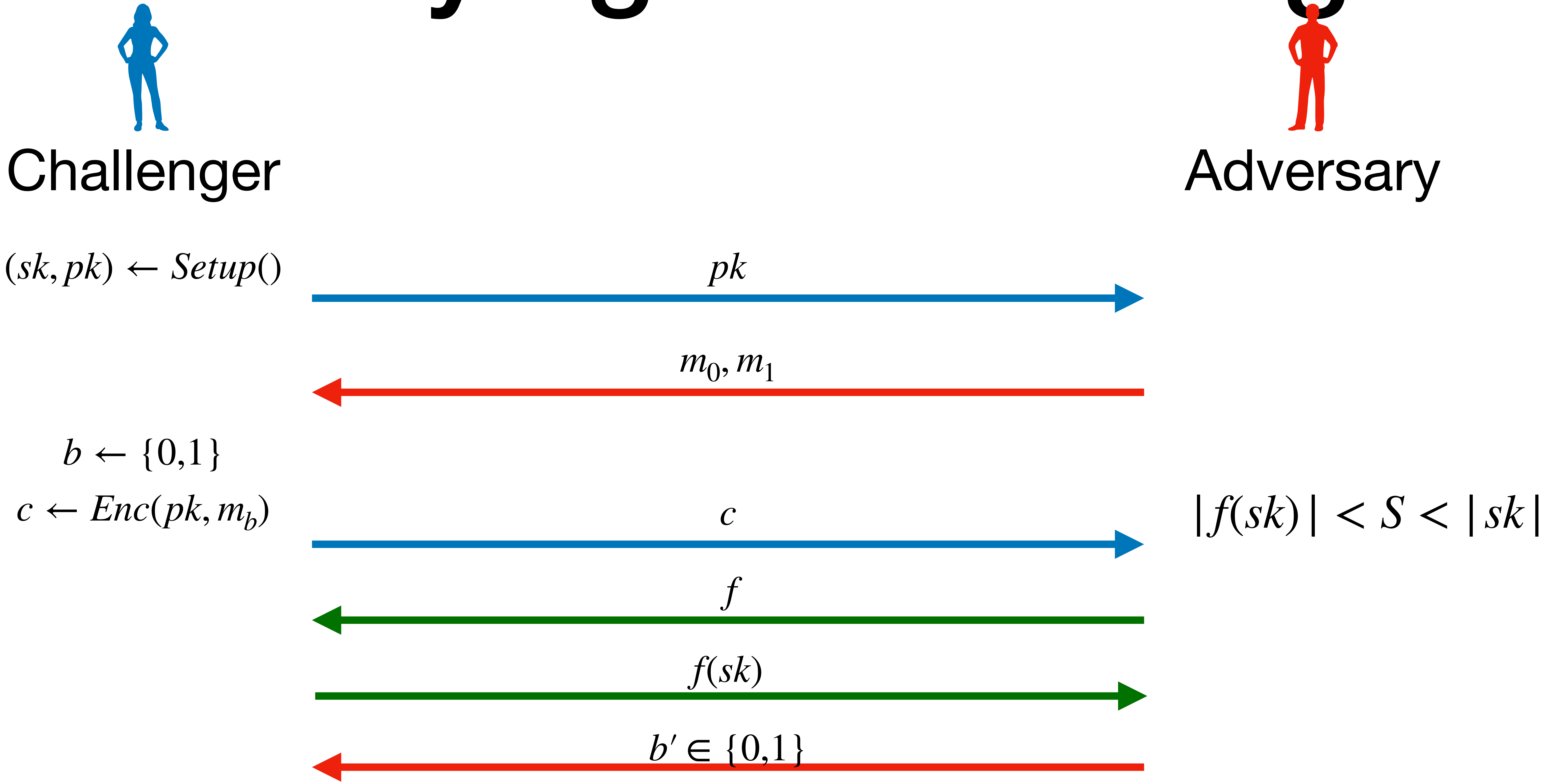
f

$f(sk)$

$b' \in \{0,1\}$

Adversary wins if $b = b'$

Security against Leakage



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk →

← m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c →

$|f(sk)| < S < |sk|$

← f

$f(sk)$ →

← $b' \in \{0,1\}$

Adversary wins if $b = b'$

Leakage Resilient Schemes

Leakage Resilient Schemes

- [Canetti et al. 00] and [Dodis et al. 01] gave construction where f returns bits of sk.

Leakage Resilient Schemes

- [Canetti et al. 00] and [Dodis et al. 01] gave construction where f returns bits of sk.
- [Dziembowski06], [Di Crescenzo et al.06], [Akavia et al.09], etc. considered arbitrary function f.

Leakage Resilient Schemes

- [Canetti et al. 00] and [Dodis et al. 01] gave construction where f returns bits of sk.

- [Dziembowski06], [Di Crescenzo et al.06], [Akavia et al.09], etc. considered arbitrary function f.

- Other works include [Dodis et al.09], [Brakerski et al.10], [Dodis et al.10], [Faonio et al.15] and many more.

Can the entire secret key be exposed?

Can the entire secret key be exposed?

- Does not make sense if entire secret key and ciphertext is given to adversary.

Can the entire secret key be exposed?

- Does not make sense if entire secret key and ciphertext is given to adversary.

- May be possible for adversary to attain the entire secret key but store only a part of the ciphertext. For example, cloud storage.

Incompressible (PKE) Security

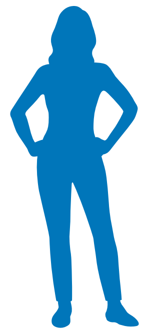
Incompressible (PKE) Security

Incompressible (PKE) Security

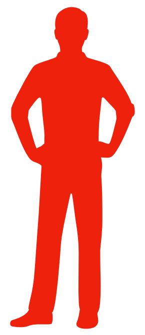
Challenger

Adversary 1

Incompressible (PKE) Security



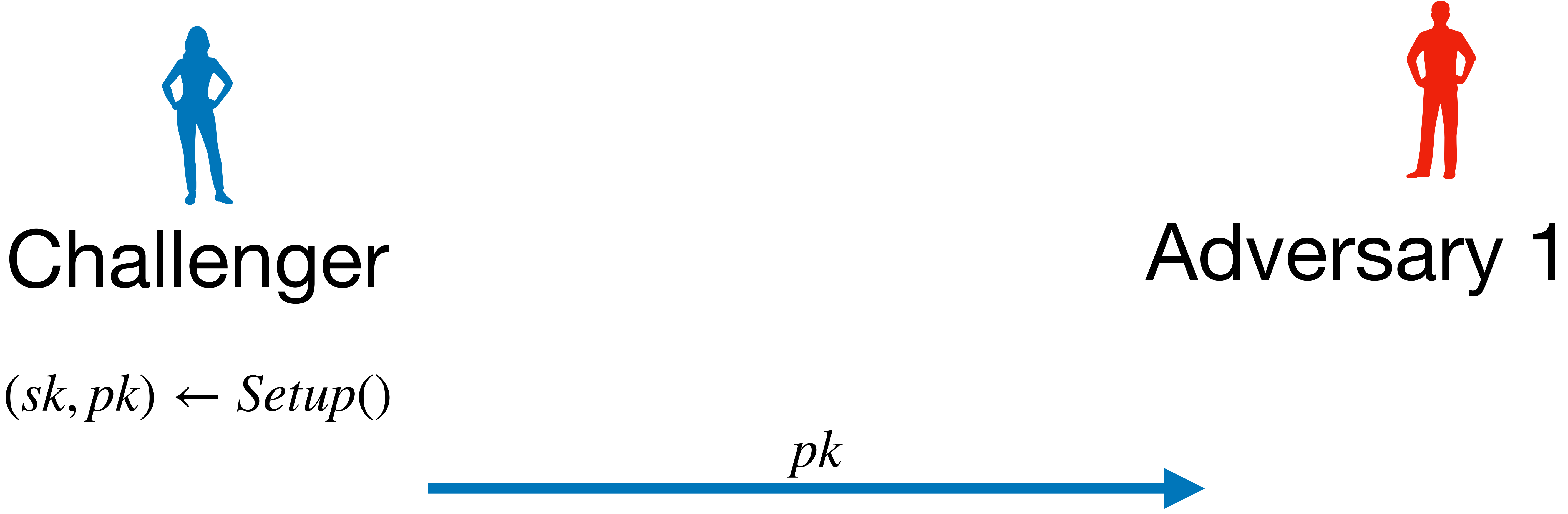
Challenger



Adversary 1

$(sk, pk) \leftarrow Setup()$

Incompressible (PKE) Security



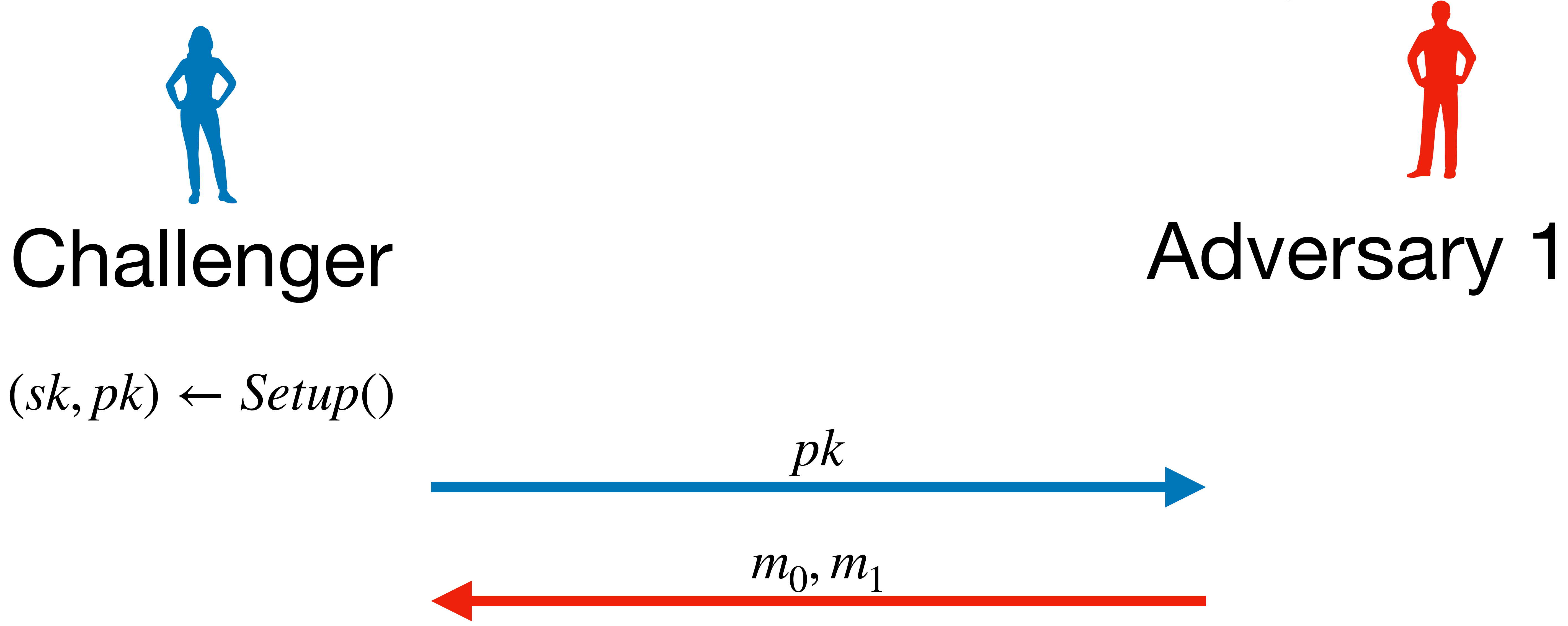
Challenger

Adversary 1

$(sk, pk) \leftarrow Setup()$

$$\xrightarrow{pk}$$

Incompressible (PKE) Security



Challenger

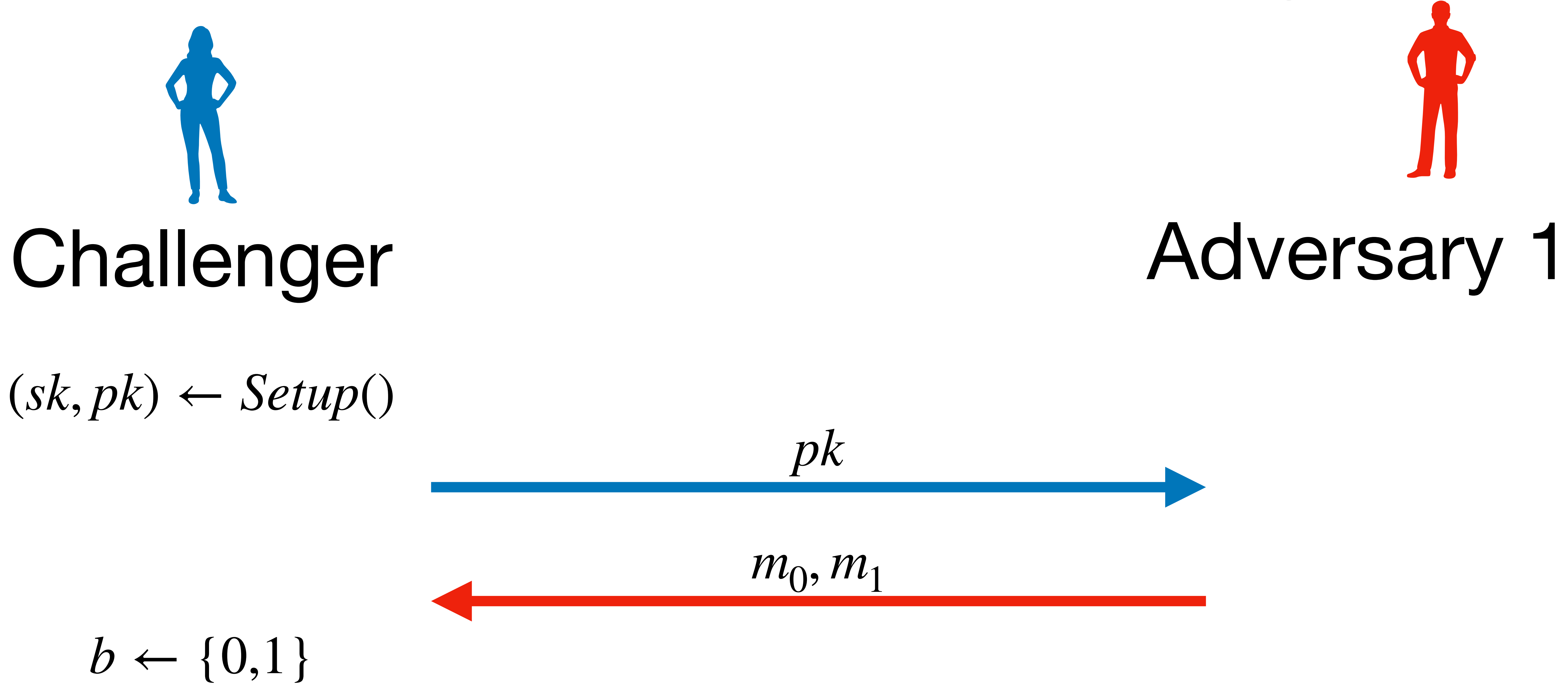
Adversary 1

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

Incompressible (PKE) Security



Challenger

Adversary 1

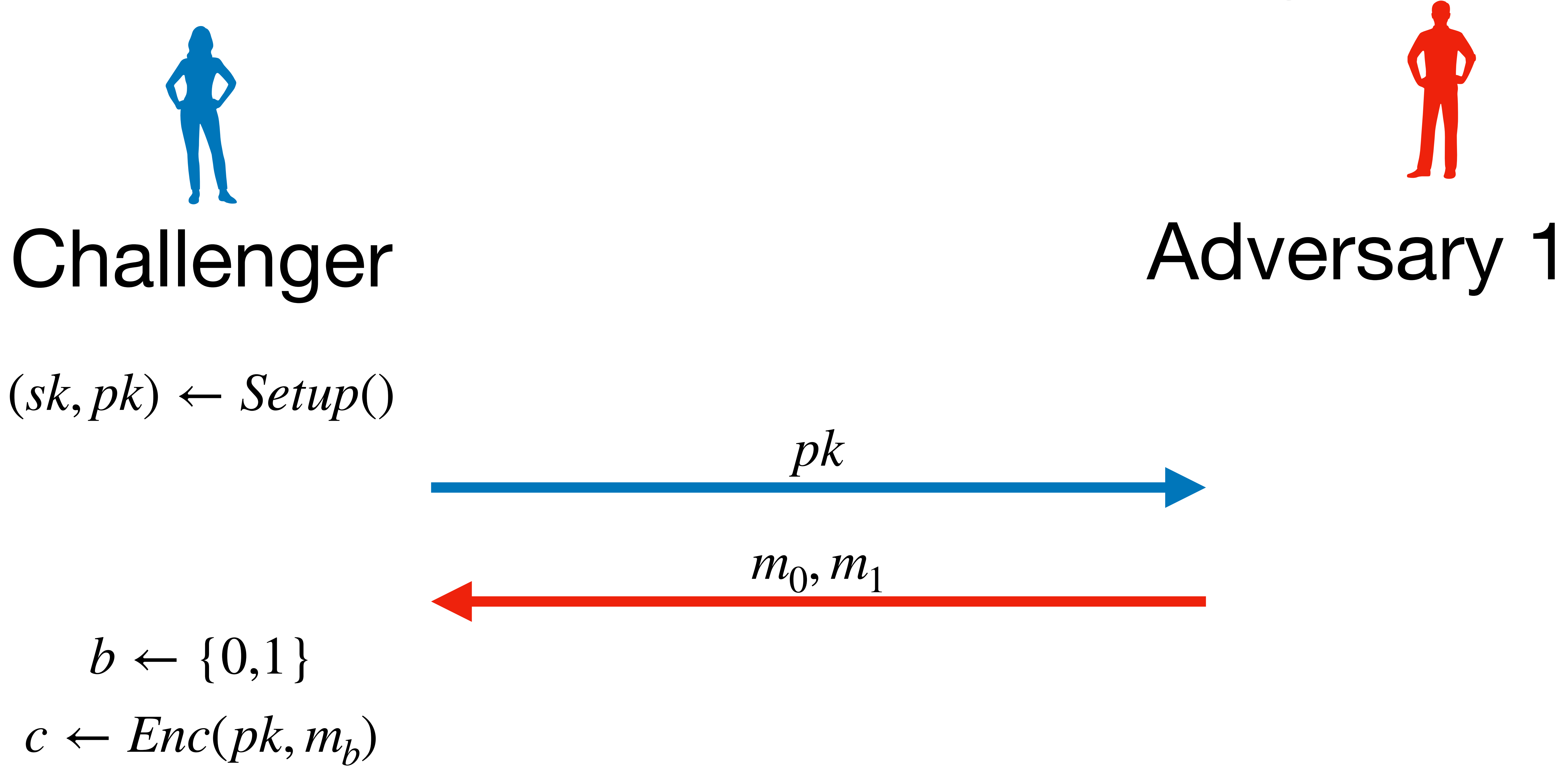
$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

Incompressible (PKE) Security



Challenger

Adversary 1

$(sk, pk) \leftarrow Setup()$

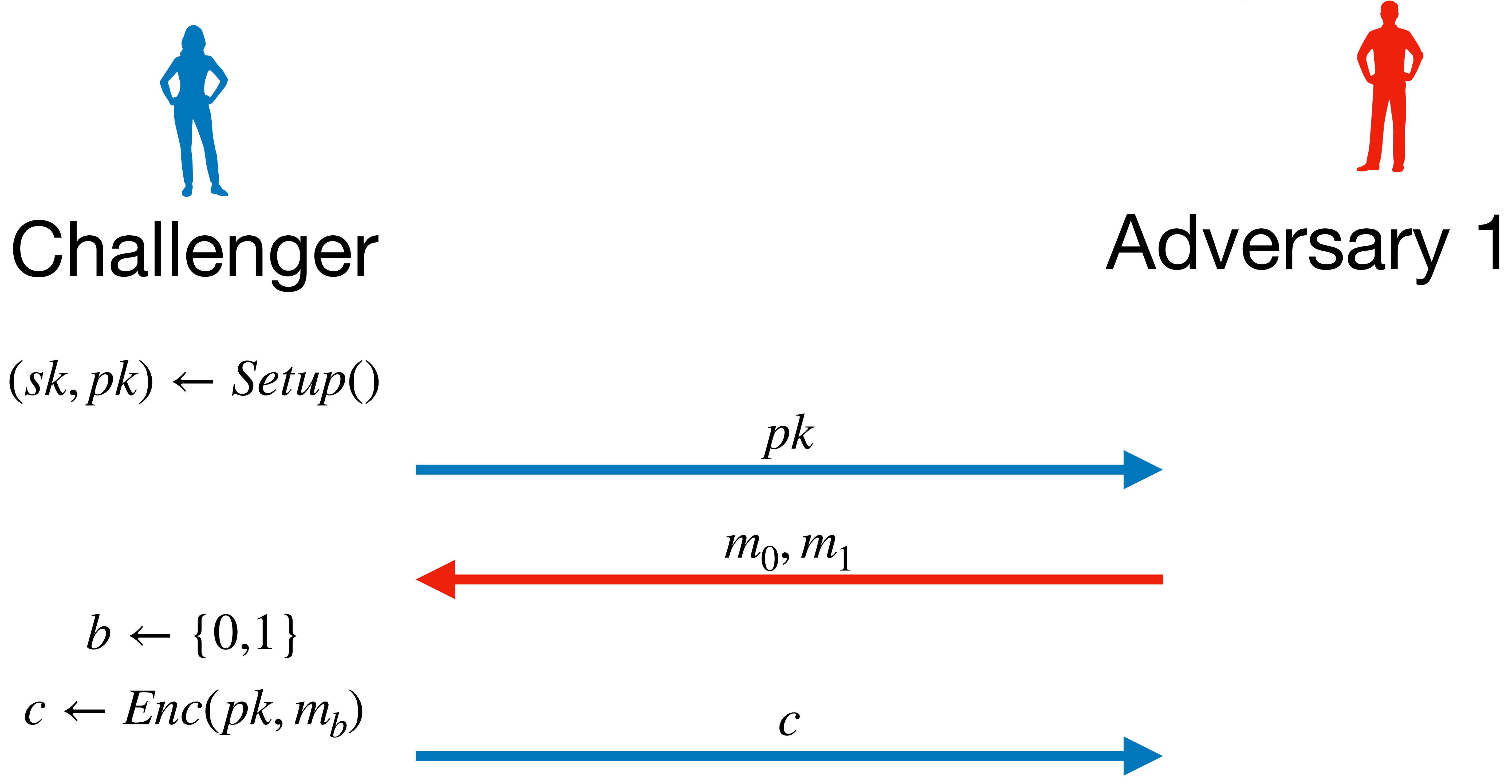
pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

Incompressible (PKE) Security



Challenger

Adversary 1

$(sk, pk) \leftarrow Setup()$

pk $\longrightarrow$

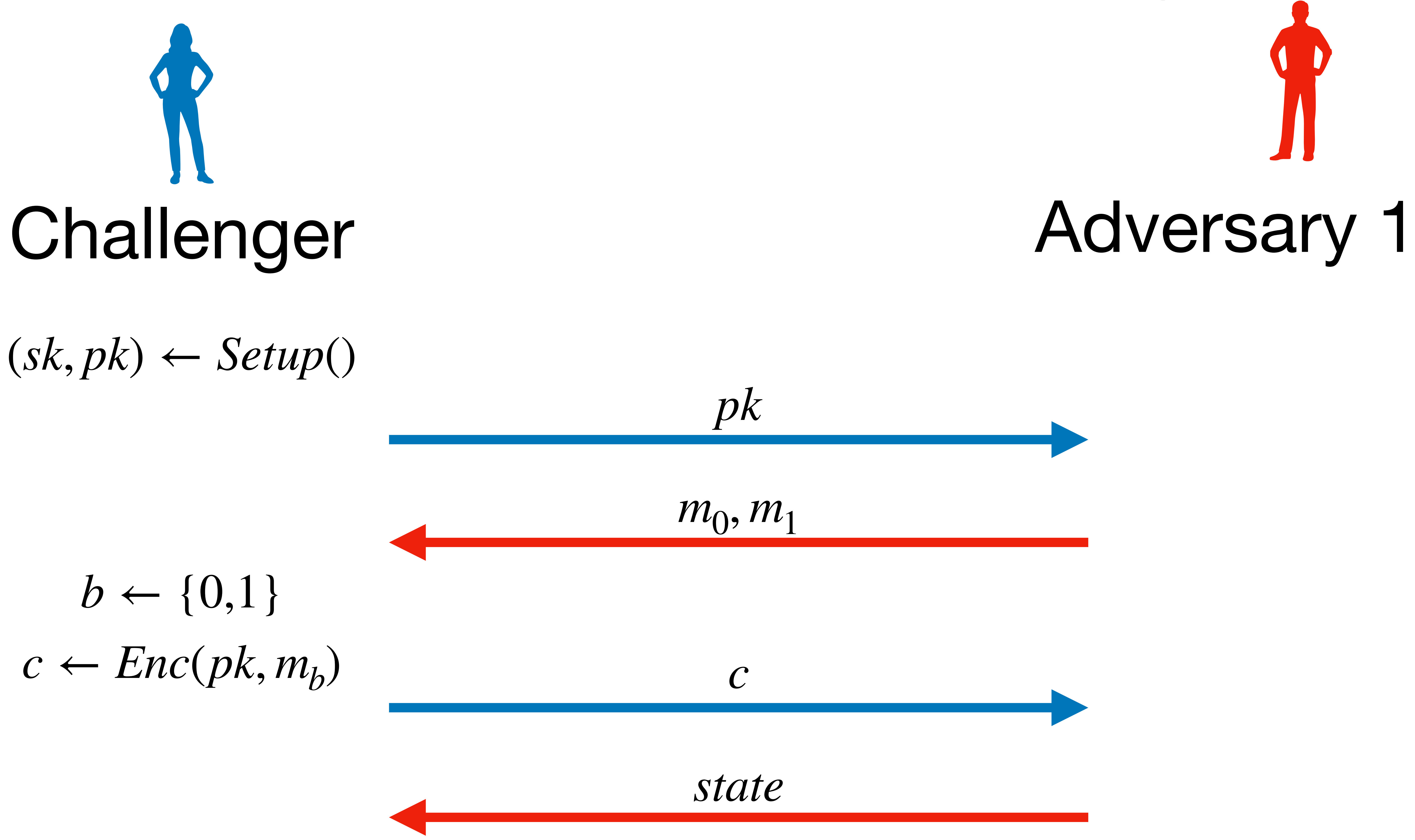
m_0, m_1 $\longleftarrow$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c $\longrightarrow$

Incompressible (PKE) Security



Challenger

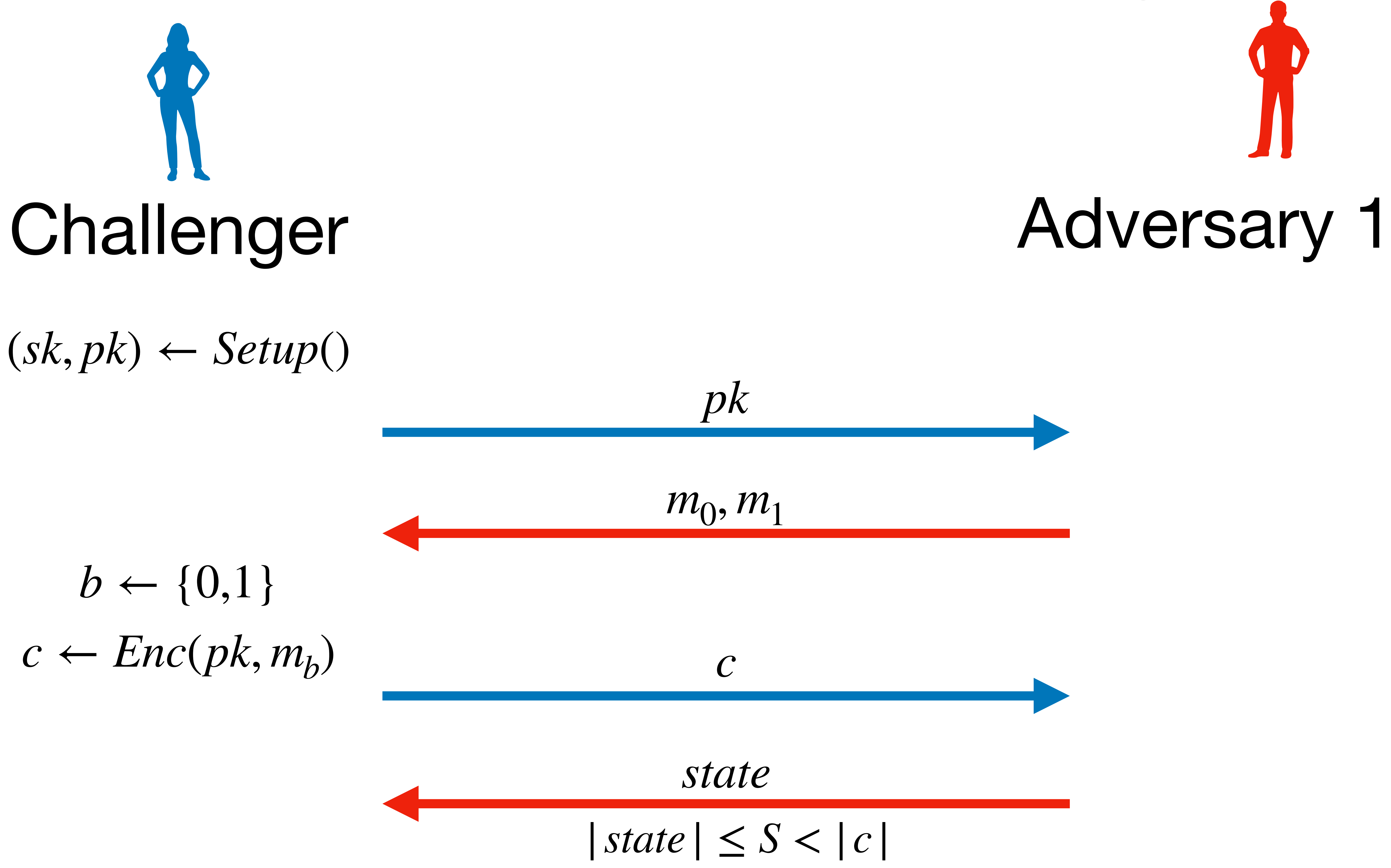
Adversary 1

$(sk, pk) \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

Incompressible (PKE) Security



Challenger

Adversary 1

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

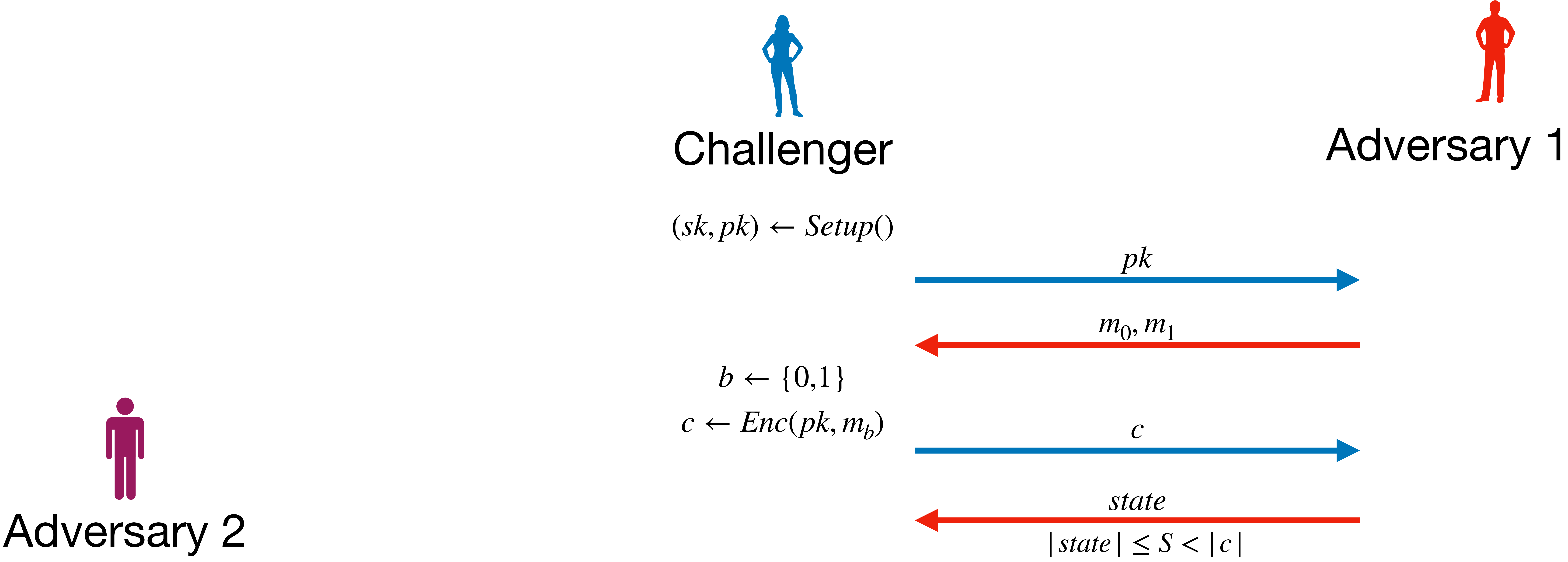
$c \leftarrow Enc(pk, m_b)$

c

$state$

$|state| \leq S < |c|$

Incompressible (PKE) Security



Challenger

Adversary 1

Adversary 2

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

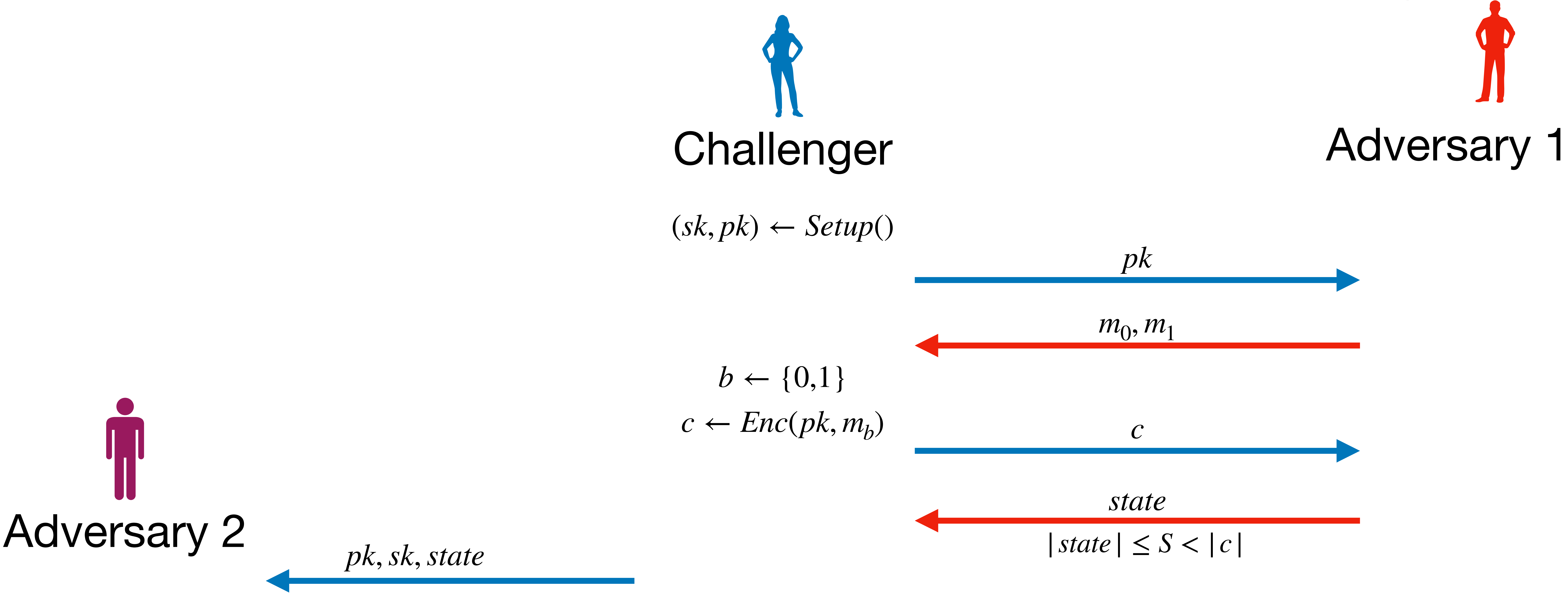
$c \leftarrow Enc(pk, m_b)$

c

$state$

$|state| \leq S < |c|$

Incompressible (PKE) Security



Challenger

Adversary 1

Adversary 2

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

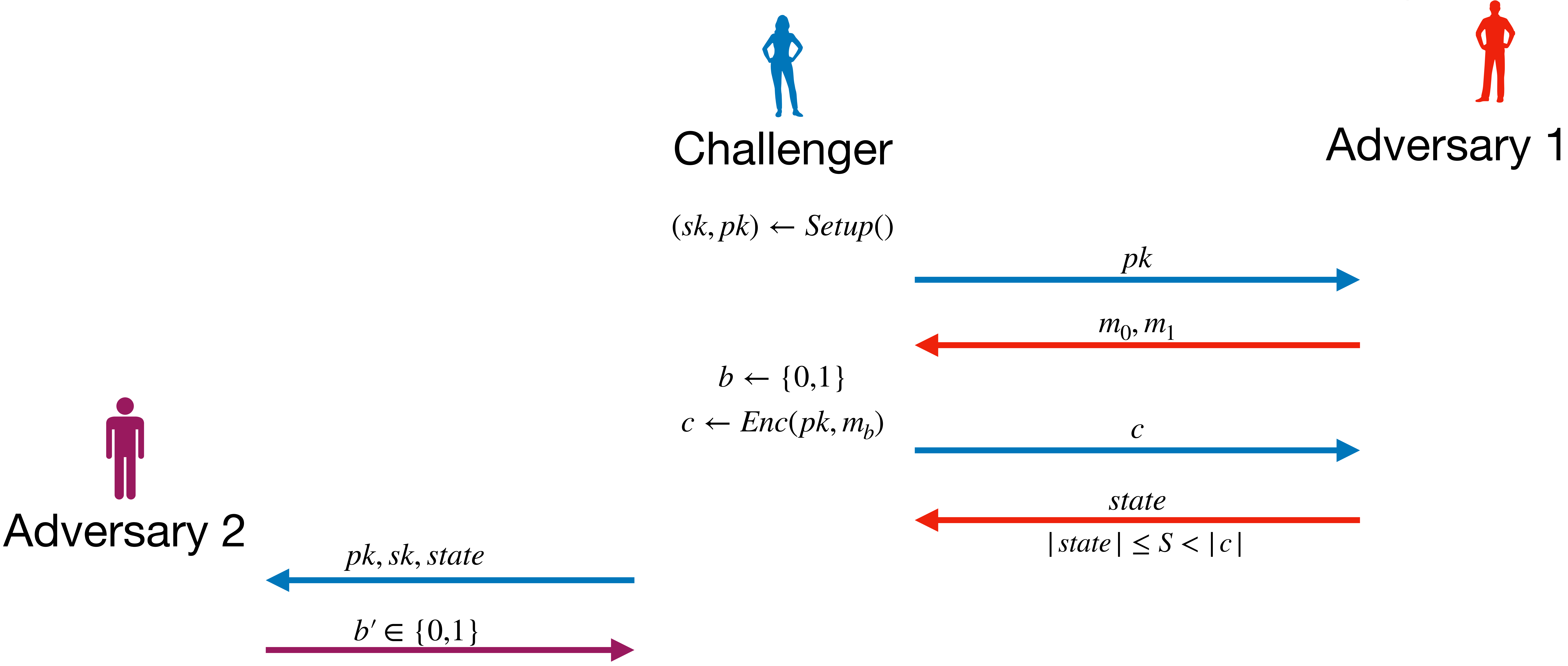
c

$state$

$|state| \leq S < |c|$

$pk, sk, state$

Incompressible (PKE) Security



Challenger

Adversary 1

Adversary 2

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

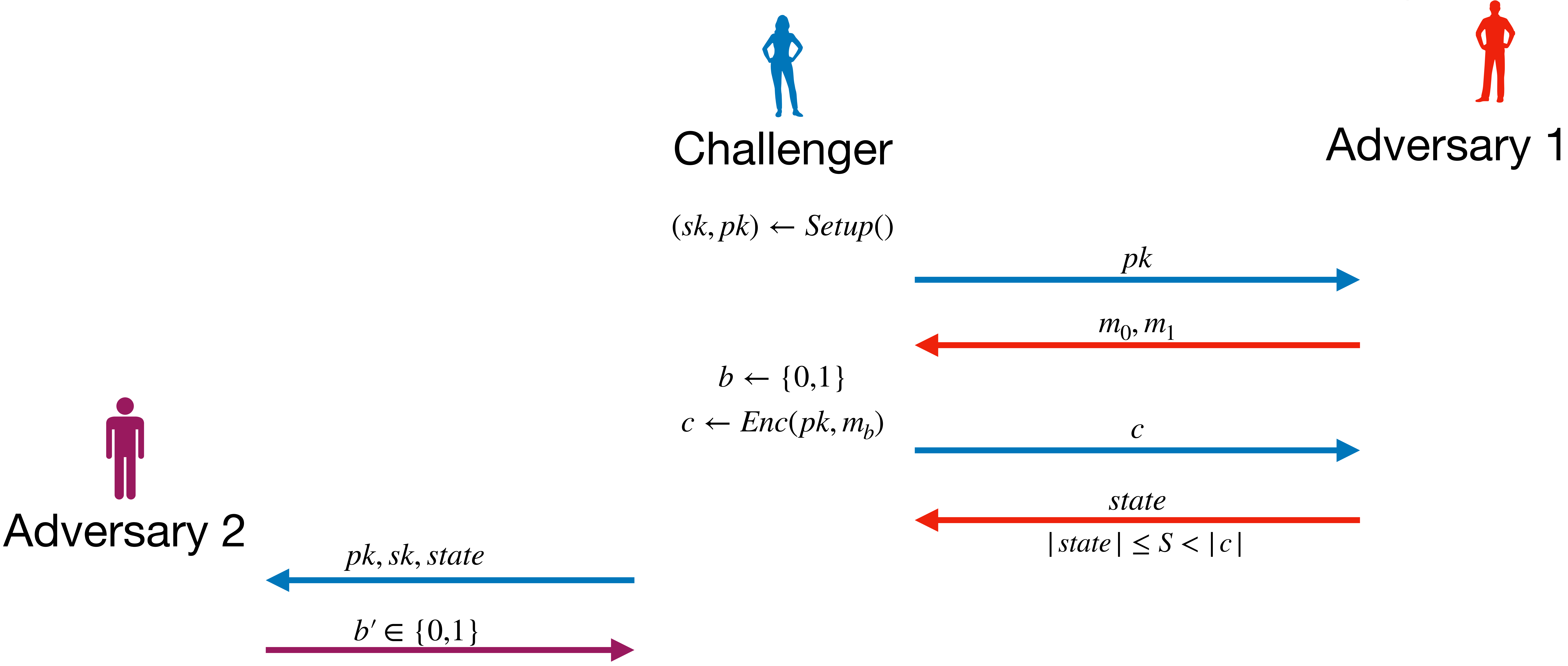
$state$

$|state| \leq S < |c|$

$pk, sk, state$

$b' \in \{0,1\}$

Incompressible (PKE) Security



Challenger

Adversary 1

Adversary 2

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

$state$

$|state| \leq S < |c|$

$pk, sk, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

Incompressible SKE Schemes

Incompressible SKE Schemes

- Dziembowski gave the first construction under standard assumptions (bad rate)

Incompressible SKE Schemes

- Dziembowski gave the first construction under standard assumptions (bad rate)
- Guan et al. gave a rate-1 construction based on LWE and DCR (using incompressible encoding)

Incompressible PKE Schemes

Incompressible PKE Schemes

- Guan et al. gave a construction (bad rate) based on PKE. They also gave a rate 1 construction using Indistinguishable Obfuscator (iO).

Incompressible PKE Schemes

- Guan et al. gave a construction (bad rate) based on PKE. They also gave a rate 1 construction using Indistinguishable Obfuscator (iO).

- Branco et al. gave a construction based on hardness of LWE and DDH which is rate 1.

Incompressible PKE Schemes

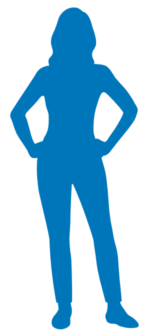
- Guan et al. gave a construction (bad rate) based on PKE. They also gave a rate 1 construction using Indistinguishable Obfuscator (iO).
- Branco et al. gave a construction based on hardness of LWE and DDH which is rate 1.
- Our Result : a generic transformation from PKE to incompressible PKE. This also works for more advanced notions of encryption.

Incompressible SKE & PKE

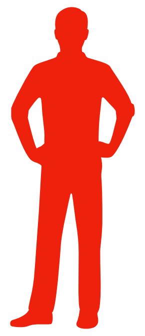
Incompressible (SKE) Security

Incompressible (SKE) Security

Incompressible (SKE) Security

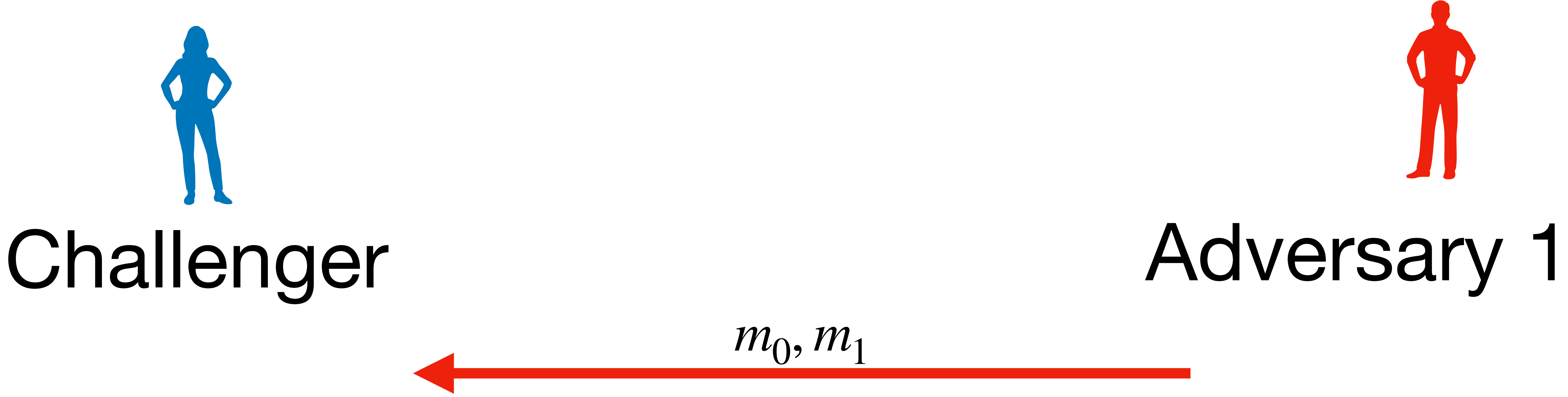


Challenger



Adversary 1

Incompressible (SKE) Security

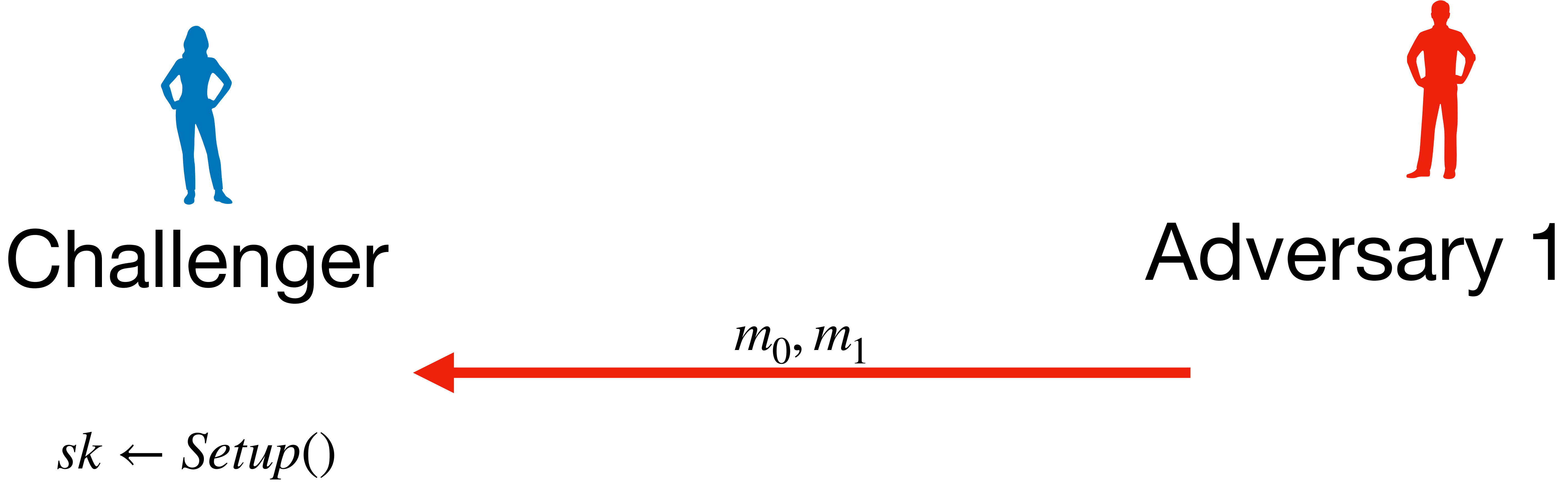


Challenger

Adversary 1

m_0, m_1

Incompressible (SKE) Security



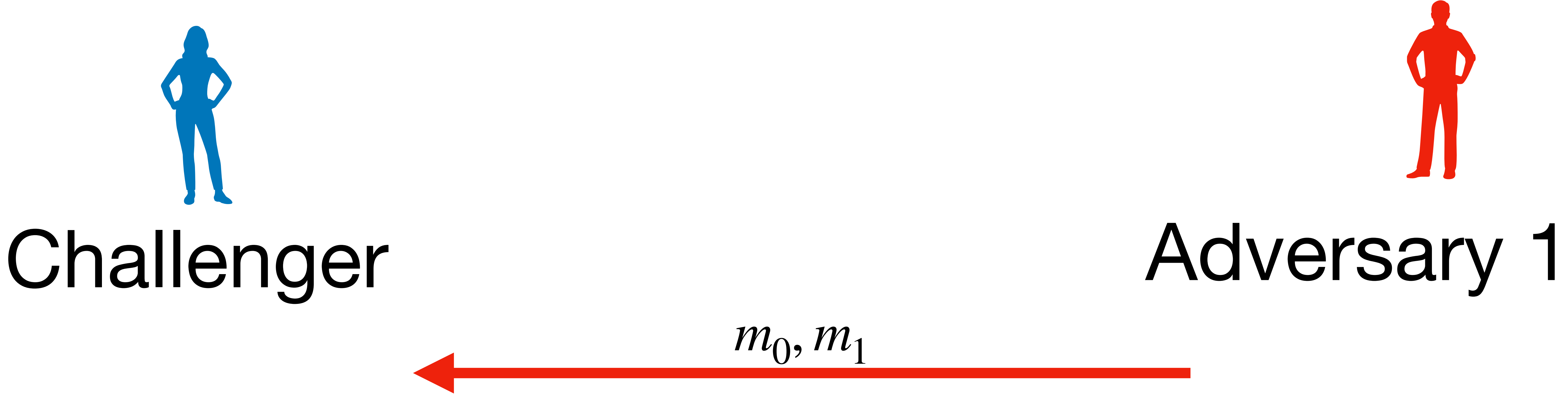
Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

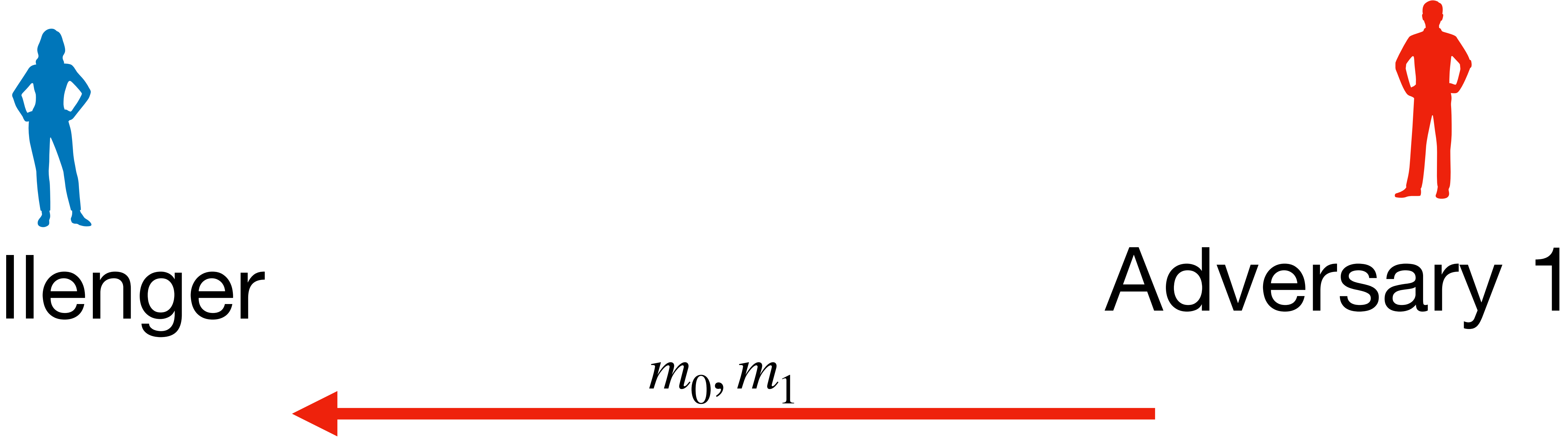
Incompressible (SKE) Security



$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

Incompressible (SKE) Security



Challenger

Adversary 1

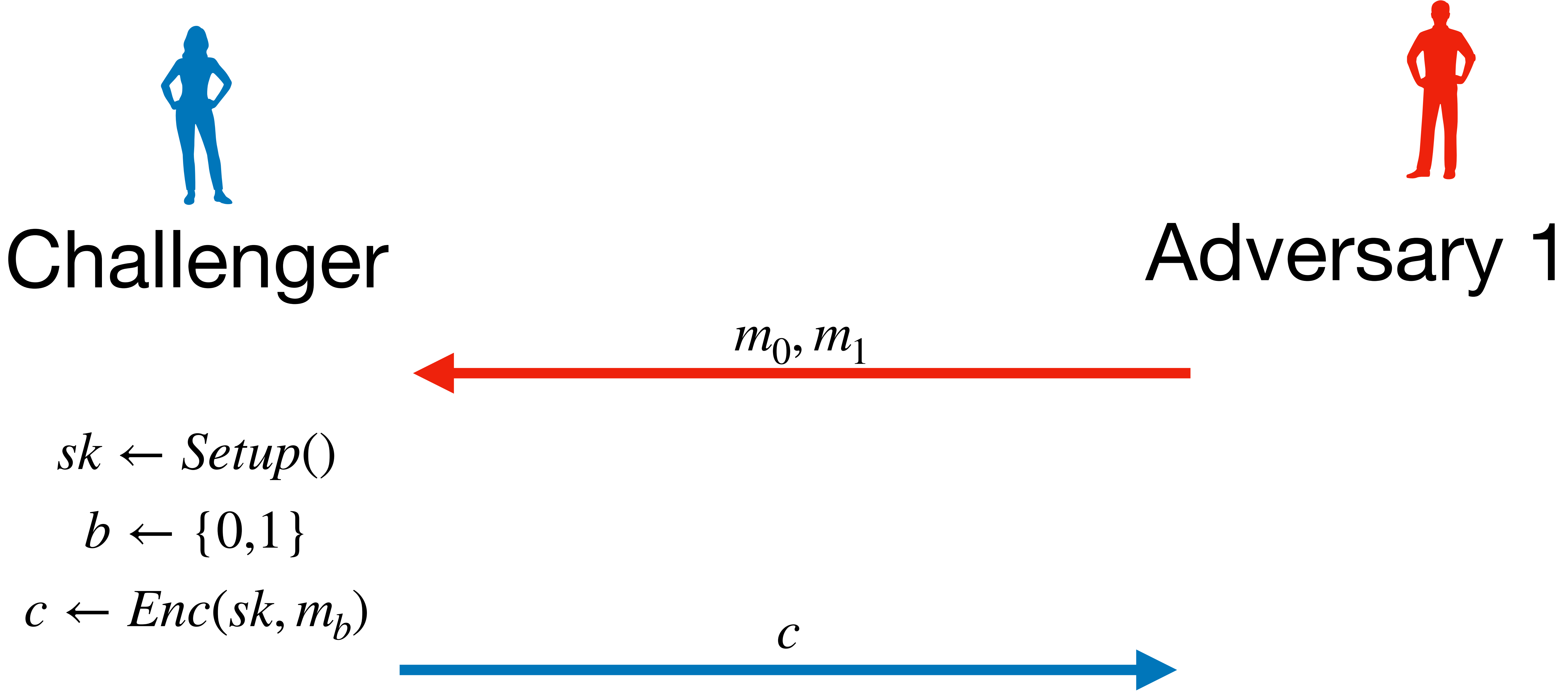
m_0, m_1

$$sk \leftarrow Setup()$$

$$b \leftarrow \{0,1\}$$

$$c \leftarrow Enc(sk, m_b)$$

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

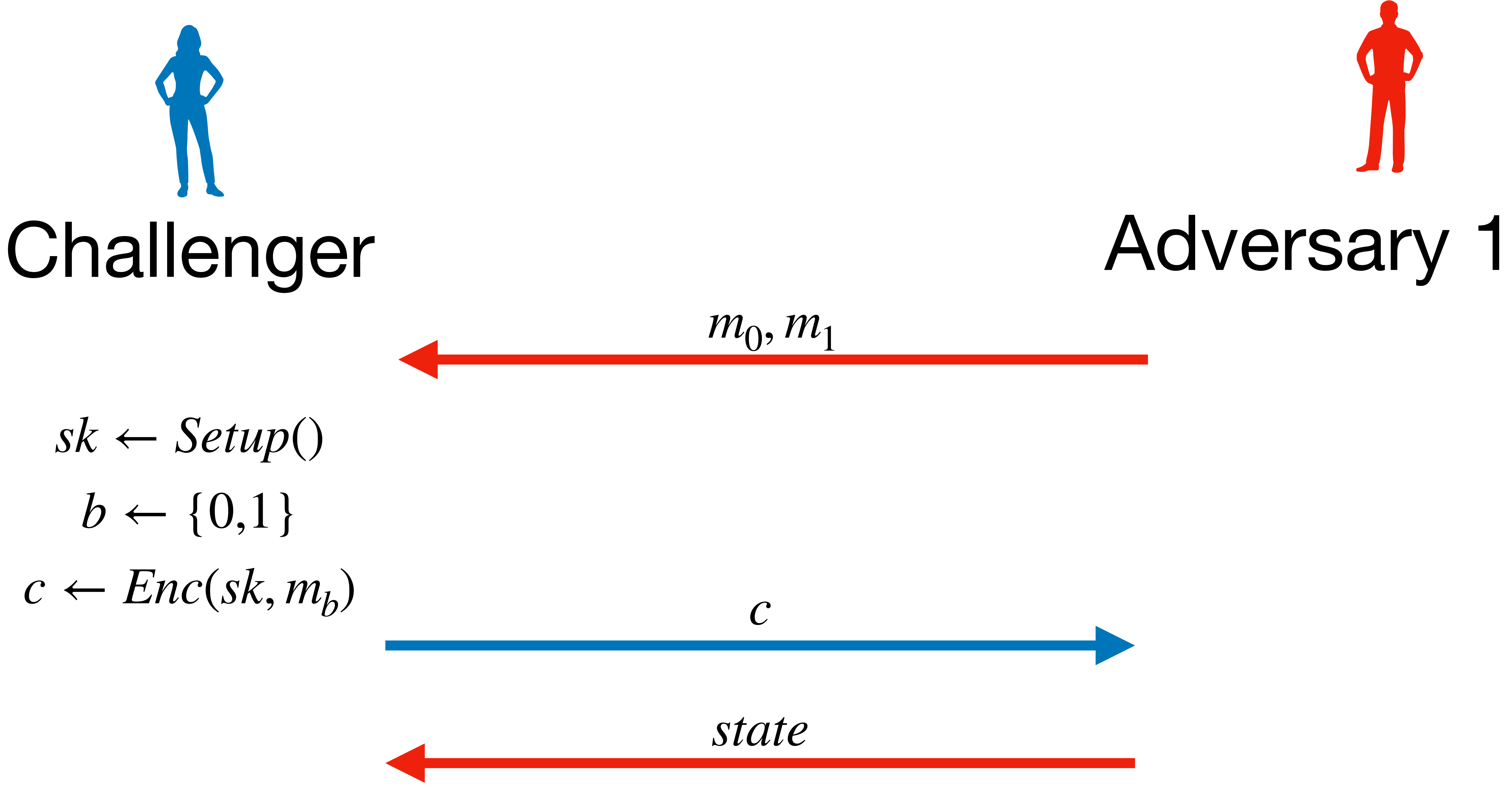
$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

c

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

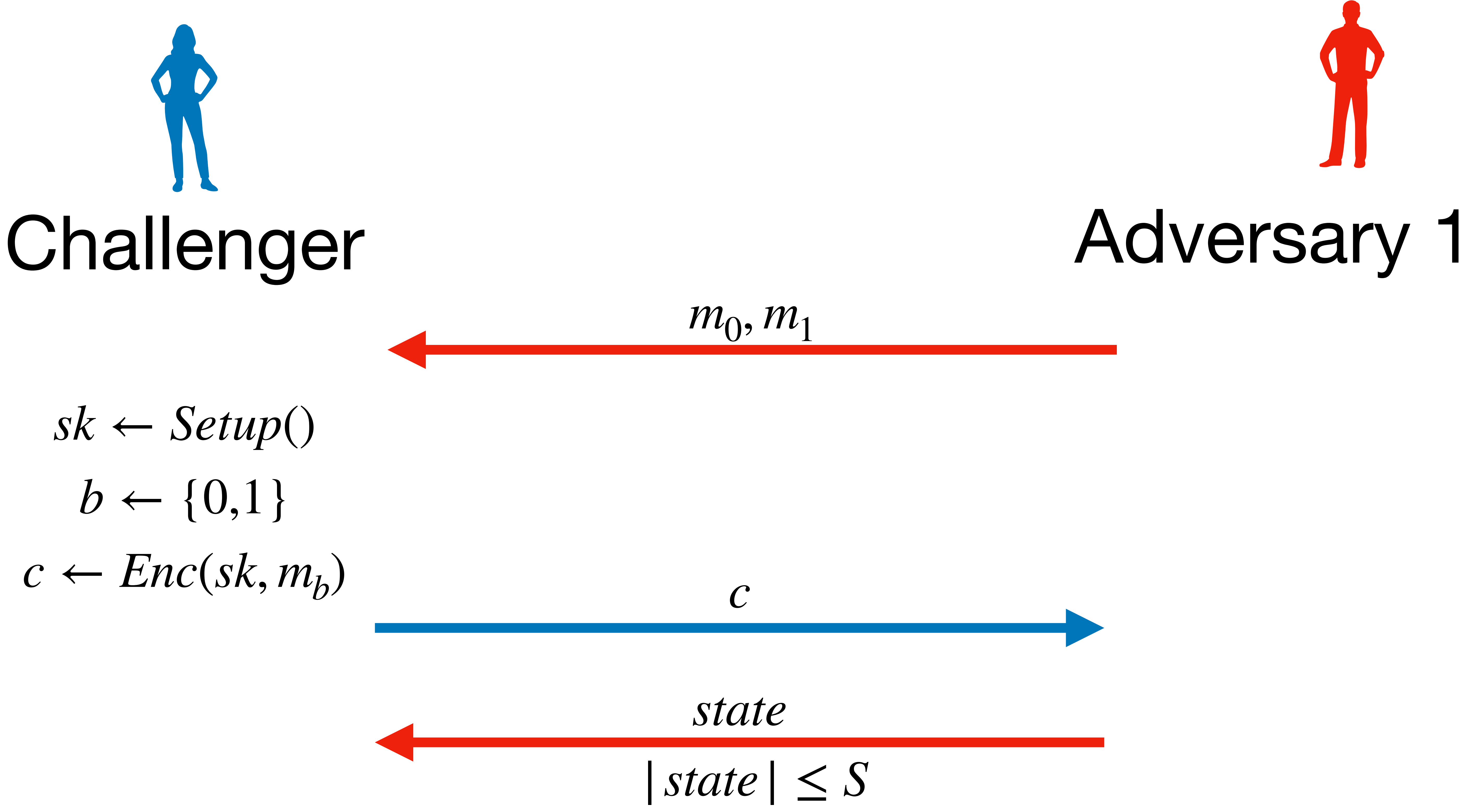
$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

c

$state$

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

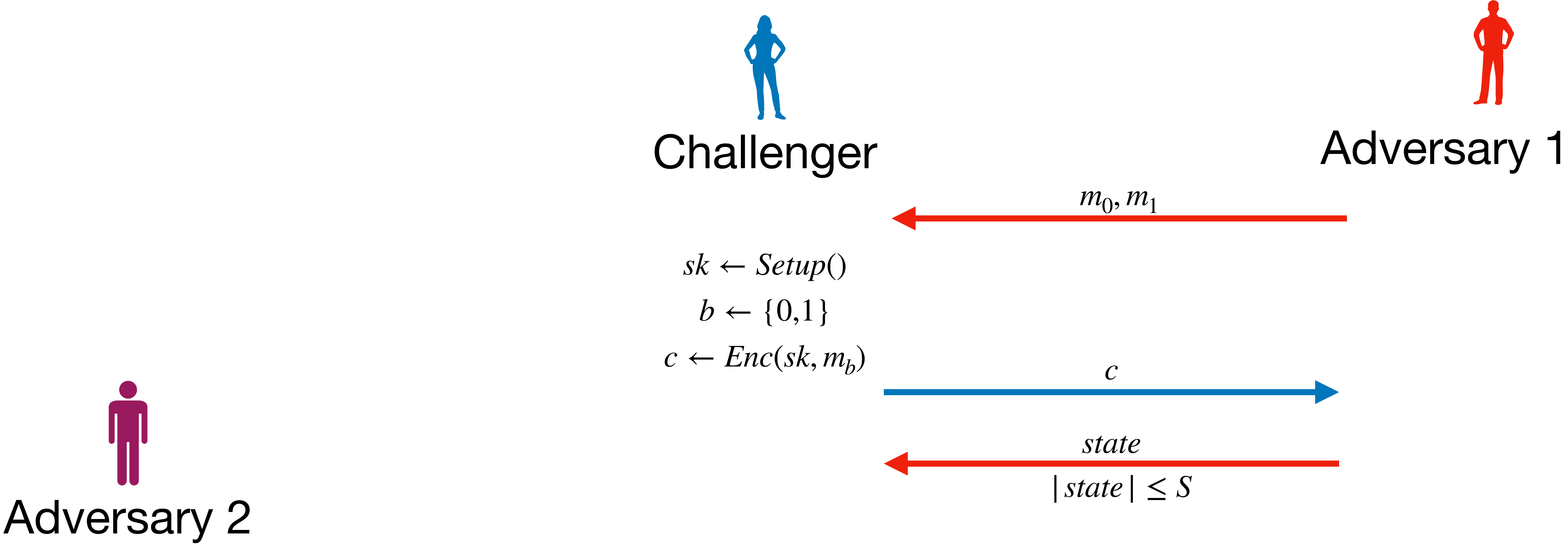
$c \leftarrow Enc(sk, m_b)$

c

$state$

$|state| \leq S$

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

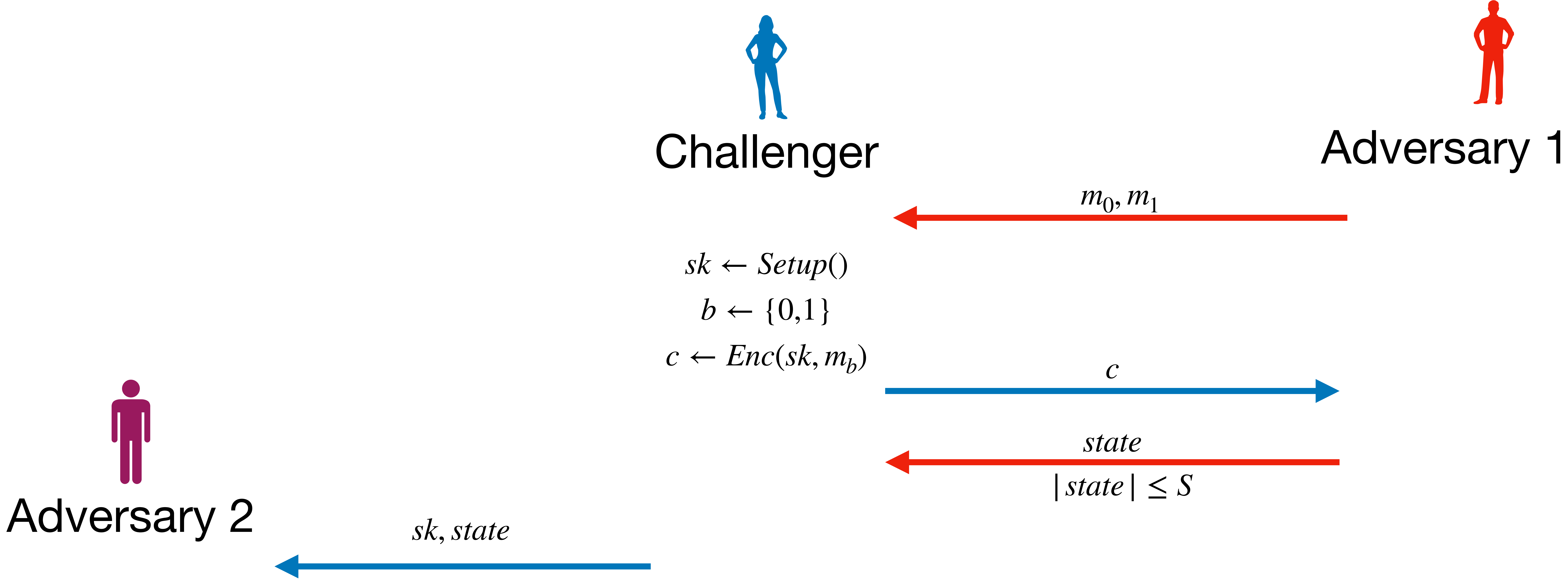
c

$state$

$|state| \leq S$

Adversary 2

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

c

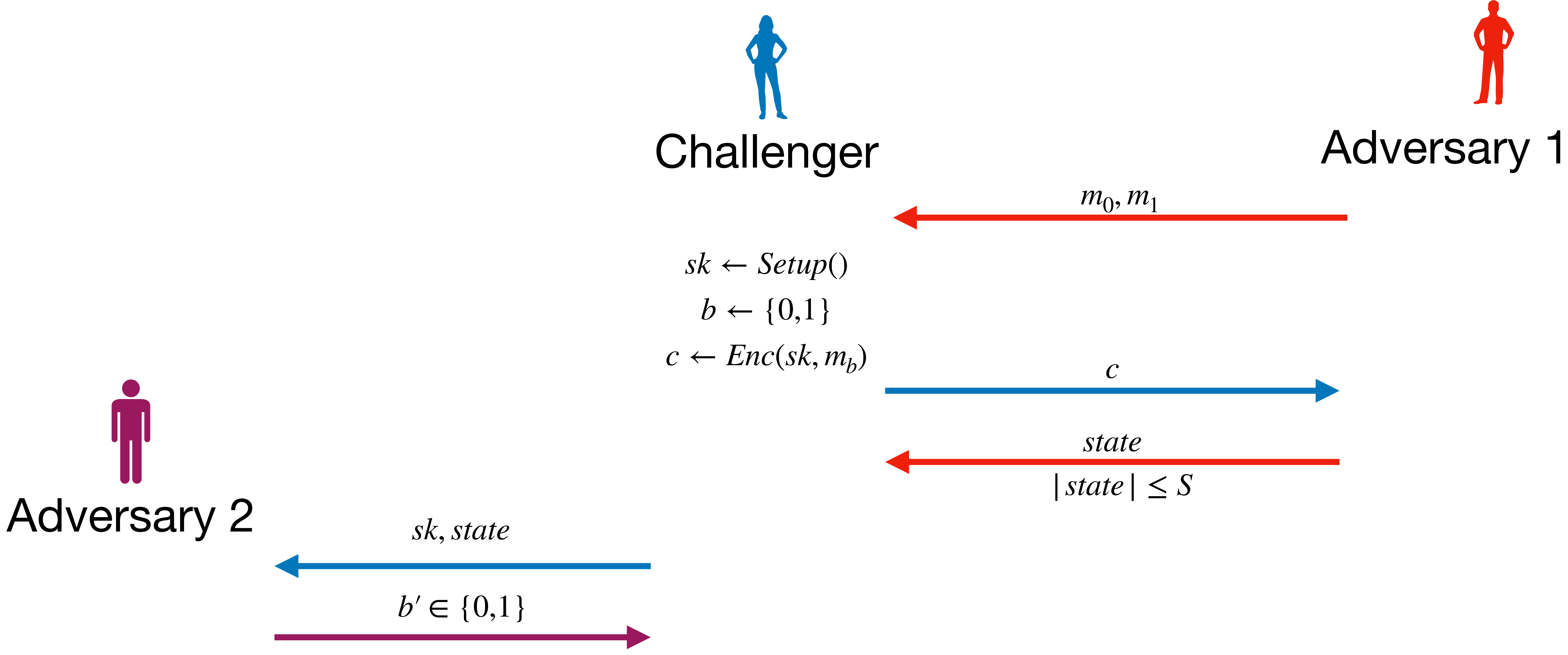
$state$

$|state| \leq S$

Adversary 2

$sk, state$

Incompressible (SKE) Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

c

$state$

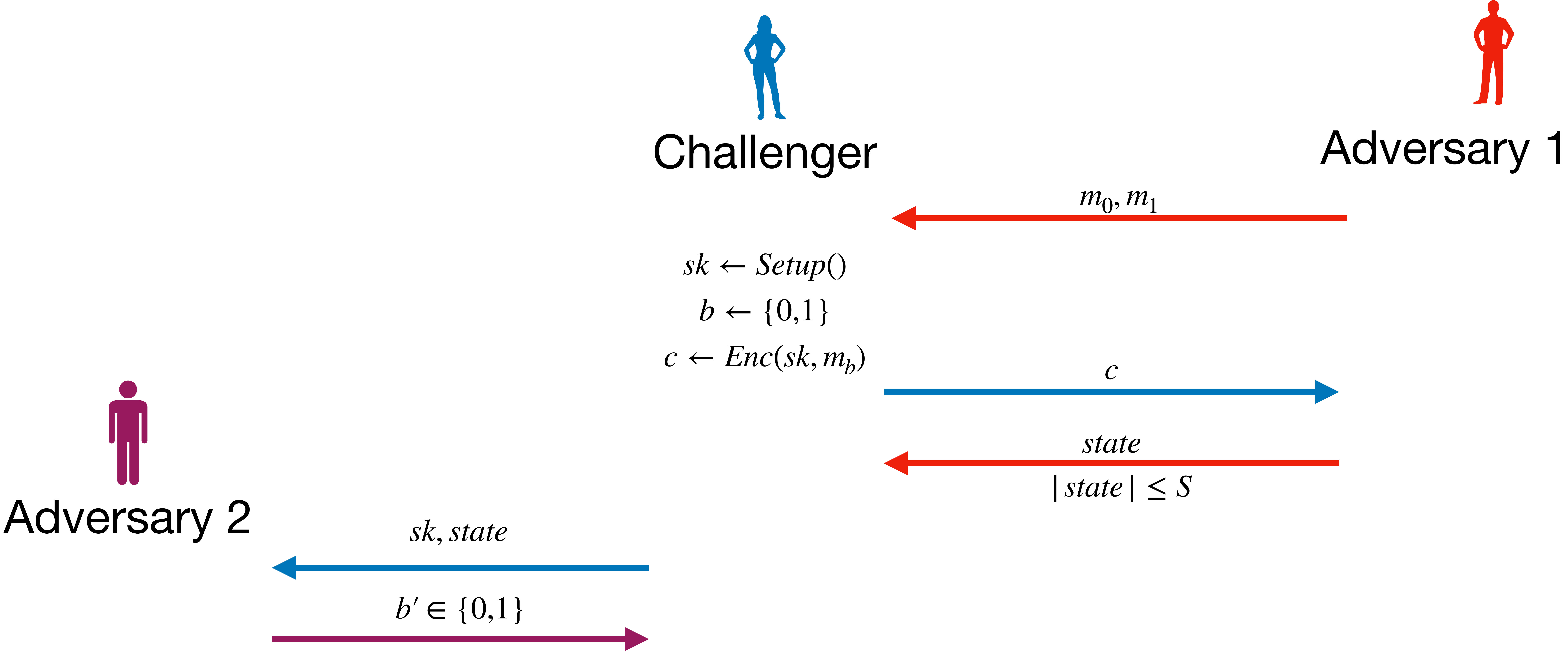
$|state| \leq S$

Adversary 2

$sk, state$

$b' \in \{0,1\}$

Incompressible (SKE) Security



Challenger

Adversary 1

Adversary 2

m_0, m_1

$sk \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(sk, m_b)$

c

$state$

$|state| \leq S$

$sk, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

One Time Pad is Compressible

One Time Pad is Compressible

- In OTP scheme, the key $sk \in \{0,1\}^n$ is a randomly generated string. To encrypt a message $m \in \{0,1\}^n$, compute $c = m \oplus sk$.

One Time Pad is Compressible

- In OTP scheme, the key $sk \in \{0,1\}^n$ is a randomly generated string. To encrypt a message $m \in \{0,1\}^n$, compute $c = m \oplus sk$.

- Consider $m_0 = 0^n$ and $m_1 = 1^n$. After receiving c, the first adversary creates $state = c[0]$.

One Time Pad is Compressible

- In OTP scheme, the key $sk \in \{0,1\}^n$ is a randomly generated string. To encrypt a message $m \in \{0,1\}^n$, compute $c = m \oplus sk$.

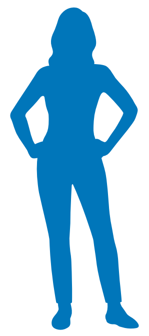
- Consider $m_0 = 0^n$ and $m_1 = 1^n$. After receiving c, the first adversary creates $state = c[0]$.

- Only receiving sk, the second adversary returns $b' = state[0] \oplus sk[0]$.

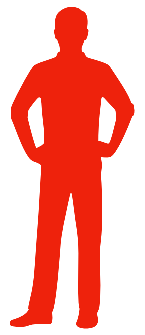
Incompressible Security

Incompressible Security

Incompressible Security

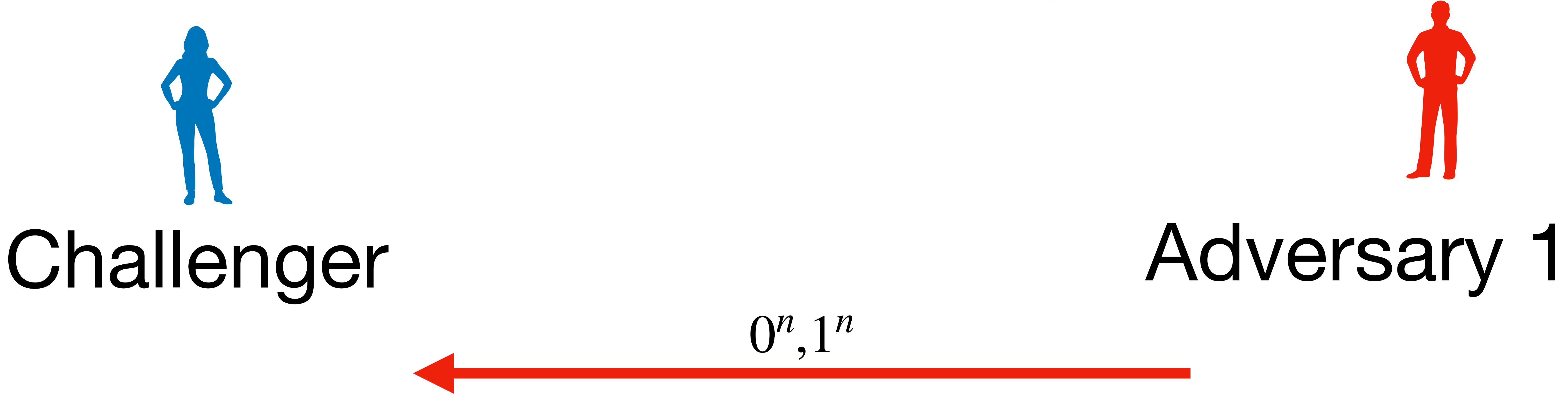


Challenger



Adversary 1

Incompressible Security

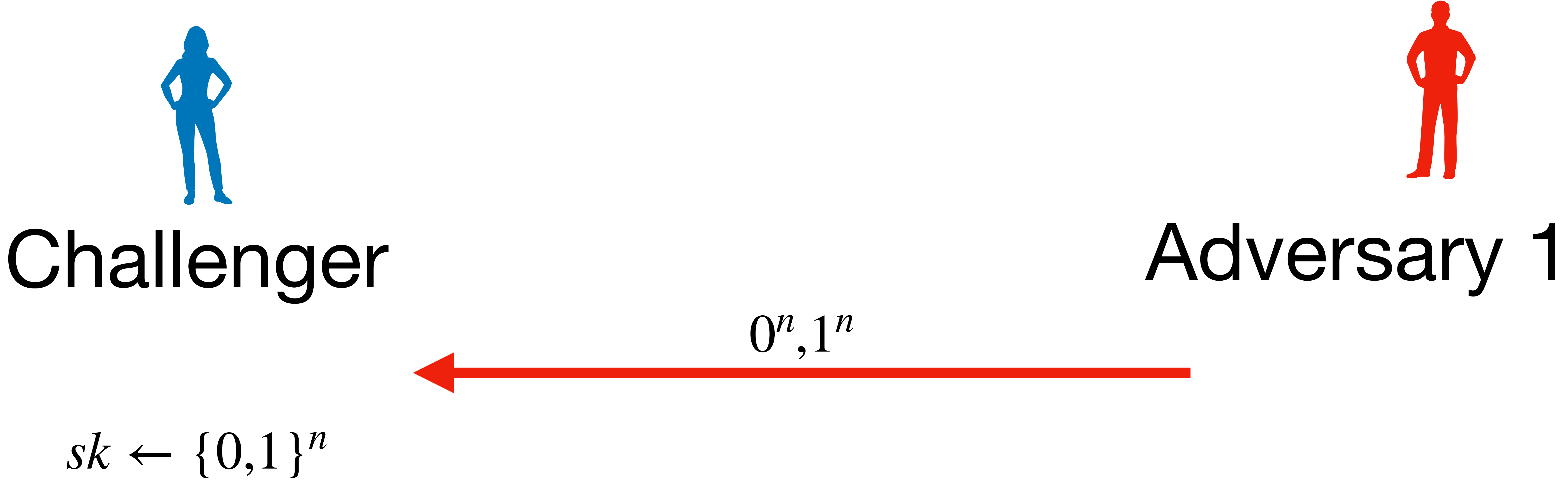


Challenger

Adversary 1

$0^n, 1^n$

Incompressible Security



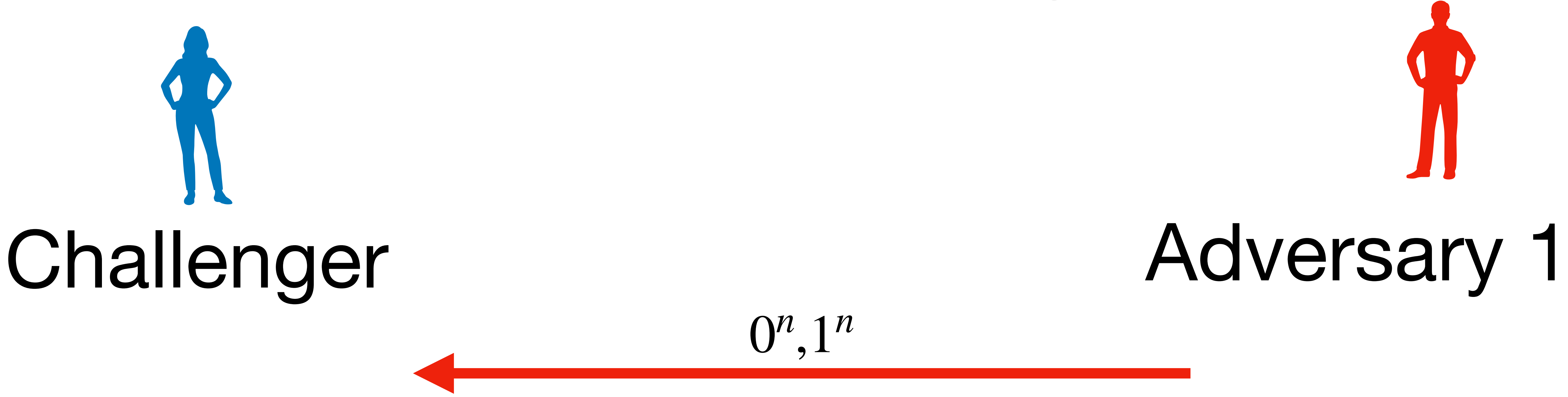
Challenger

Adversary 1

$0^n, 1^n$

$sk \leftarrow \{0,1\}^n$

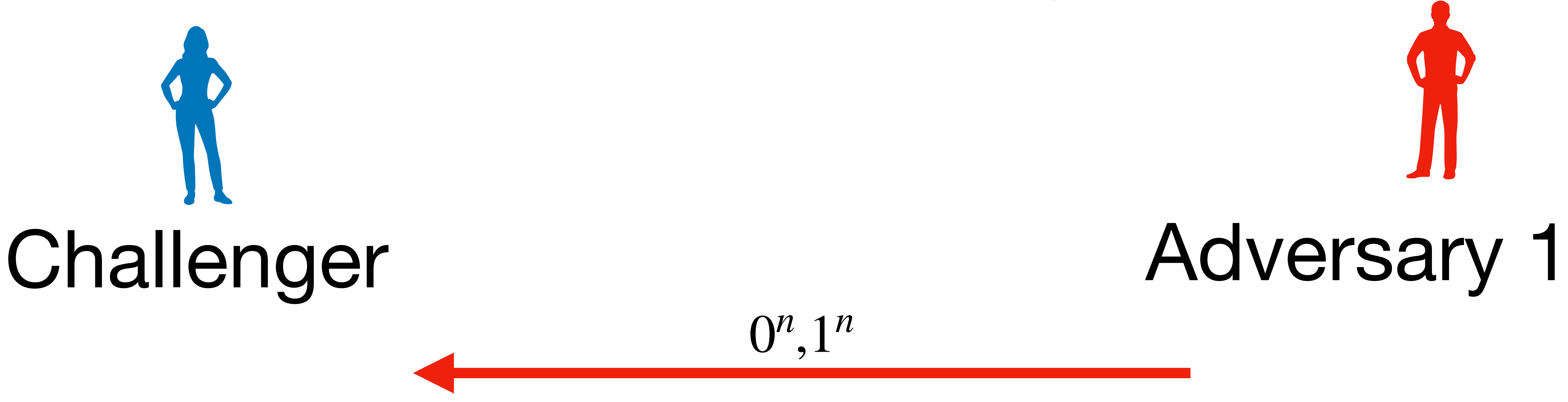
Incompressible Security



$sk \leftarrow \{0,1\}^n$

$b \leftarrow \{0,1\}$

Incompressible Security

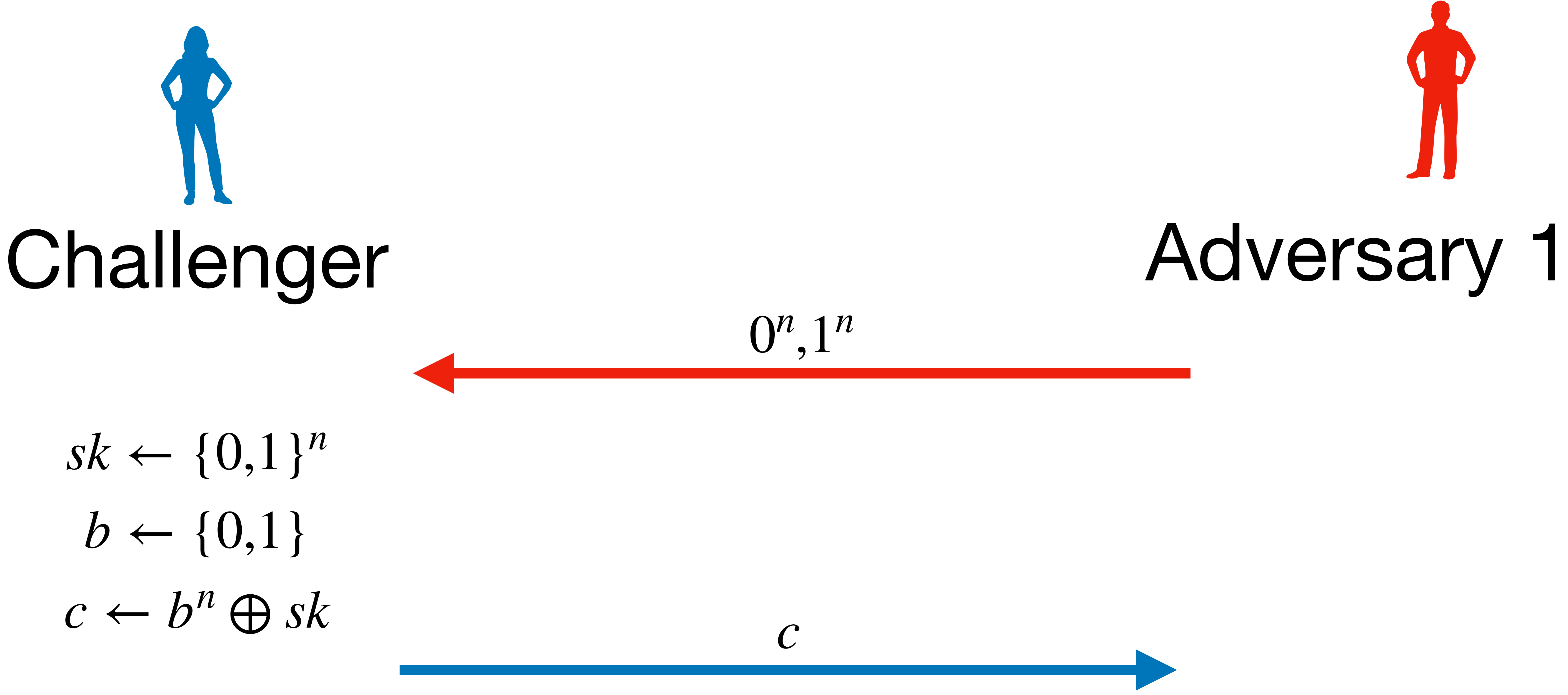


$$sk \leftarrow \{0,1\}^n$$

$$b \leftarrow \{0,1\}$$

$$c \leftarrow b^n \oplus sk$$

Incompressible Security



Challenger

Adversary 1

$0^n, 1^n$

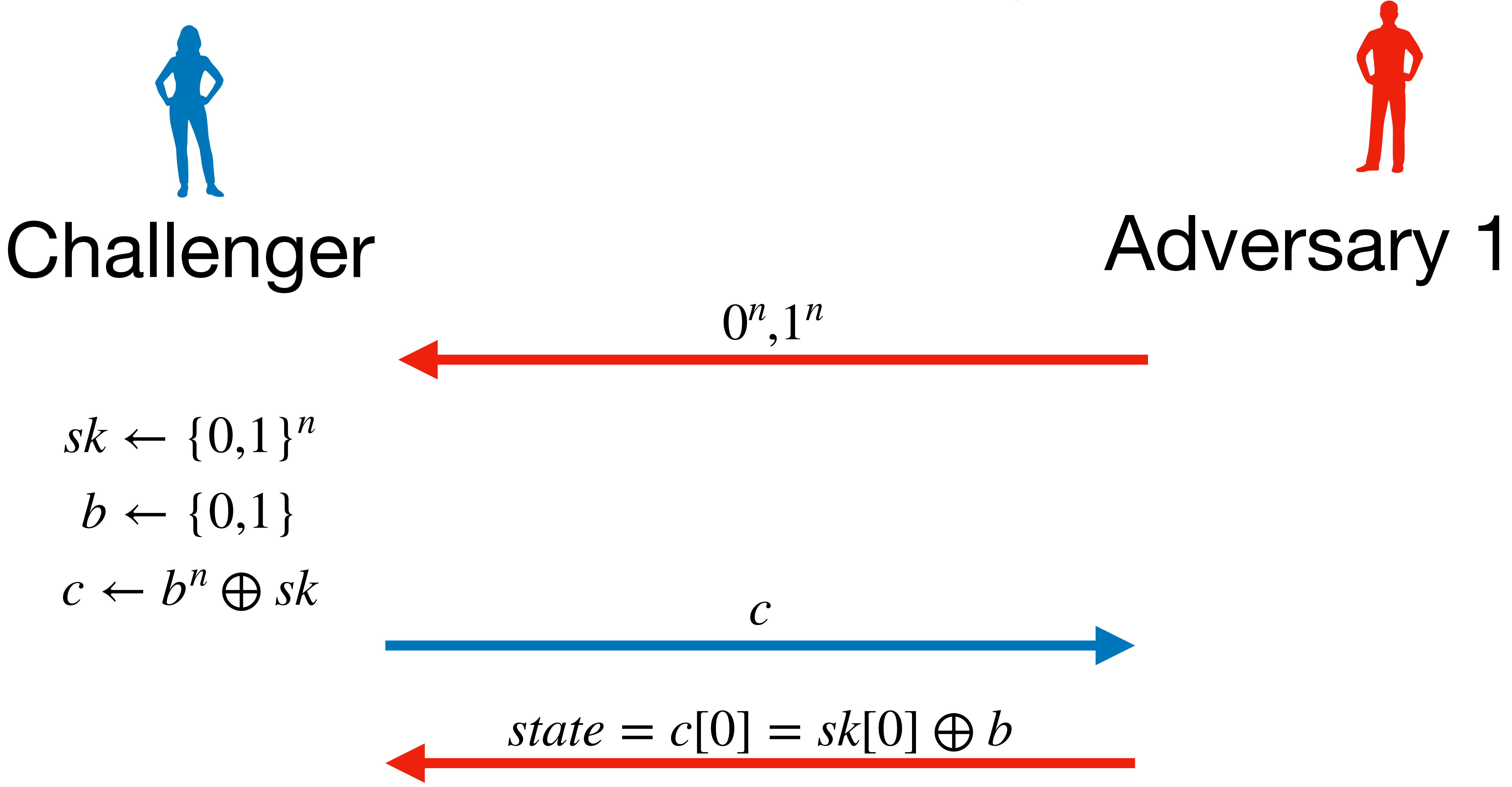
$sk \leftarrow \{0,1\}^n$

$b \leftarrow \{0,1\}$

$c \leftarrow b^n \oplus sk$

c

Incompressible Security



$0^n, 1^n$

$sk \leftarrow \{0,1\}^n$

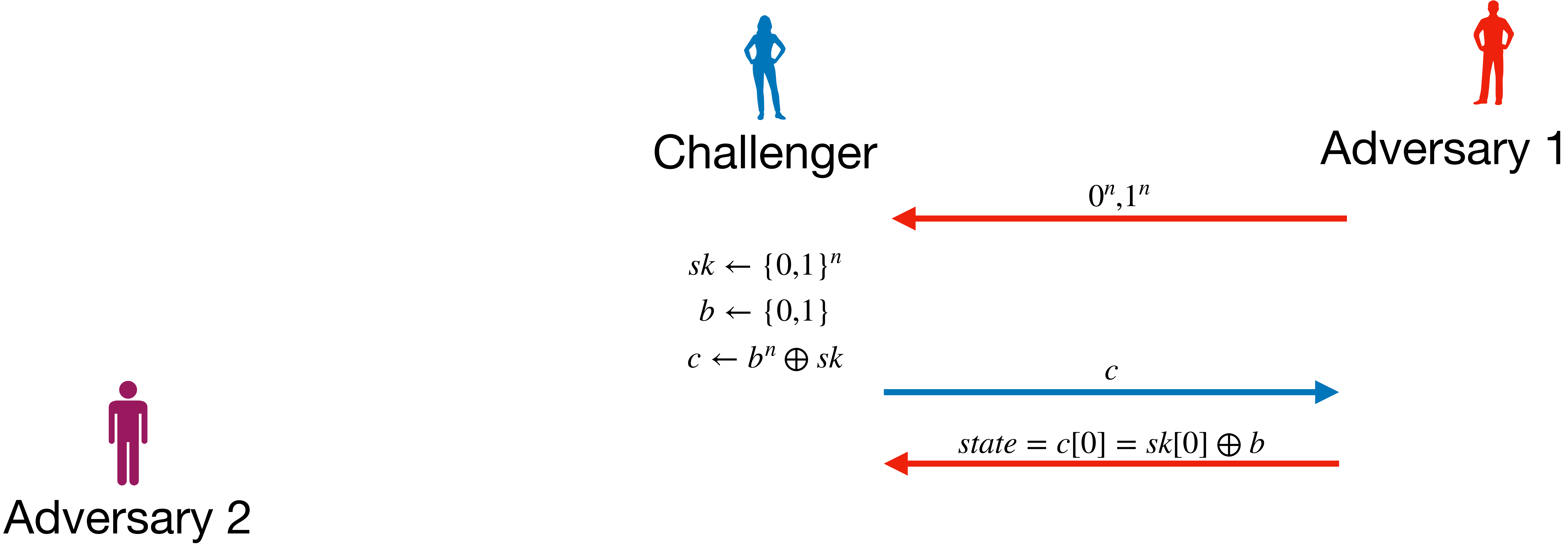
$b \leftarrow \{0,1\}$

$c \leftarrow b^n \oplus sk$

c

$state = c[0] = sk[0] \oplus b$

Incompressible Security



Challenger

Adversary 1

$0^n, 1^n$

$sk \leftarrow \{0,1\}^n$

$b \leftarrow \{0,1\}$

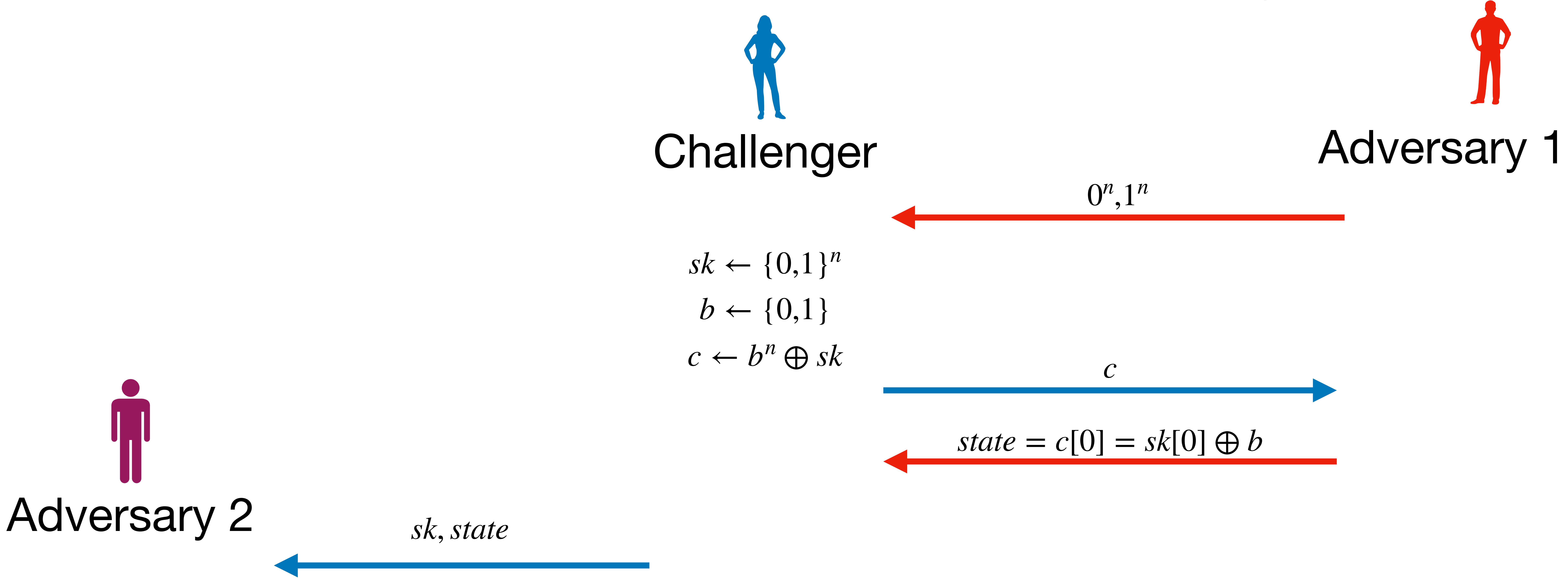
$c \leftarrow b^n \oplus sk$

c

$state = c[0] = sk[0] \oplus b$

Adversary 2

Incompressible Security



Challenger

Adversary 1

$0^n, 1^n$

$sk \leftarrow \{0,1\}^n$

$b \leftarrow \{0,1\}$

$c \leftarrow b^n \oplus sk$

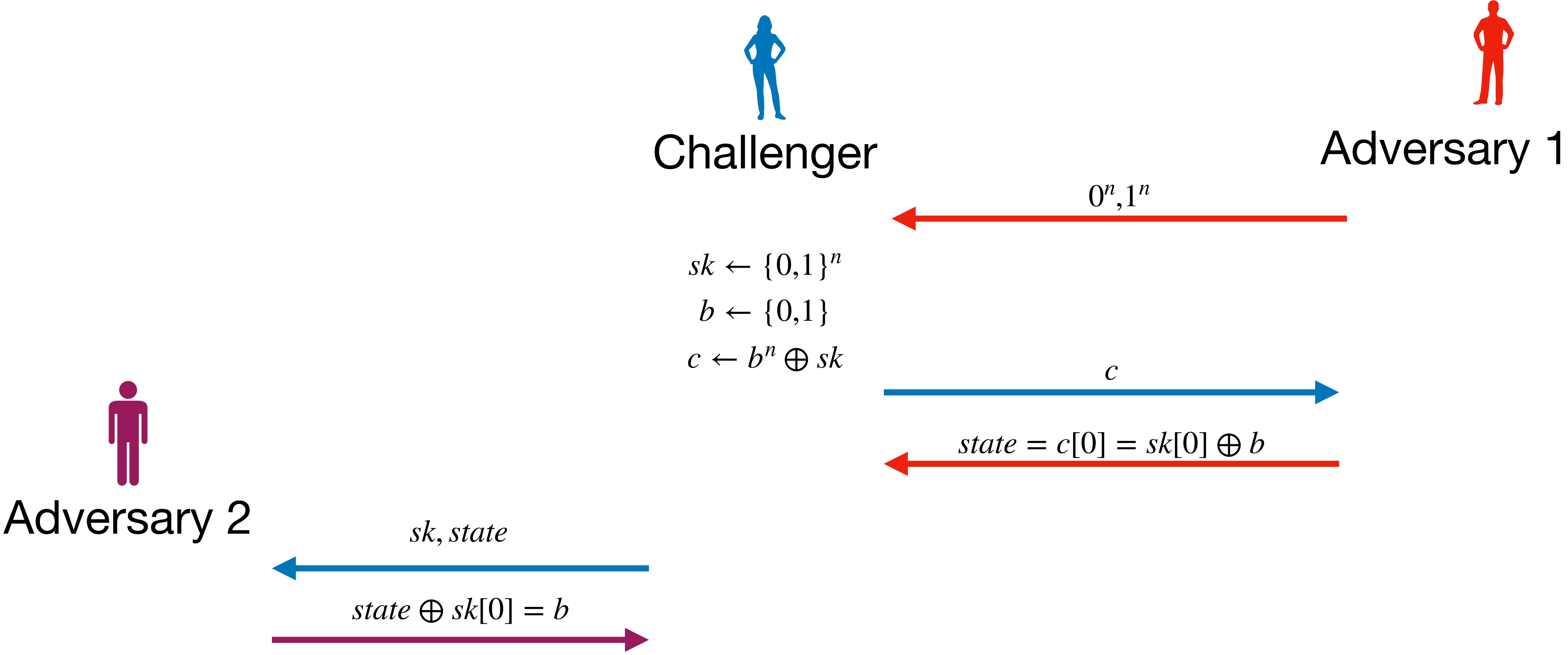
c

$state = c[0] = sk[0] \oplus b$

Adversary 2

$sk, state$

Incompressible Security



Challenger

Adversary 1

$0^n, 1^n$

$sk \leftarrow \{0,1\}^n$

$b \leftarrow \{0,1\}$

$c \leftarrow b^n \oplus sk$

c

$state = c[0] = sk[0] \oplus b$

Adversary 2

$sk, state$

$state \oplus sk[0] = b$

Fixing OTP (Dziembowski06)

Fixing OTP (Dziembowski06)

- Issue is OTP is secure when secret key is kept hidden.

Fixing OTP (Dziembowski06)

- Issue is OTP is secure when secret key is kept hidden.
- Somehow generate a new key sk' from original secret key sk and then use OTP.

Fixing OTP (Dziembowski06)

- Issue is OTP is secure when secret key is kept hidden.
- Somehow generate a new key sk' from original secret key sk and then use OTP.
- Secret key sk is a truly random string.

Fixing OTP (Dziembowski06)

- Issue is OTP is secure when secret key is kept hidden.
- Somehow generate a new key sk' from original secret key sk and then use OTP.
- Secret key sk is a truly random string.
- To encrypt a message m, compute $sk' = Ext(R; sk)$ which will used in OTP. Here, R is a huge random string.

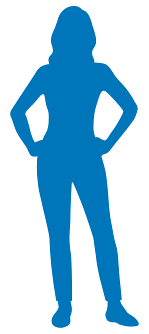
Fixing OTP (Dziembowski06)

- Issue is OTP is secure when secret key is kept hidden.
- Somehow generate a new key sk' from original secret key sk and then use OTP.
- Secret key sk is a truly random string.
- To encrypt a message m, compute $sk' = Ext(R; sk)$ which will used in OTP. Here, R is a huge random string.
- Compute $c = (R, m \oplus sk')$.

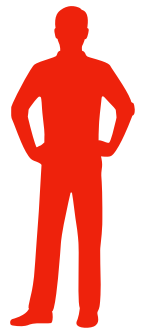
Incompressible Security

Incompressible Security

Incompressible Security

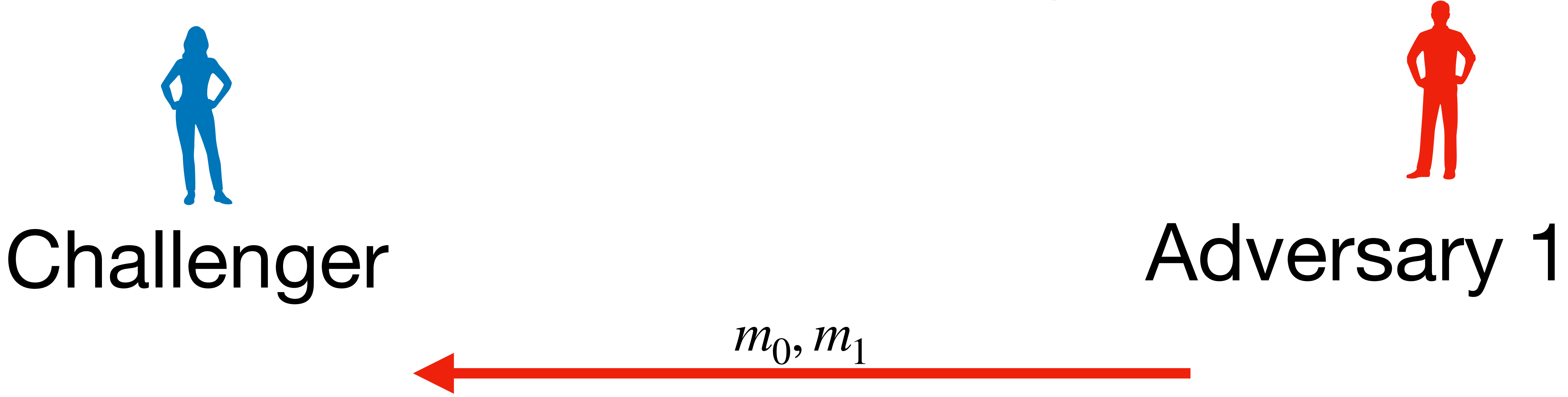


Challenger



Adversary 1

Incompressible Security

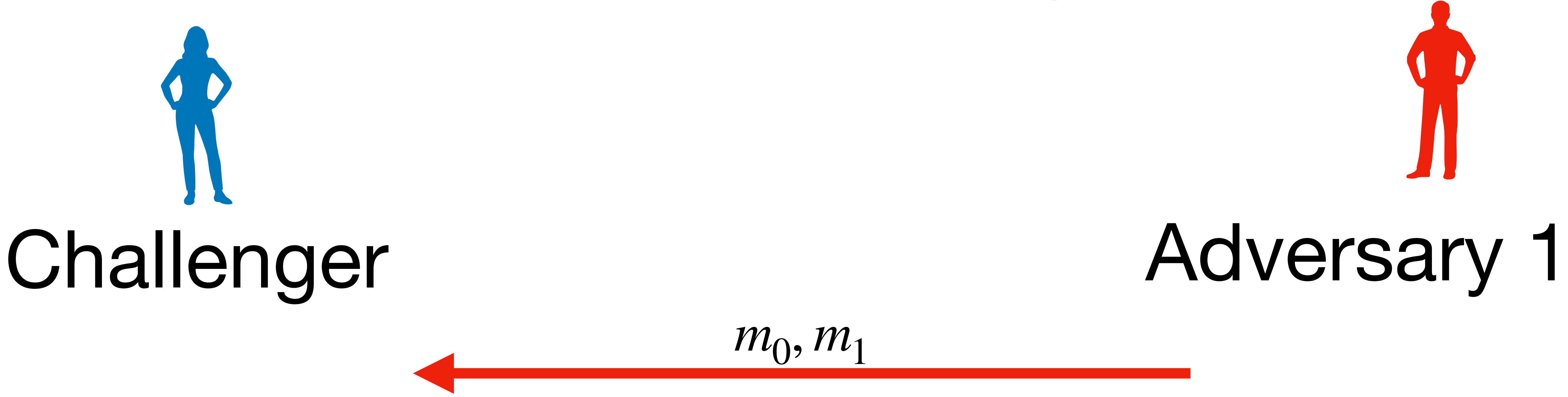


Challenger

Adversary 1

m_0, m_1

Incompressible Security



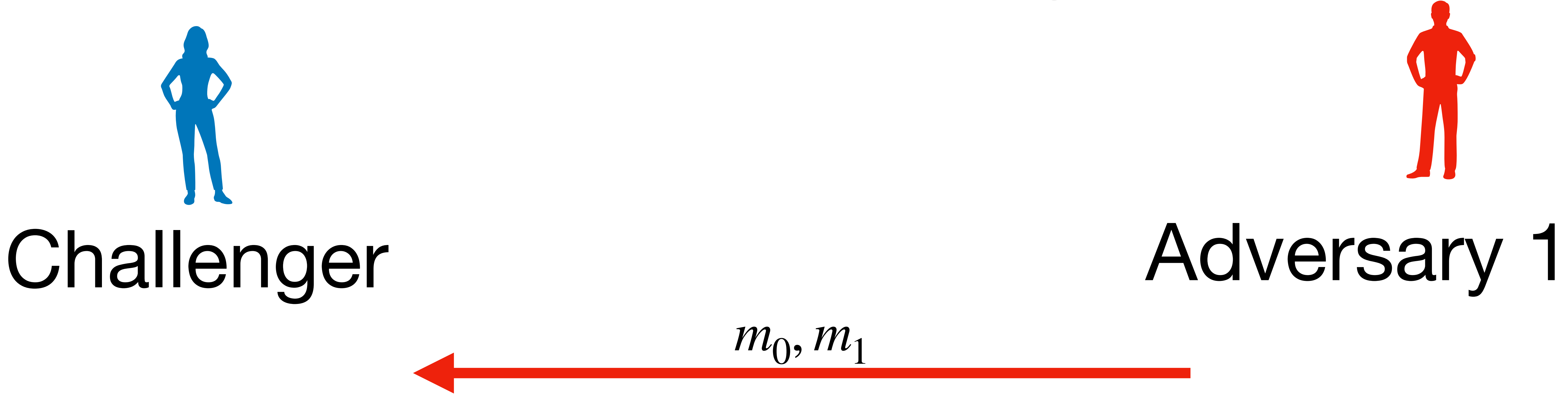
Challenger

Adversary 1

m_0, m_1

$sk \leftarrow \{0,1\}^\ell$

Incompressible Security



Challenger

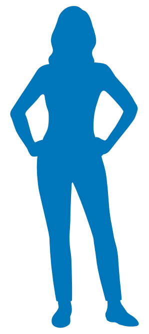
Adversary 1

m_0, m_1

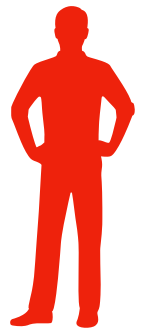
$$sk \leftarrow \{0,1\}^\ell$$

$$b \leftarrow \{0,1\}$$

Incompressible Security



Challenger



Adversary 1

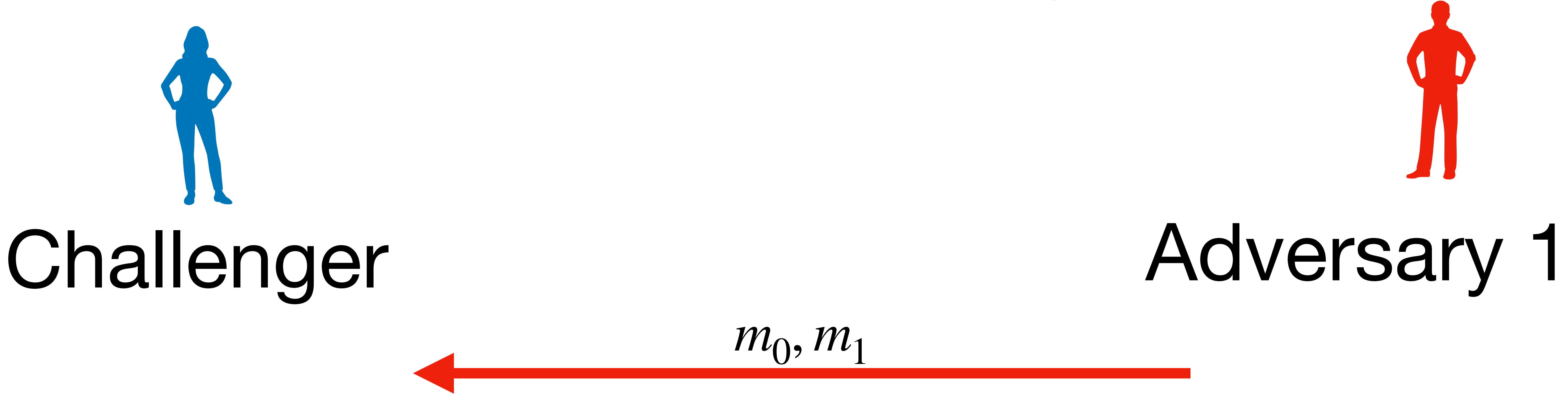
m_0, m_1

$$sk \leftarrow \{0,1\}^{\ell}$$

$$b \leftarrow \{0,1\}$$

$$sk' = Ext(R; sk)$$

Incompressible Security



Challenger

Adversary 1

m_0, m_1

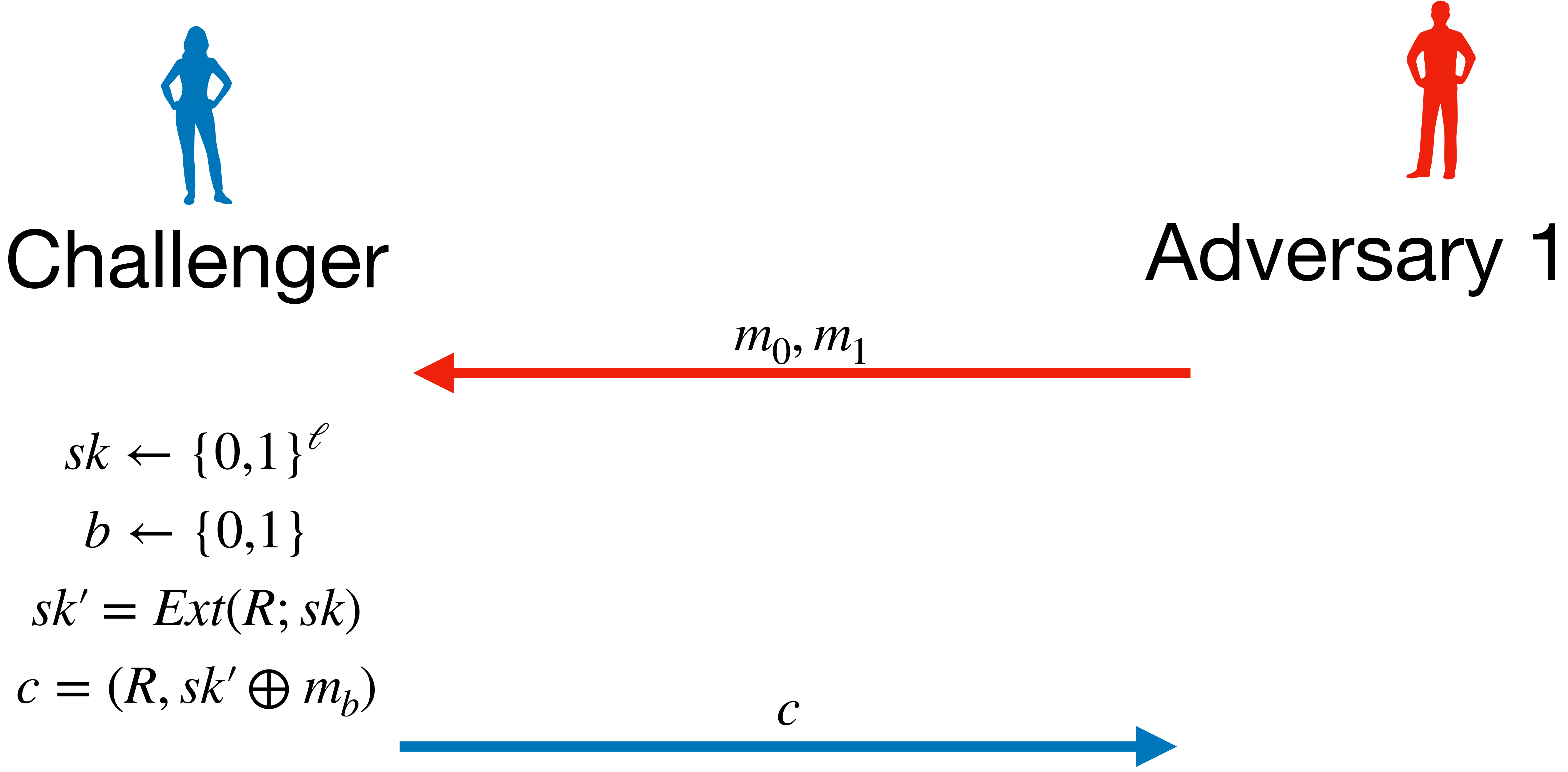
$$sk \leftarrow \{0,1\}^{\ell}$$

$$b \leftarrow \{0,1\}$$

$$sk' = Ext(R; sk)$$

$$c = (R, sk' \oplus m_b)$$

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$$sk \leftarrow \{0,1\}^{\ell}$$

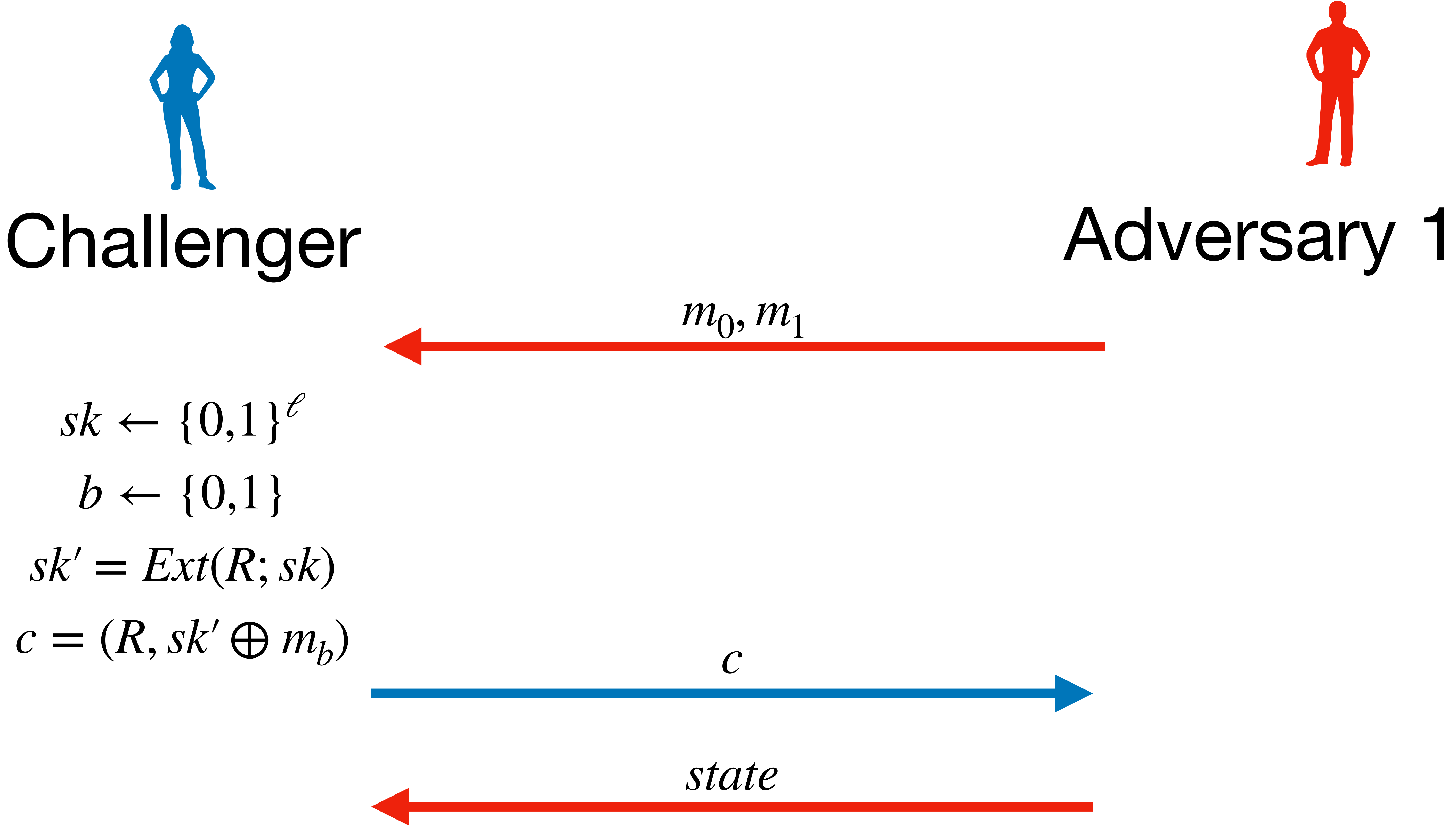
$$b \leftarrow \{0,1\}$$

$$sk' = Ext(R; sk)$$

$$c = (R, sk' \oplus m_b)$$

c

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$$sk \leftarrow \{0,1\}^{\ell}$$

$$b \leftarrow \{0,1\}$$

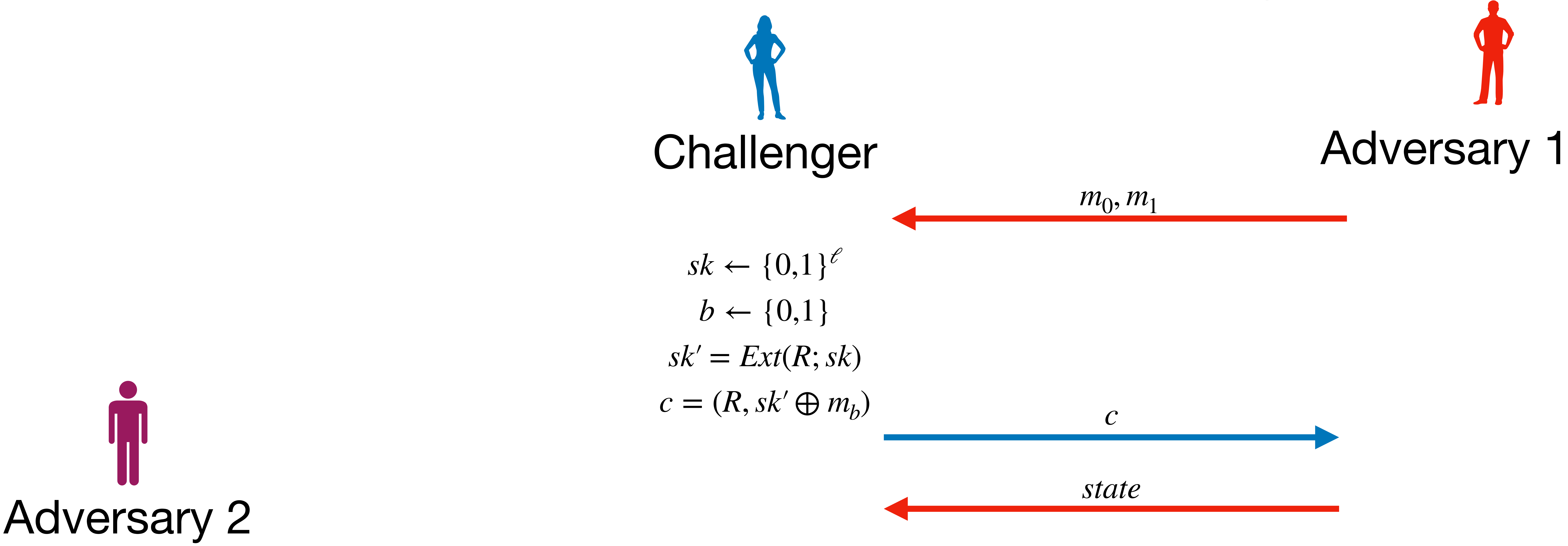
$$sk' = Ext(R; sk)$$

$$c = (R, sk' \oplus m_b)$$

c

$state$

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow \{0,1\}^\ell$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

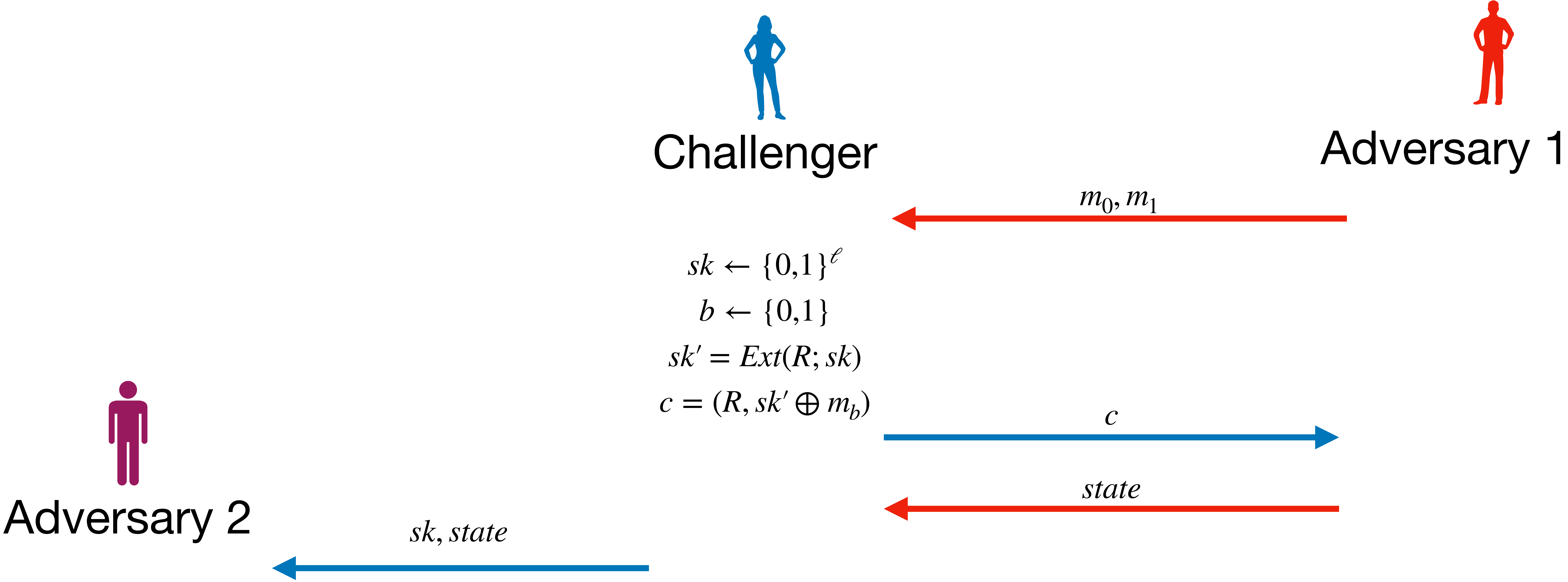
$c = (R, sk' \oplus m_b)$

c

$state$

Adversary 2

Incompressible Security



Challenger

Adversary 1

Adversary 2

m_0, m_1

$sk \leftarrow \{0,1\}^{\ell}$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

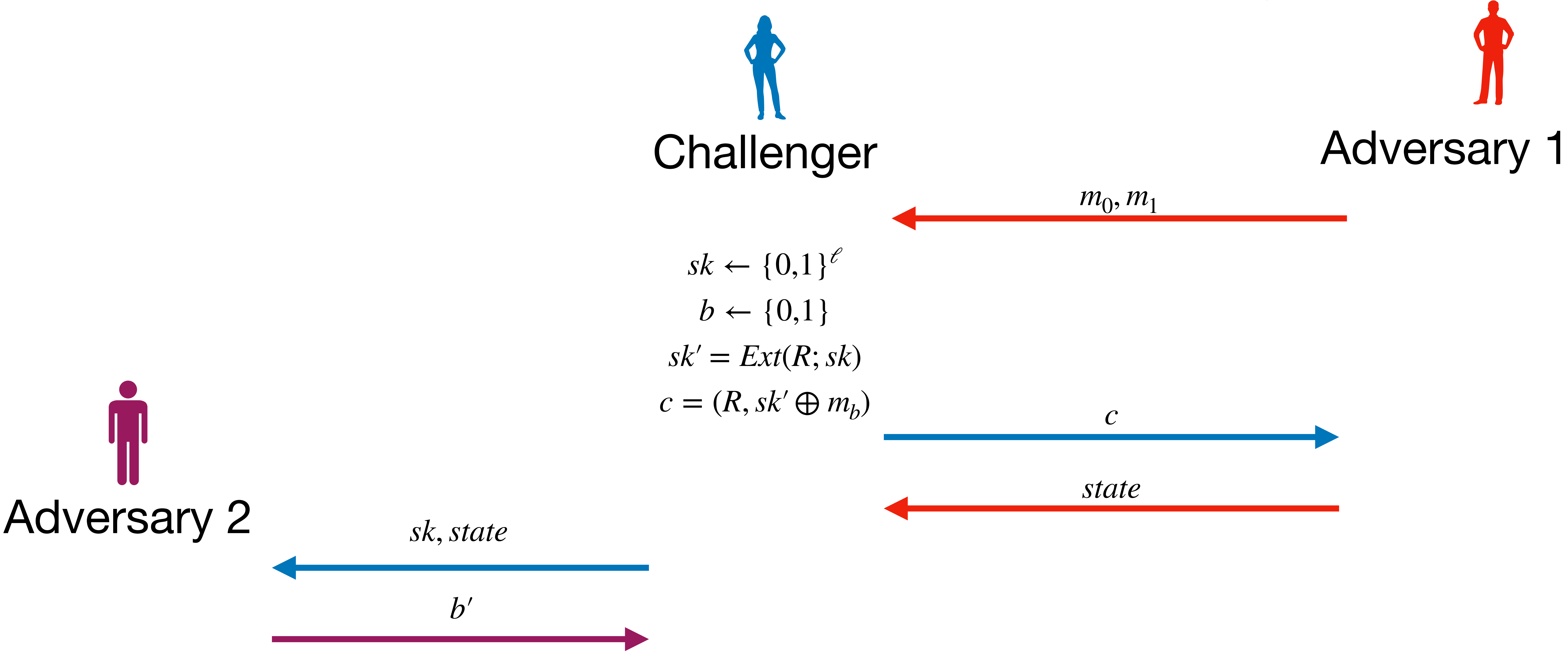
$c = (R, sk' \oplus m_b)$

c

$state$

$sk, state$

Incompressible Security



Challenger

Adversary 1

Adversary 2

m_0, m_1

$sk \leftarrow \{0,1\}^\ell$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

$c = (R, sk' \oplus m_b)$

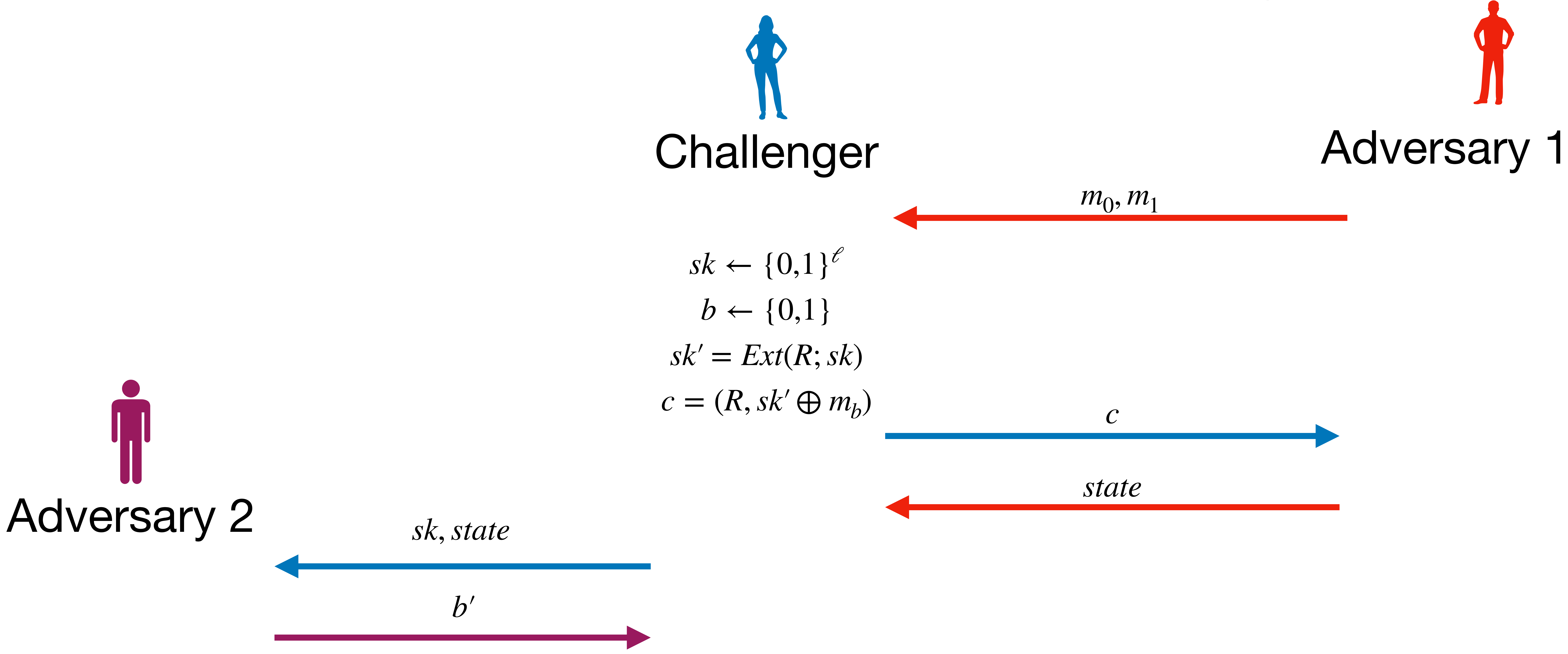
c

$state$

$sk, state$

b'

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow \{0,1\}^\ell$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

$c = (R, sk' \oplus m_b)$

c

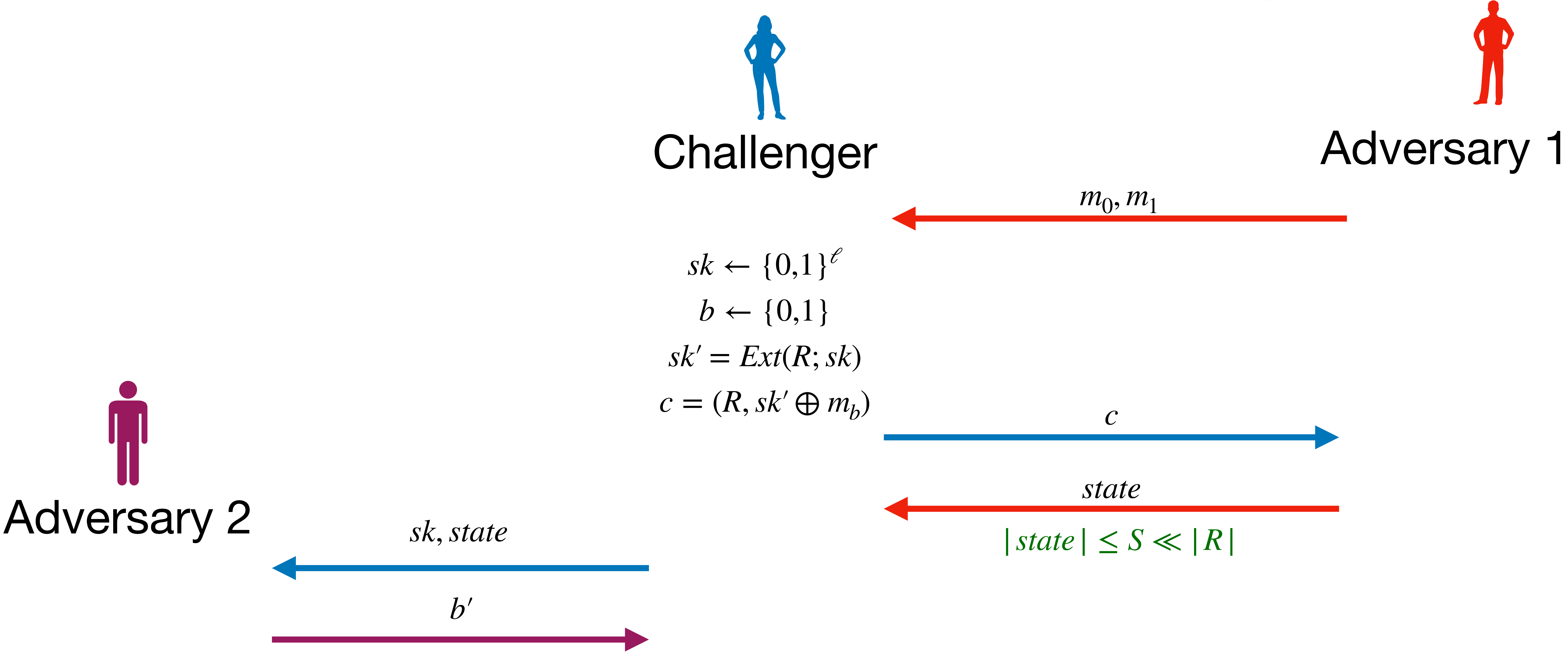
$state$

Adversary 2

$sk, state$

b'

Incompressible Security



Challenger

Adversary 1

Adversary 2

m_0, m_1

$sk \leftarrow \{0,1\}^{\ell}$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

$c = (R, sk' \oplus m_b)$

c

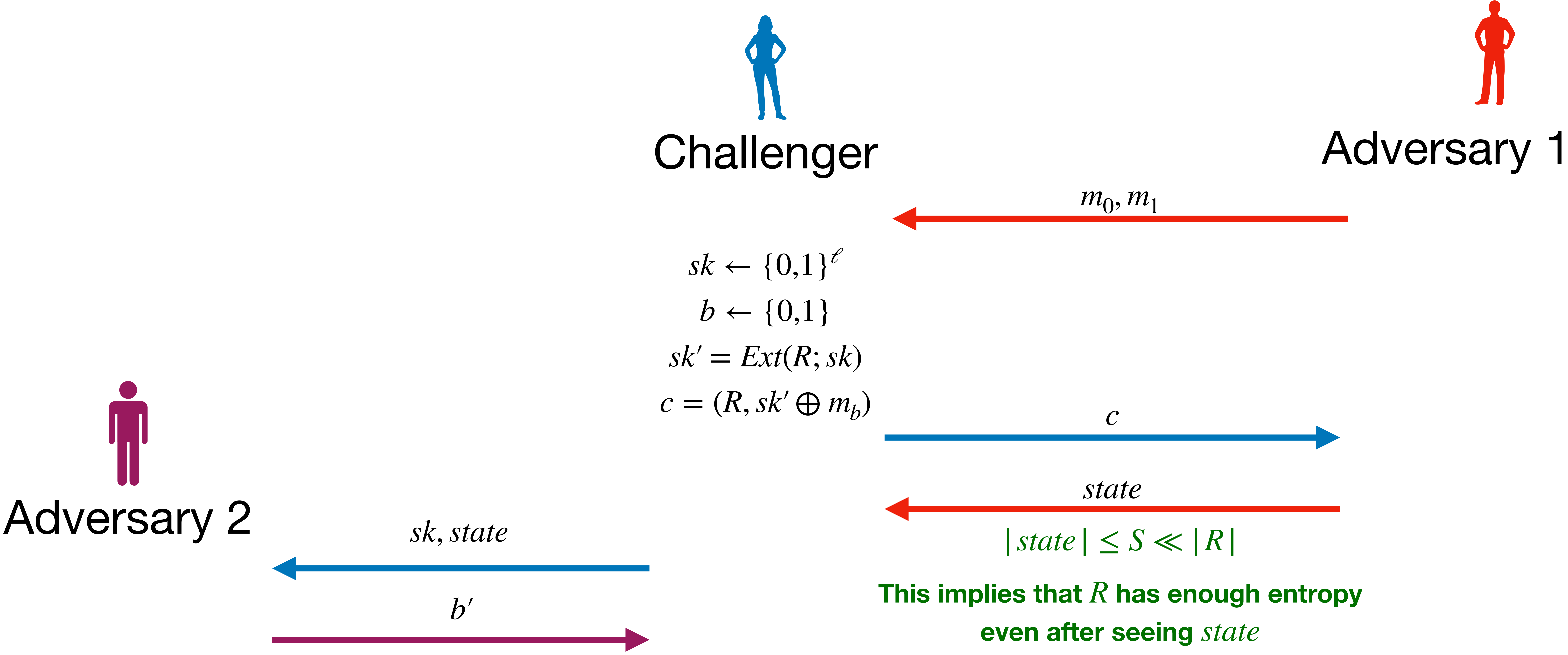
$state$

$|state| \leq S \ll |R|$

$sk, state$

b'

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$$sk \leftarrow \{0,1\}^{\ell}$$

$$b \leftarrow \{0,1\}$$

$$sk' = Ext(R; sk)$$

$$c = (R, sk' \oplus m_b)$$

c

$state$

$|state| \le S \ll |R|$

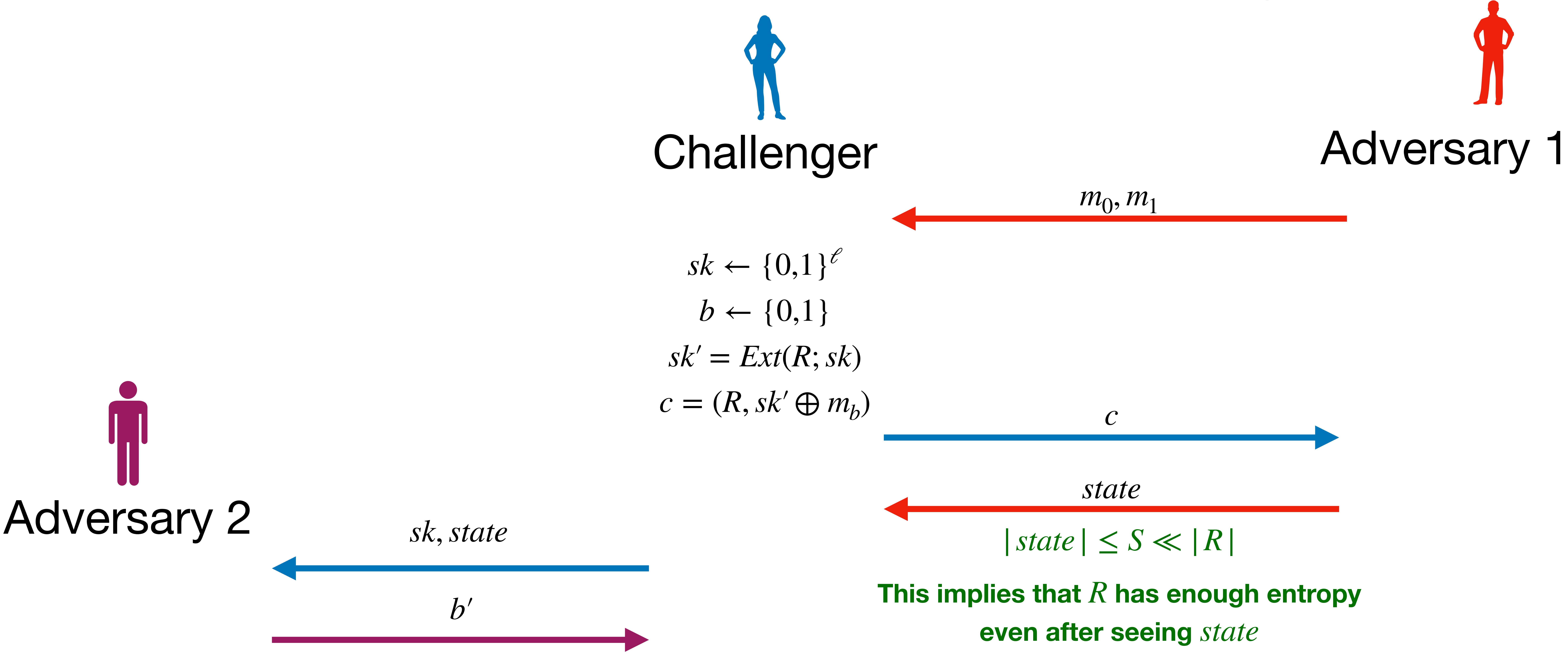
This implies that R has enough entropy even after seeing $state$

Adversary 2

$sk, state$

b'

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow \{0,1\}^{\ell}$

$b \leftarrow \{0,1\}$

$sk' = Ext(R; sk)$

$c = (R, sk' \oplus m_b)$

c

$state$

$|state| \leq S \ll |R|$

Adversary 2

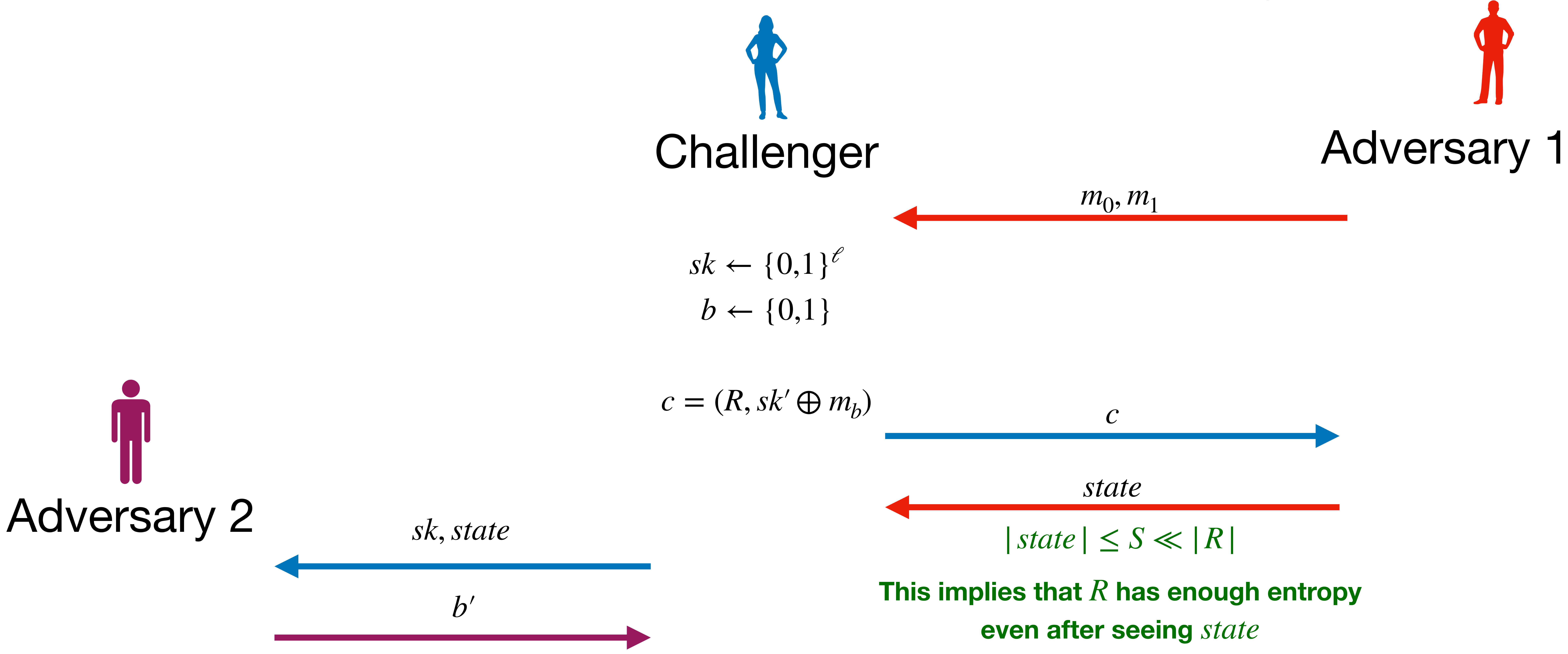
$sk, state$

b'

This implies that R has enough entropy even after seeing $state$

$(sk, Ext(R; sk))$ is statistically close to a truly random string even in the presence of $state$

Incompressible Security



Challenger

Adversary 1

m_0, m_1

$sk \leftarrow \{0,1\}^{\ell}$

$b \leftarrow \{0,1\}$

$c = (R, sk' \oplus m_b)$

c

$state$

$|state| \leq S \ll |R|$

Adversary 2

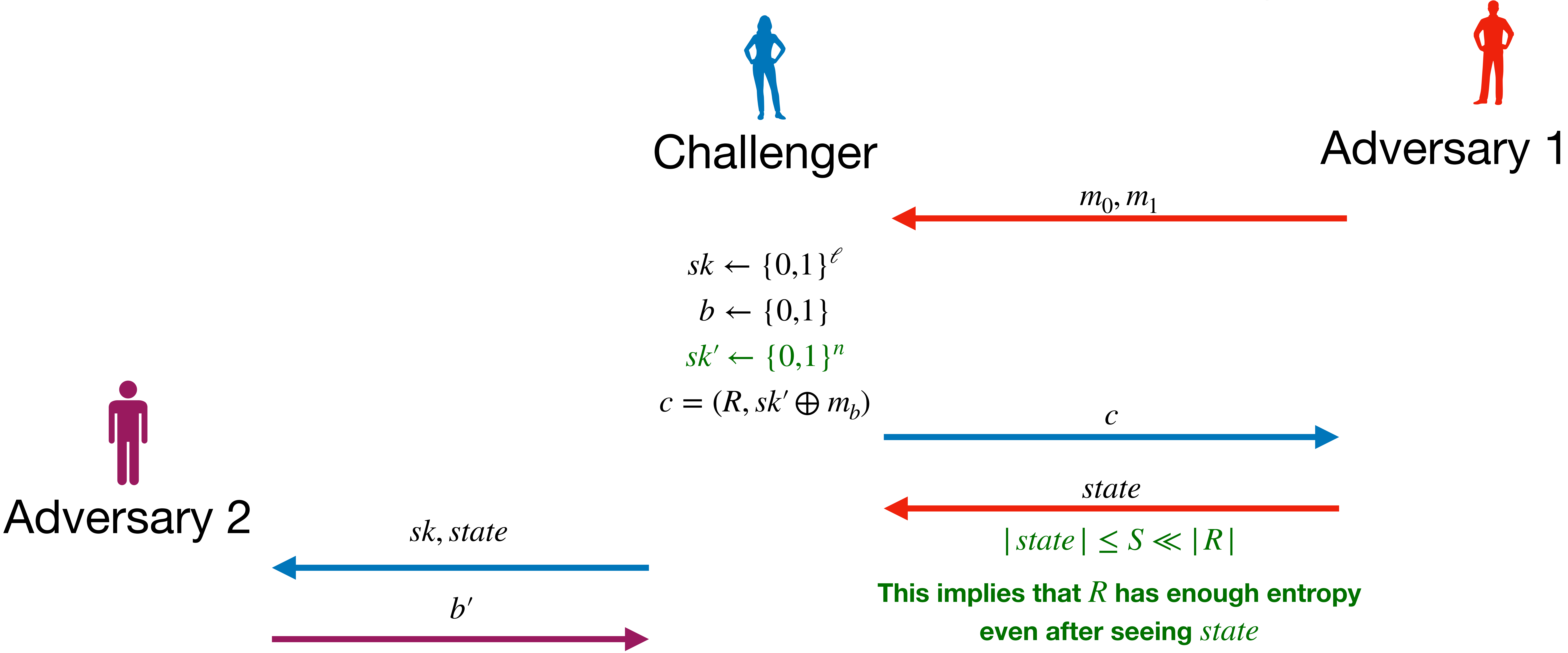
$sk, state$

b'

This implies that R has enough entropy even after seeing $state$

$(sk, Ext(R; sk))$ is statistically close to a truly random string even in the presence of $state$

Incompressible Security



Challenger

Adversary 1

Adversary 2

m_0, m_1

$sk \leftarrow \{0,1\}^\ell$

$b \leftarrow \{0,1\}$

$sk' \leftarrow \{0,1\}^n$

$c = (R, sk' \oplus m_b)$

c

$state$

$|state| \leq S \ll |R|$

$sk, state$

b'

This implies that R has enough entropy even after seeing $state$

$(sk, Ext(R; sk))$ is statistically close to a truly random string even in the presence of $state$

Our Construction for Incomp PKE

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.
- Garbling scheme - Given a circuit $C : \{0,1\}^n \rightarrow \{0,1\}^m$, a garbling scheme generates a new circuit $\tilde{C}$ and $2n$ labels

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.
- Garbling scheme - Given a circuit $C : \{0,1\}^n \rightarrow \{0,1\}^m$, a garbling scheme generates a new circuit $\tilde{C}$ and $2n$ labels

$$lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}$$

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.
- Garbling scheme - Given a circuit $C : \{0,1\}^n \rightarrow \{0,1\}^m$, a garbling scheme generates a new circuit $\tilde{C}$ and $2n$ labels

$$lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}$$

$$lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}$$

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.
- Garbling scheme - Given a circuit $C : \{0,1\}^n \rightarrow \{0,1\}^m$, a garbling scheme generates a new circuit $\tilde{C}$ and $2n$ labels

$$lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}$$

$$lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}$$

such that for any $(x_1, \ldots, x_n) \in \{0,1\}^n$

Our Construction for Incomp PKE

- Primitive required - PKE, incompressible SKE and garbling scheme.
- Garbling scheme - Given a circuit $C: \{0,1\}^n \rightarrow \{0,1\}^m$, a garbling scheme generates a new circuit $\tilde{C}$ and $2n$ labels

$$lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}$$

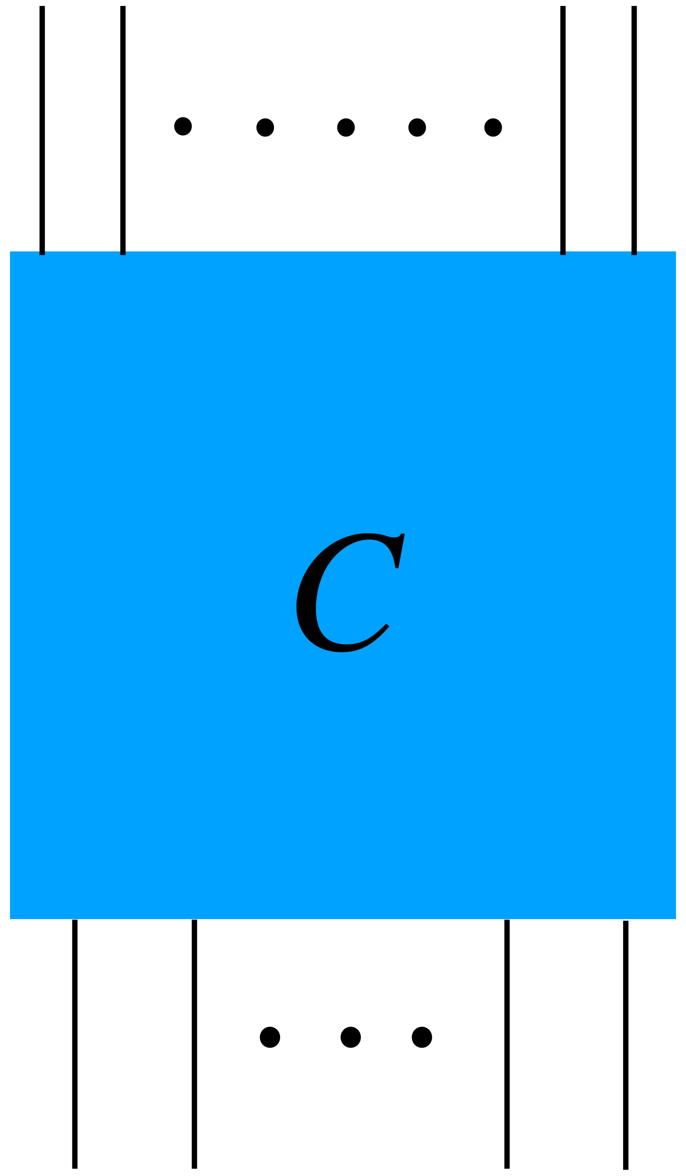
$$lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}$$

such that for any $(x_1, \ldots, x_n) \in \{0,1\}^n$

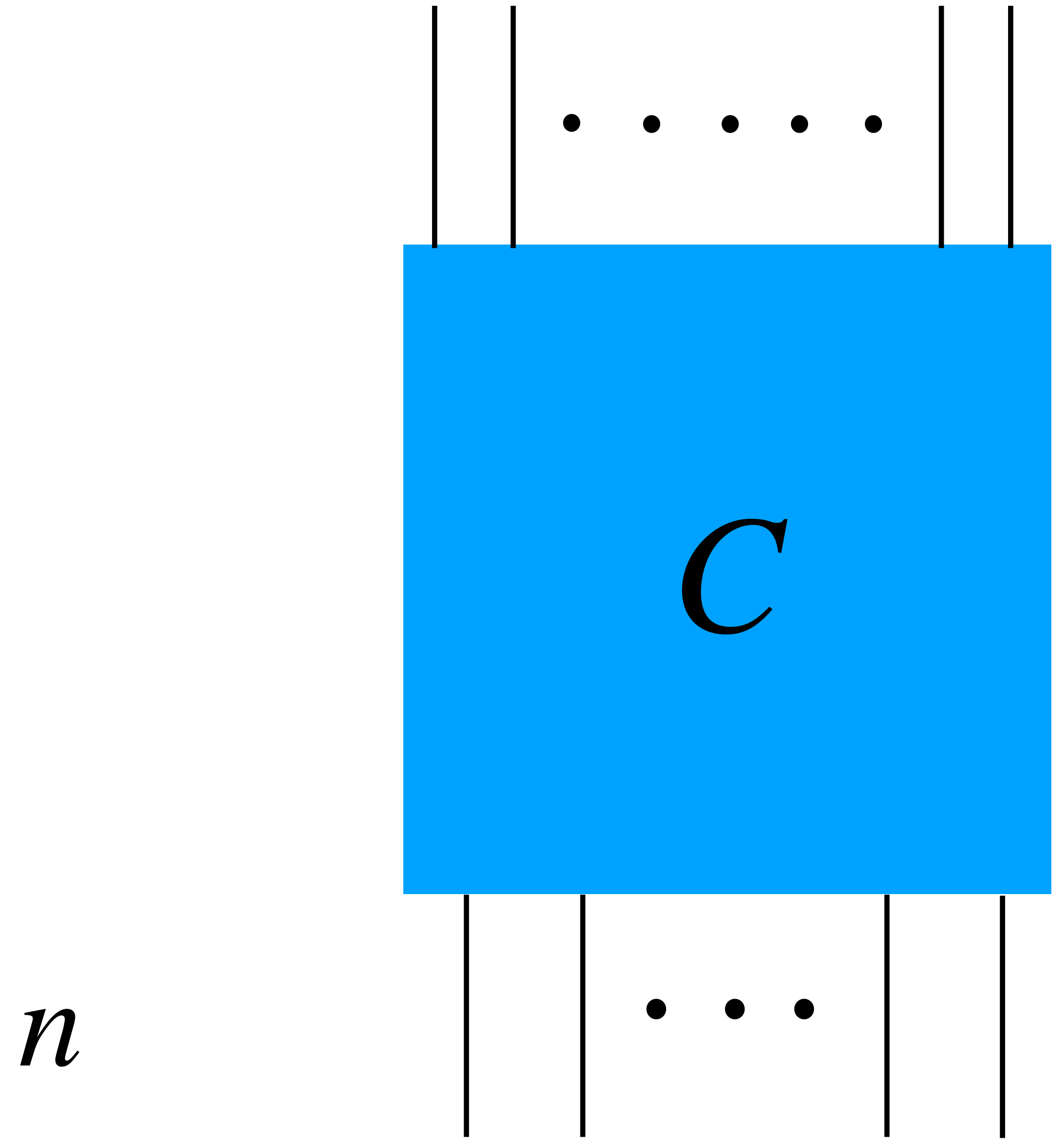
$$\tilde{C}(lab_{1,x_1}, lab_{2,x_2}, \ldots, lab_{n,x_n}) = C(x_1, \ldots, x_n)$$

Garbling Scheme

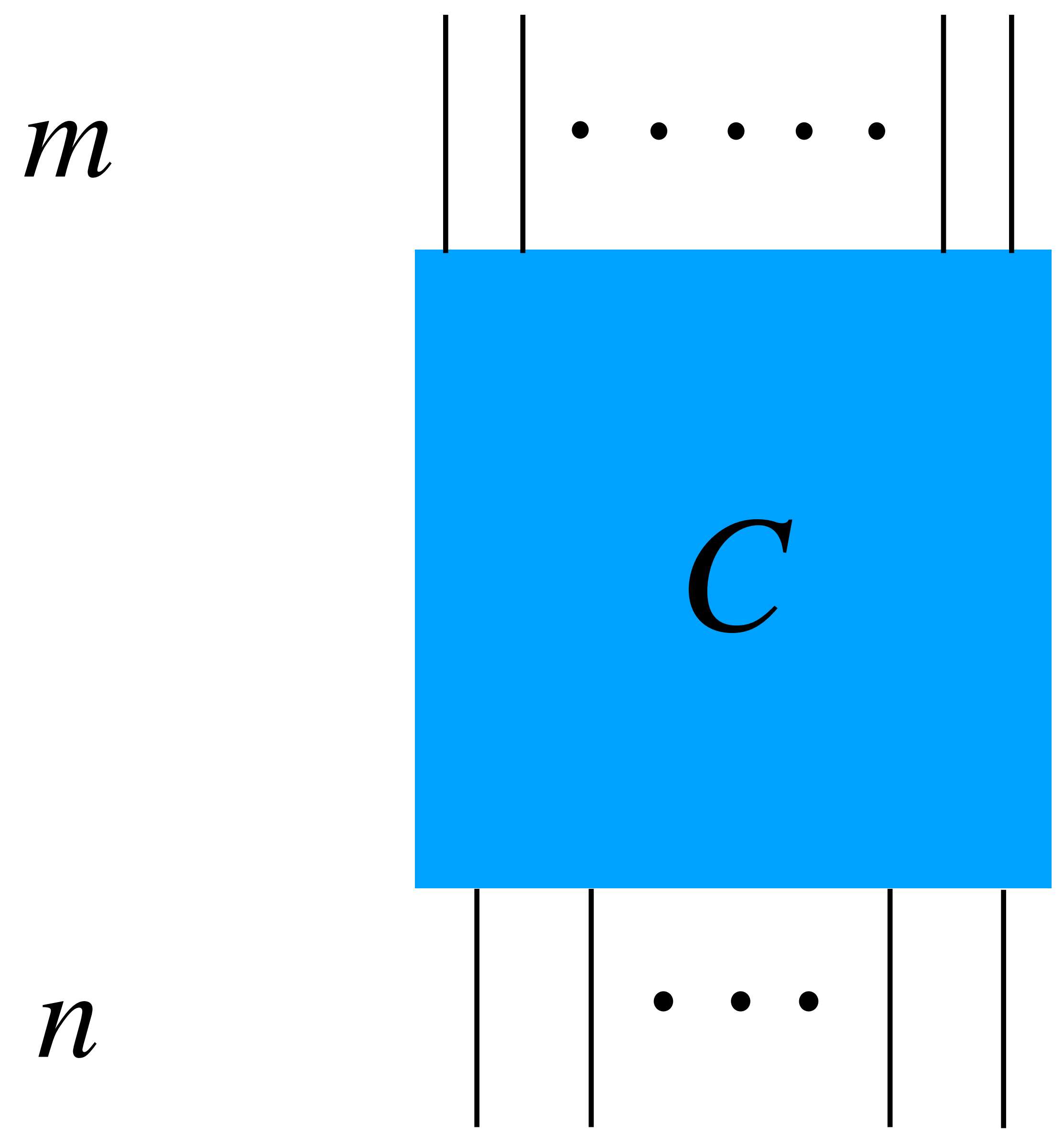
Garbling Scheme



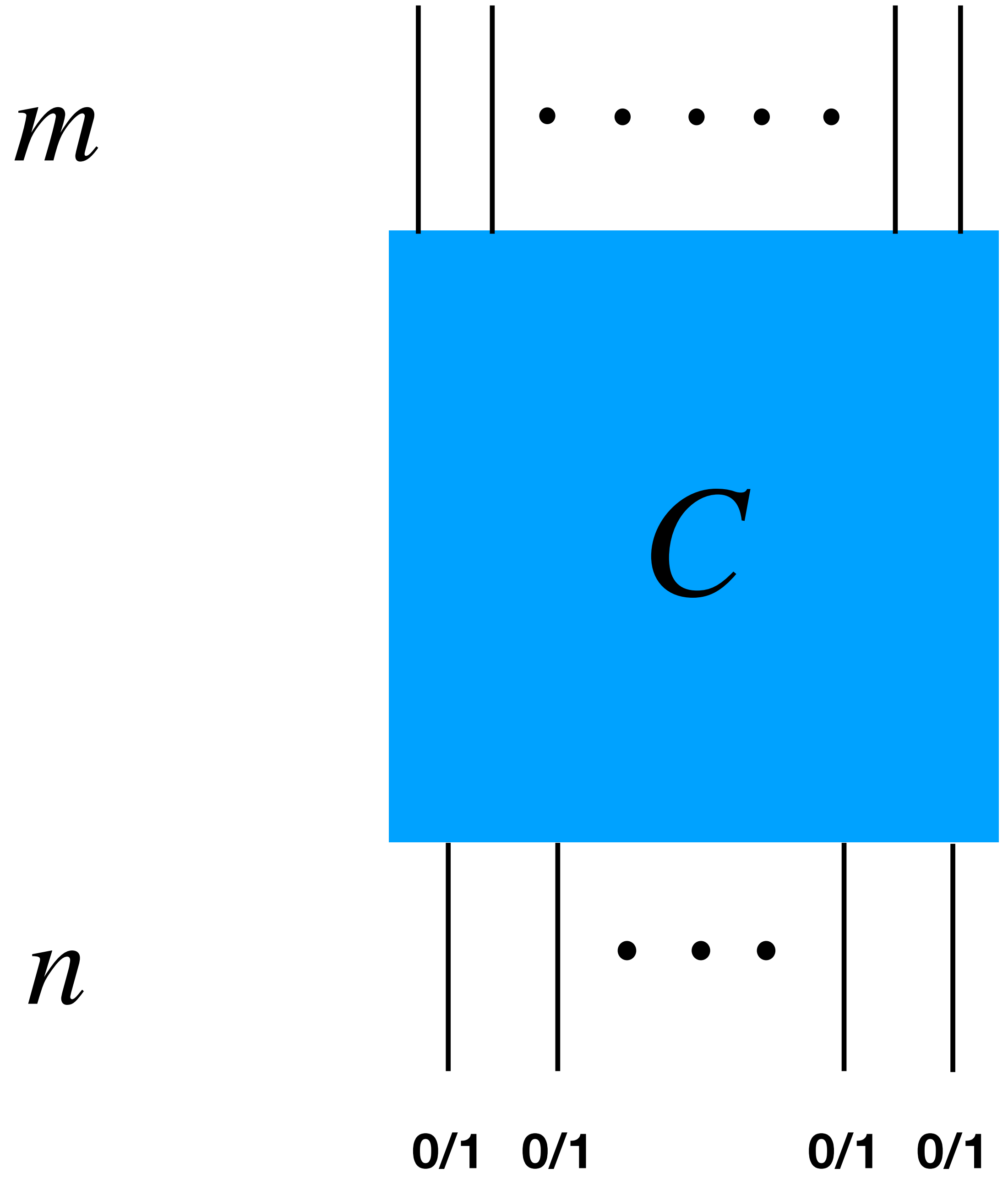
Garbling Scheme



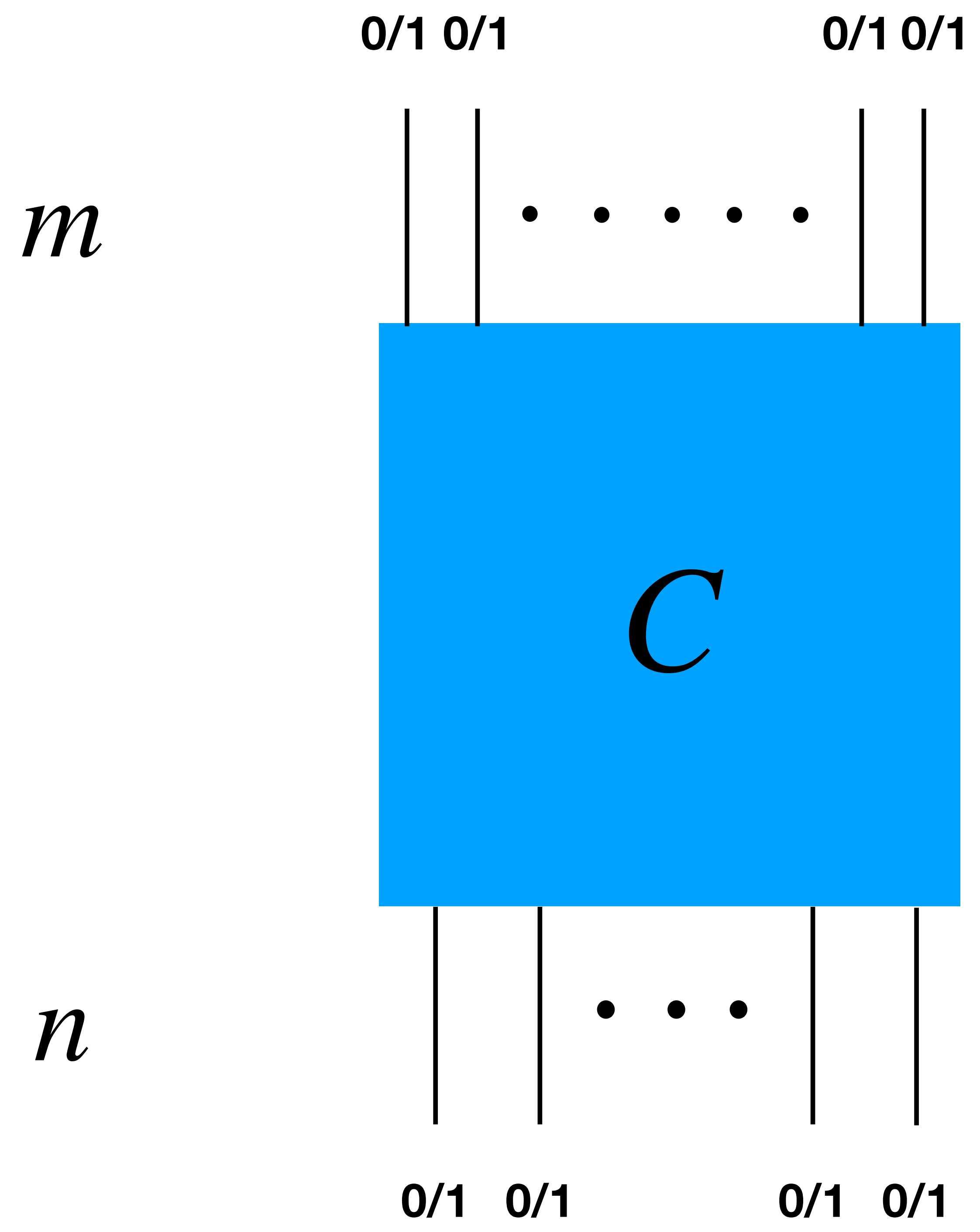
Garbling Scheme



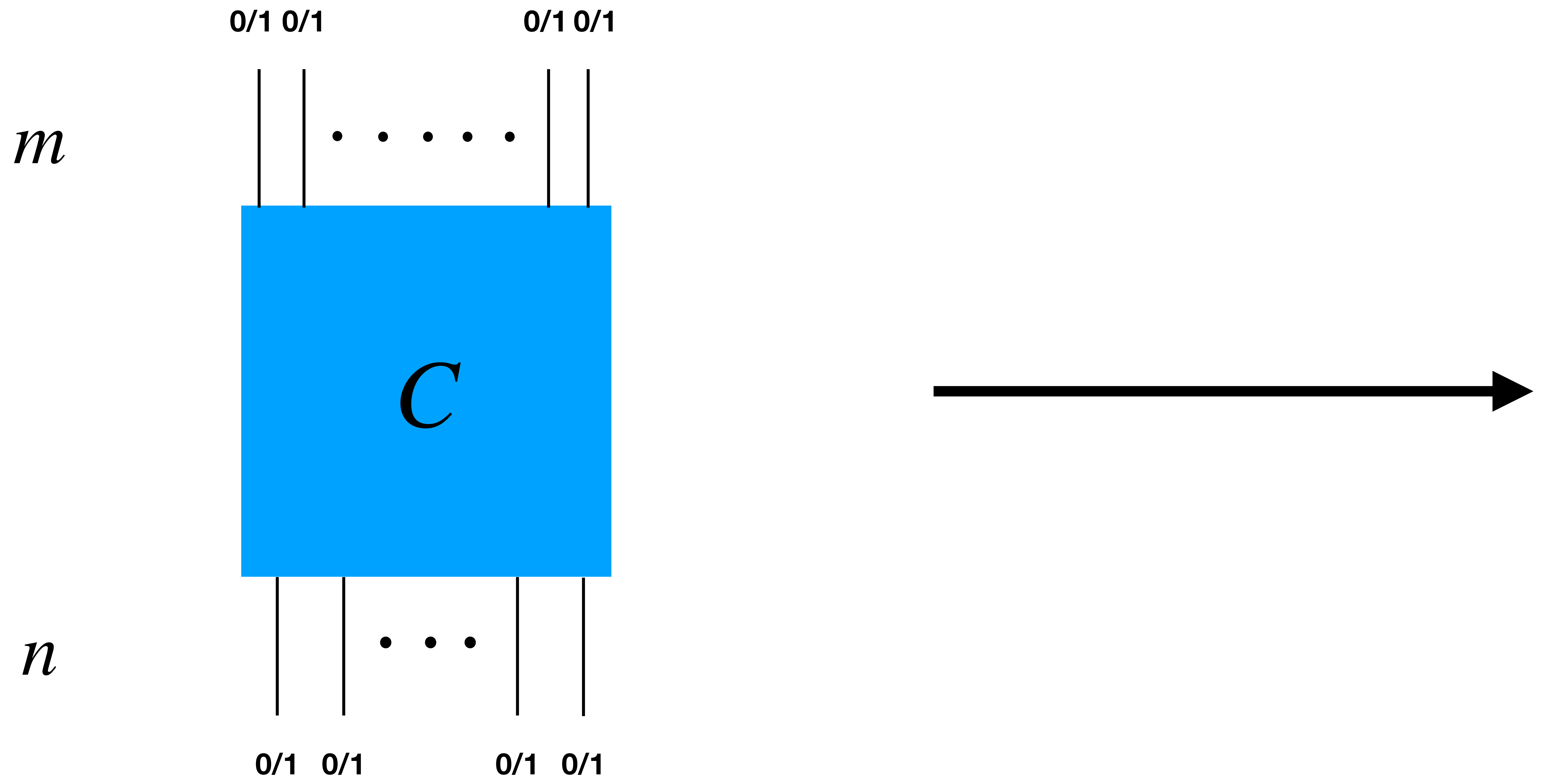
Garbling Scheme



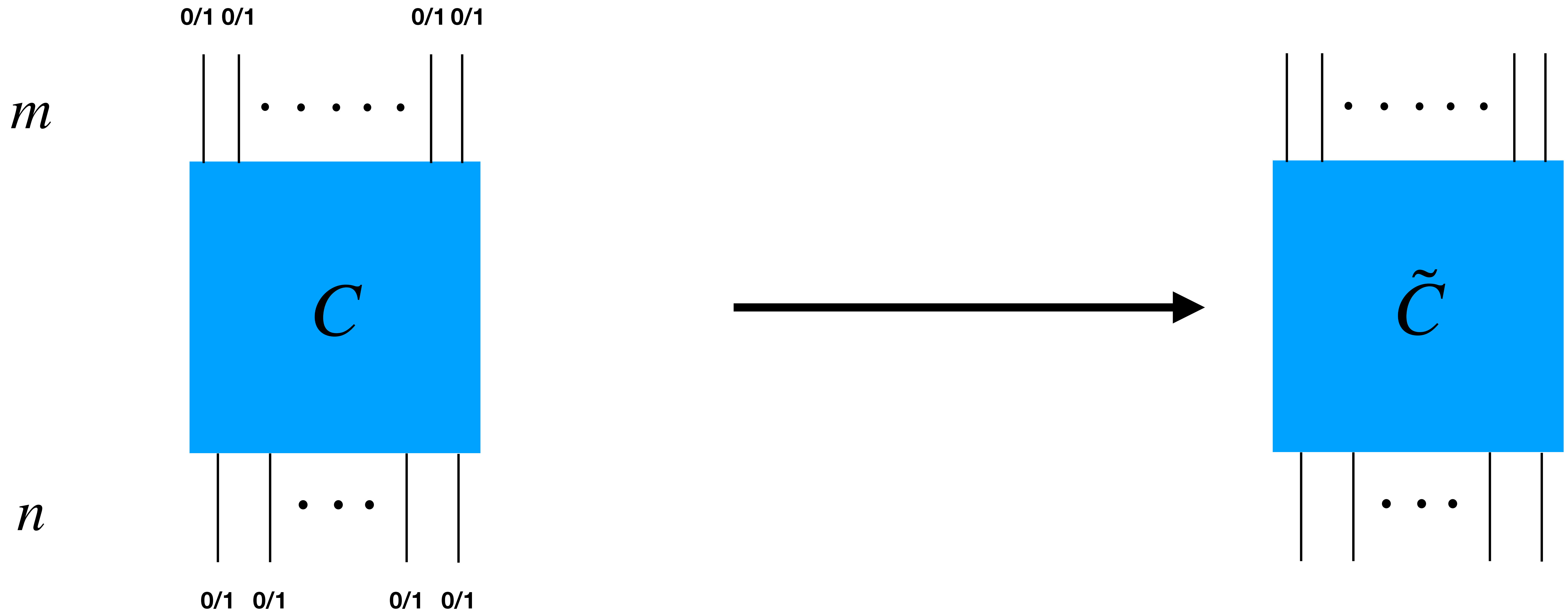
Garbling Scheme



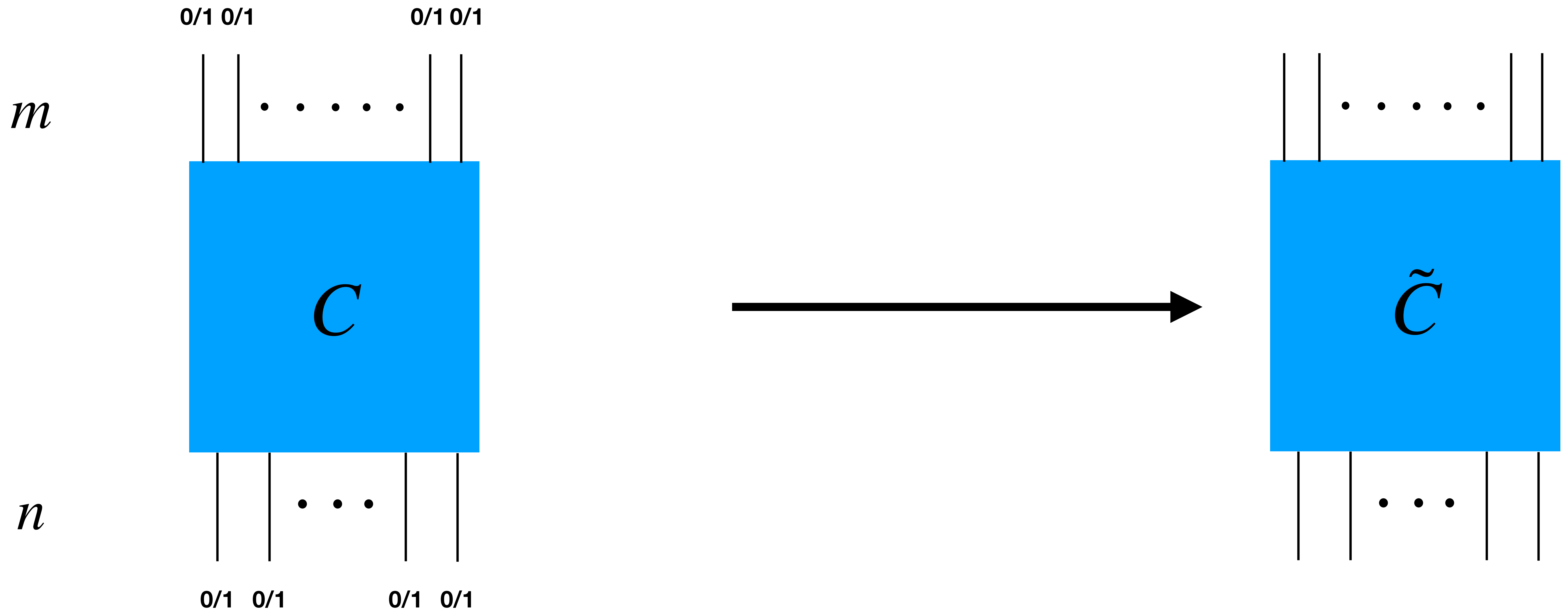
Garbling Scheme



Garbling Scheme



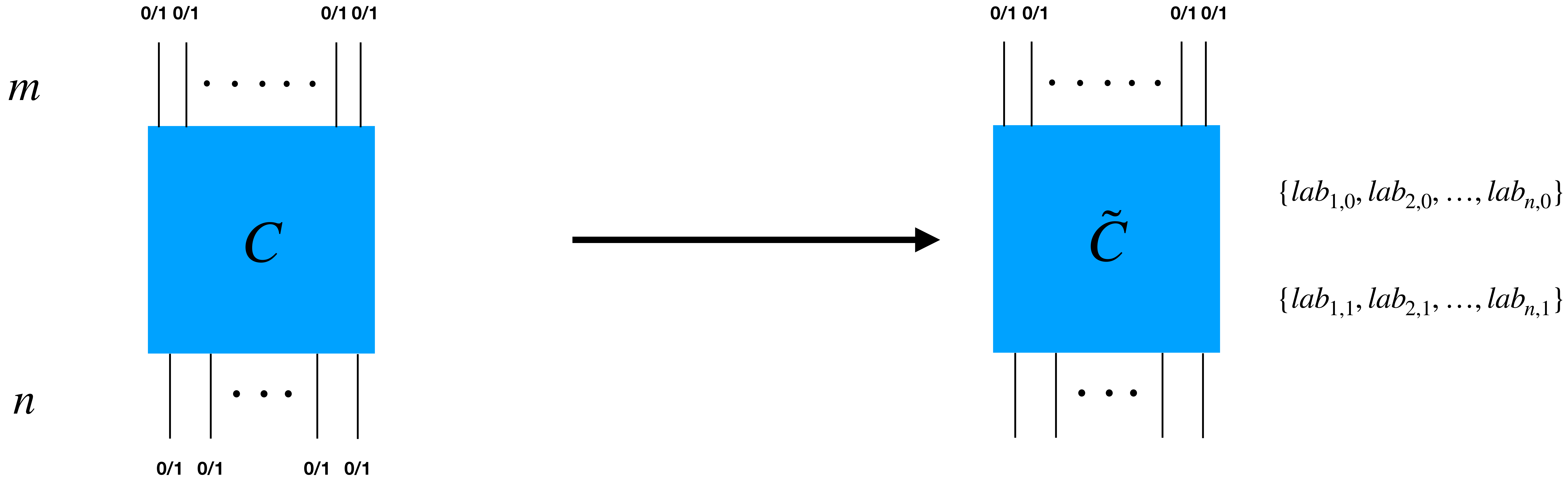
Garbling Scheme



$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$

$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$

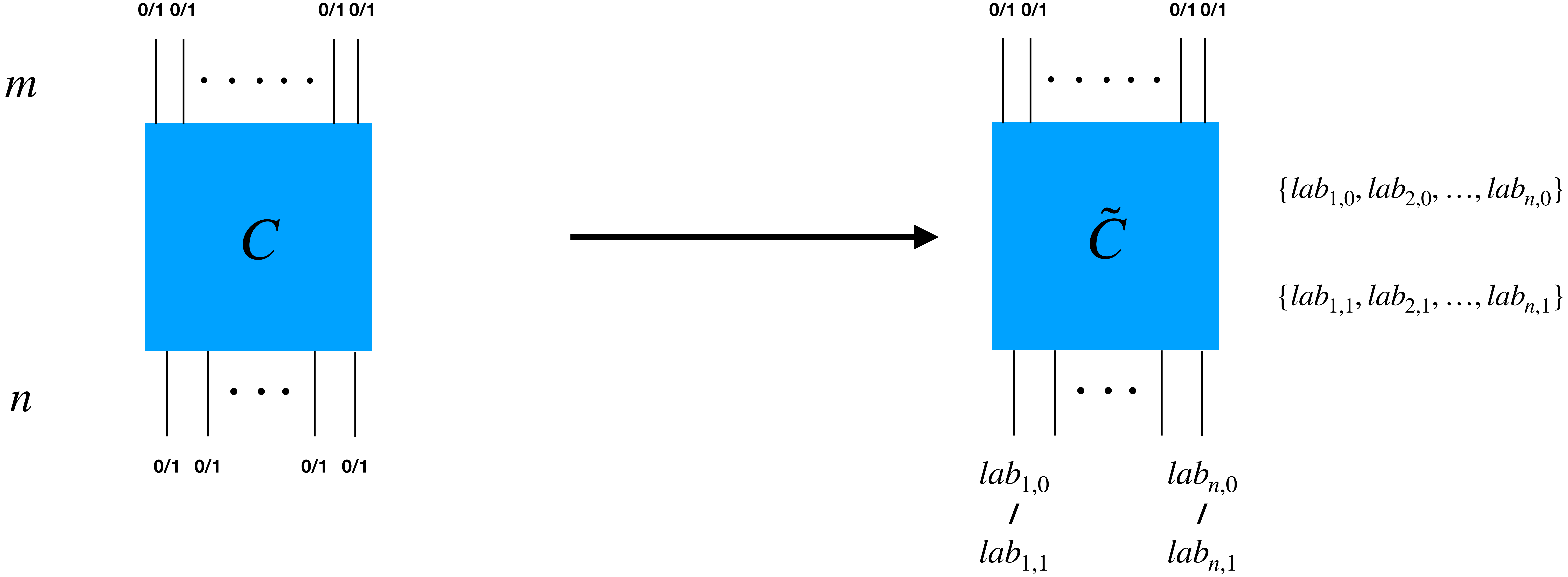
Garbling Scheme



$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$

$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$

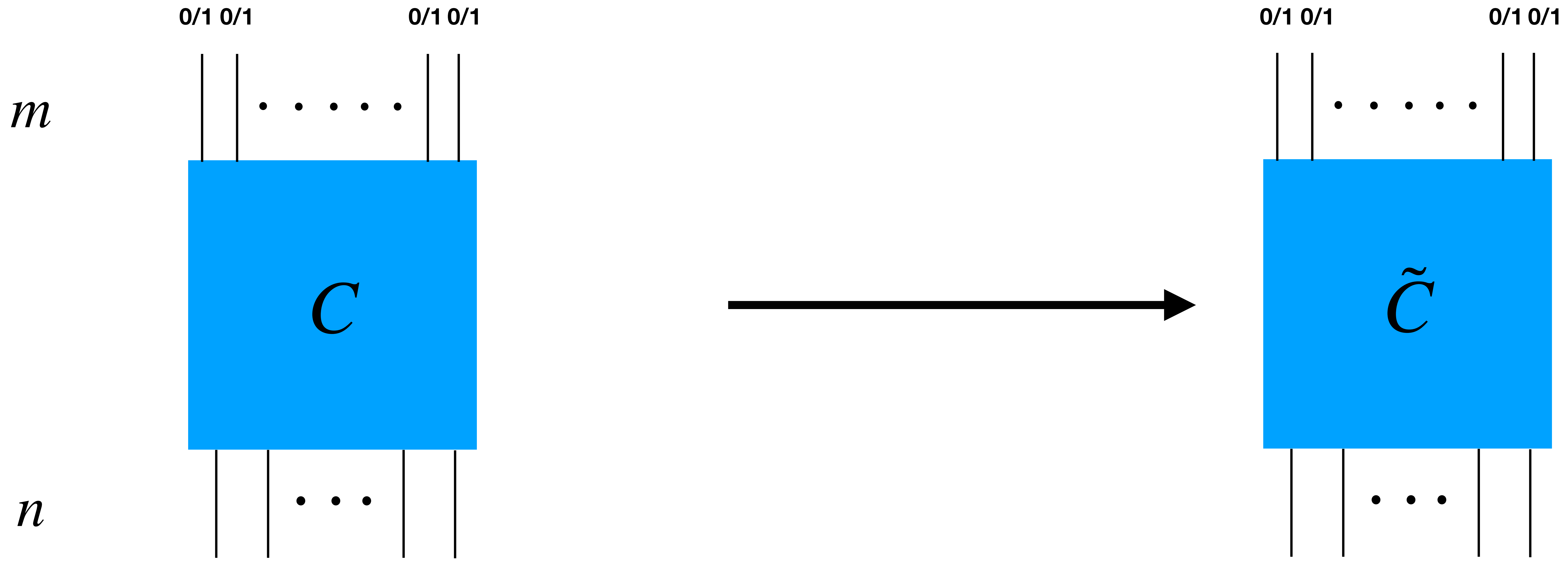
Garbling Scheme



$$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$$

$$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$$

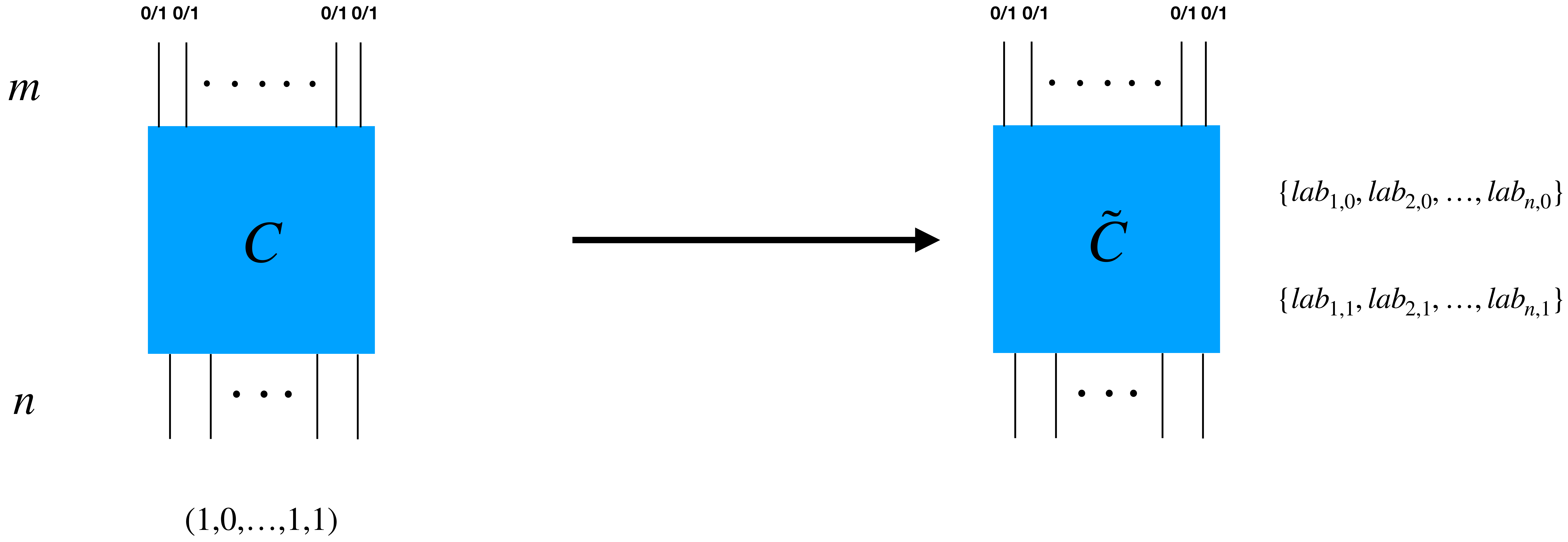
Garbling Scheme



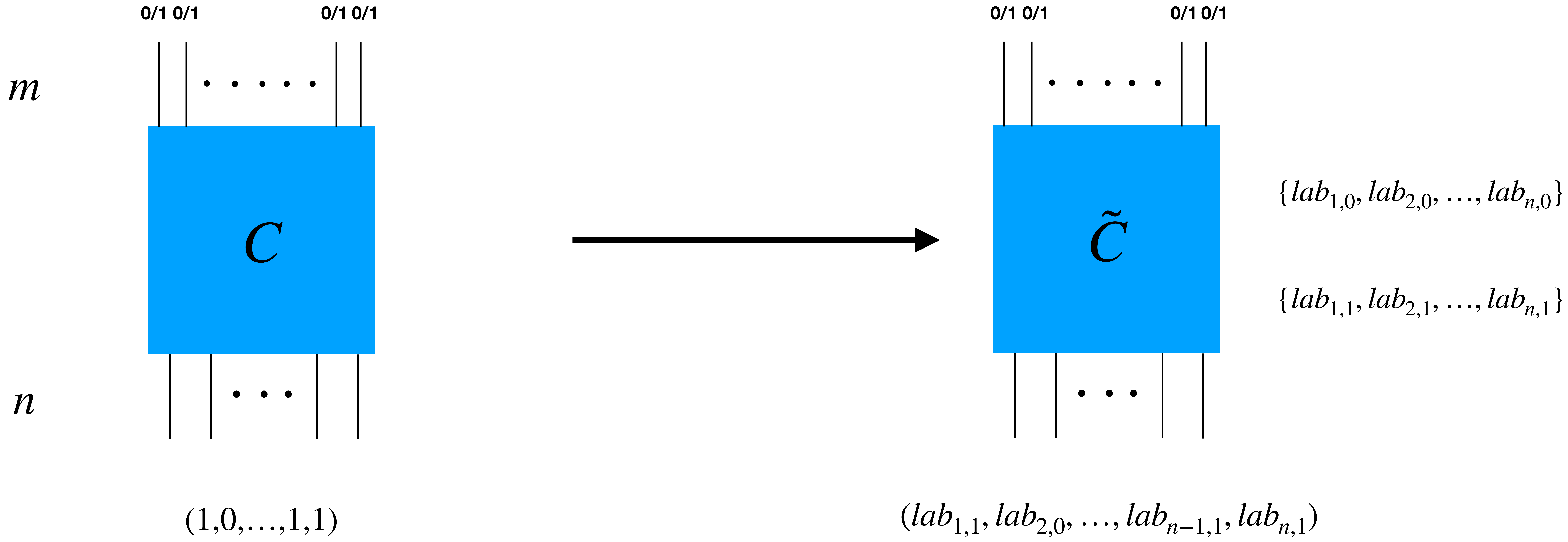
$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$

$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$

Garbling Scheme



Garbling Scheme



0/1 0/1 0/1 0/1

m

C

n

$(1,0,\ldots,1,1)$

0/1 0/1 0/1 0/1

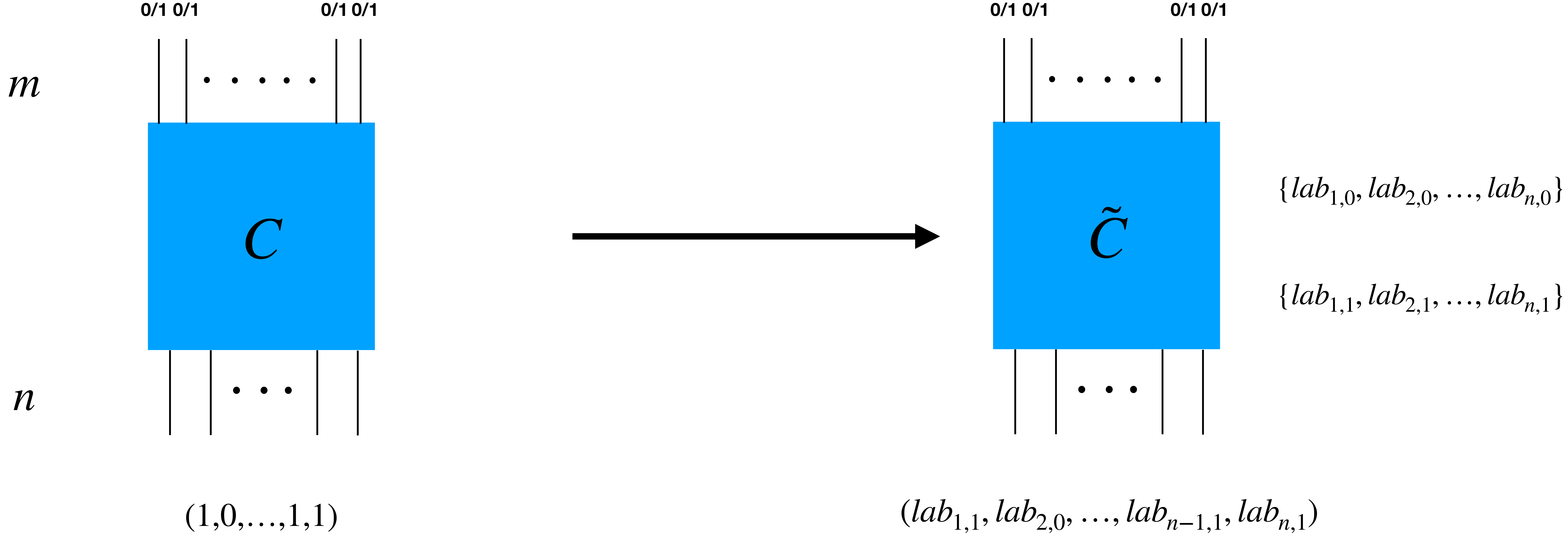
$\tilde{C}$

$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$

$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$

$(lab_{1,1}, lab_{2,0}, \ldots, lab_{n-1,1}, lab_{n,1})$

Garbling Scheme



0/1 0/1 0/1 0/1

m

C

n

$(1,0,\ldots,1,1)$

0/1 0/1 0/1 0/1

$\tilde{C}$

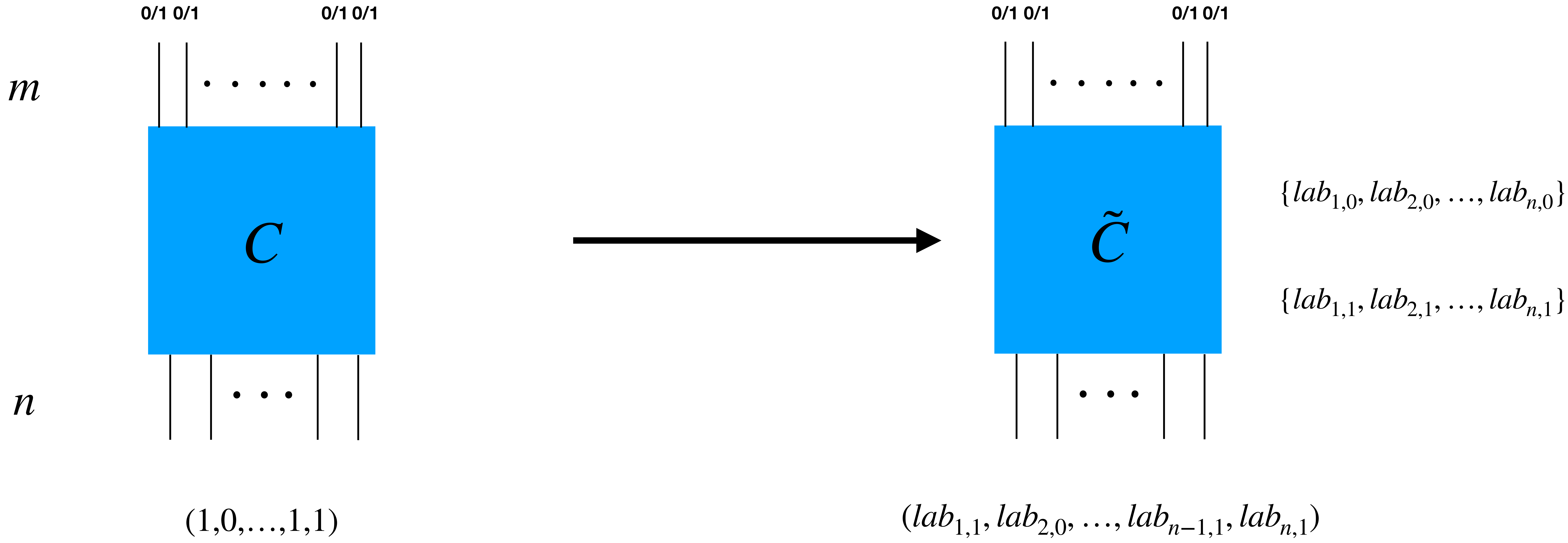
$\{lab_{1,0}, lab_{2,0}, \ldots, lab_{n,0}\}$

$\{lab_{1,1}, lab_{2,1}, \ldots, lab_{n,1}\}$

$(lab_{1,1}, lab_{2,0}, \ldots, lab_{n-1,1}, lab_{n,1})$

Correctness - For any x, $C(x) = \tilde{C}(\{lab_{i,x_i}\})$.

Garbling Scheme



Correctness - For any x, $C(x) = \tilde{C}(\{lab_{i,x_i}\})$.

Security - Given $|C|$, $C(x)$, the simulator can generate $\tilde{C}$ and $\{lab_{i,x_i}\}$

Our Construction of Incomp PKE

Our Construction of Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

Our Construction of Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,b} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,b})$

  Return $(\tilde{C}, \{c_{i,b}\})$

Our Construction of Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,b} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,b})$

  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:

  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$

  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$

  $m \leftarrow incSKE.Dec(k, incSKE.ct)$

  Return m

Correctness of our Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,b} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,b})$

  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:

  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$

  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$

  $m \leftarrow incSKE.Dec(k, incSKE.ct)$

  Return m

Correctness of our Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,b} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,b})$

  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:

  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$

  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$

  $= incSKE.Enc(k, m)$

  $m \leftarrow incSKE.Dec(k, incSKE.ct)$

  Return m

Correctness of our Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,b} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,b})$

  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:

  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$

  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$

  $= incSKE.Enc(k, m)$

  $m \leftarrow incSKE.Dec(k, incSKE.ct)$

  Return m

Security of our Incomp PKE

- $Setup()$:

  Generate $2n$ public/secret key,

  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$

  Generate $k \leftarrow incSKE.Setup()$.

  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:

  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$

  $c_{i,k_i} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,k_i})$

  $c_{i,1-k_i} \leftarrow PKE.Enc(pk_{i,b}, lab_{i,1-k_i})$

  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:

  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$

  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$

  $m \leftarrow incSKE.Dec(k, incSKE.ct)$

  Return m

Security of our Incomp PKE

- $Setup()$:
  Generate $2n$ public/secret key,
  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$
  Generate $k \leftarrow incSKE.Setup()$.
  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:
  $(\tilde{C}, lab_{i,b}) \leftarrow Garble(incSKE.Enc(\cdot, m))$
  $c_{i,k_i} \leftarrow PKE.Enc(pk_{i,k_i}, lab_{i,k_i})$
  $c_{i,1-k_i} \leftarrow PKE.Enc(pk_{i,1-k_i}, 0)$
  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:
  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$
  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$
  $m \leftarrow incSKE.Dec(k, incSKE.ct)$
  Return m

Security of our Incomp PKE

- $Setup()$:
  Generate $2n$ public/secret key,
  $$(pk_{i,b}, sk_{i,b}) \leftarrow PKE.Setup()$$
  Generate $k \leftarrow incSKE.Setup()$.
  $pk = \{pk_{i,b}\}$ and $sk = (k, \{sk_{i,k_i}\})$

- $Enc(pk, m)$:
  $(\tilde{C}, \{lab_{i,k_i}\}) \leftarrow Sim(incSKE.Enc(k, m))$
  $c_{i,k_i} \leftarrow PKE.Enc(pk_{i,k_i}, lab_{i,k_i})$
  $c_{i,1-k_i} \leftarrow PKE.Enc(pk_{i,1-k_i}, 0)$
  Return $(\tilde{C}, \{c_{i,b}\})$

- $Dec(sk, (\tilde{C}, \{c_{i,b}\}))$:
  $lab_{i,k_i} \leftarrow PKE.Dec(sk_{i,k_i}, c_{i,k_i})$
  $incSKE.ct = \tilde{C}(\{lab_{i,k_i}\})$
  $m \leftarrow incSKE.Dec(k, incSKE.ct)$
  Return m

Incompressible IBE & FE

Identity Based Encryption

Identity Based Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).

Identity Based Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m, id)$: Outputs ciphertext c.

Identity Based Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m, id)$: Outputs ciphertext c.
- $KeyGen(msk, id)$: Outputs secret key sk_{id}.

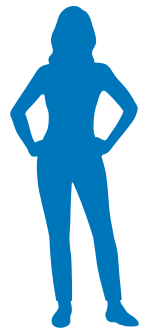
Identity Based Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m, id)$: Outputs ciphertext c.
- $KeyGen(msk, id)$: Outputs secret key sk_{id}.
- $Dec(sk_{id}, c)$: Outputs a message or error.

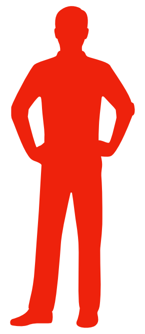
Incompressible (IBE) Security

Incompressible (IBE) Security

Incompressible (IBE) Security

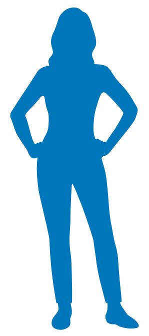


Challenger

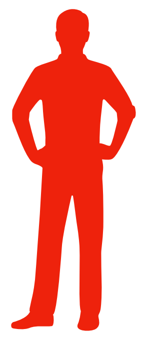


Adversary 1

Incompressible (IBE) Security



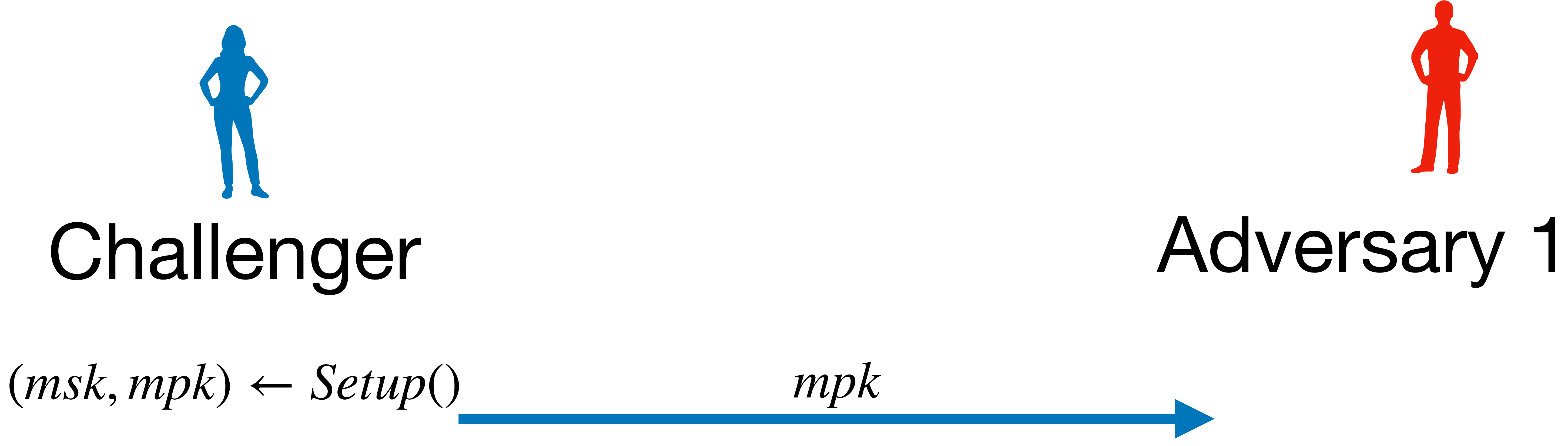
Challenger



Adversary 1

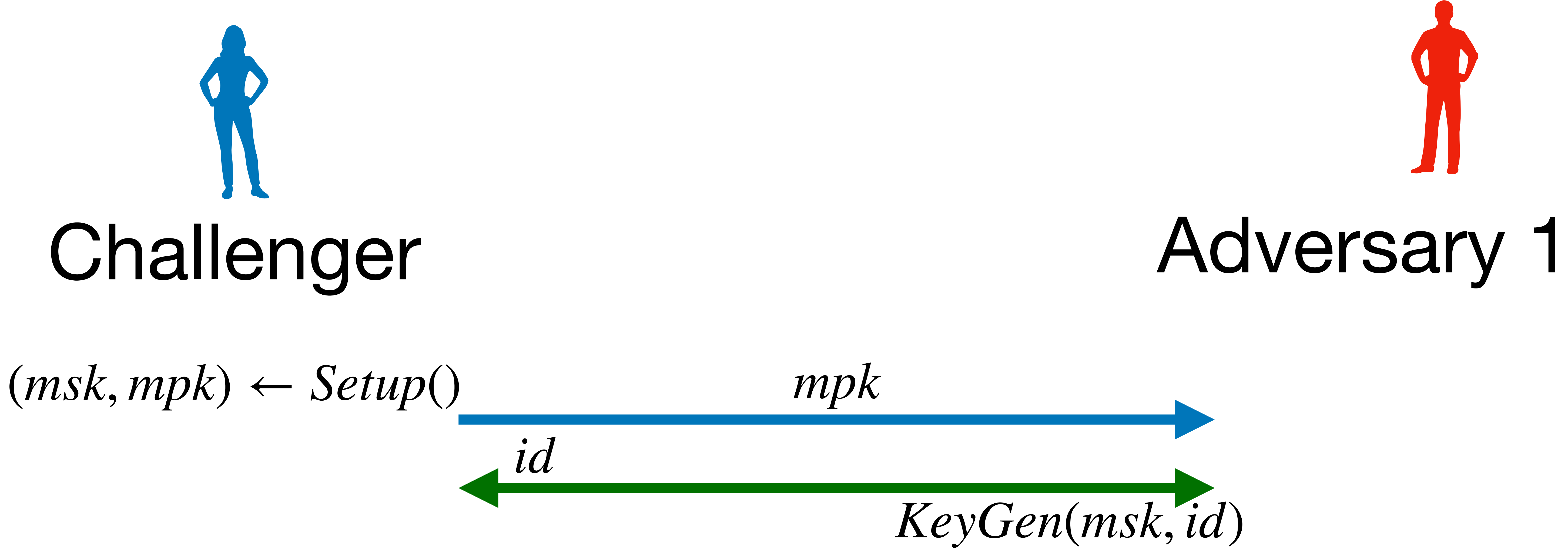
$$(msk, mpk) \leftarrow Setup()$$

Incompressible (IBE) Security



$(msk, mpk) \leftarrow Setup()$ $\xrightarrow{mpk}$

Incompressible (IBE) Security



Challenger

Adversary 1

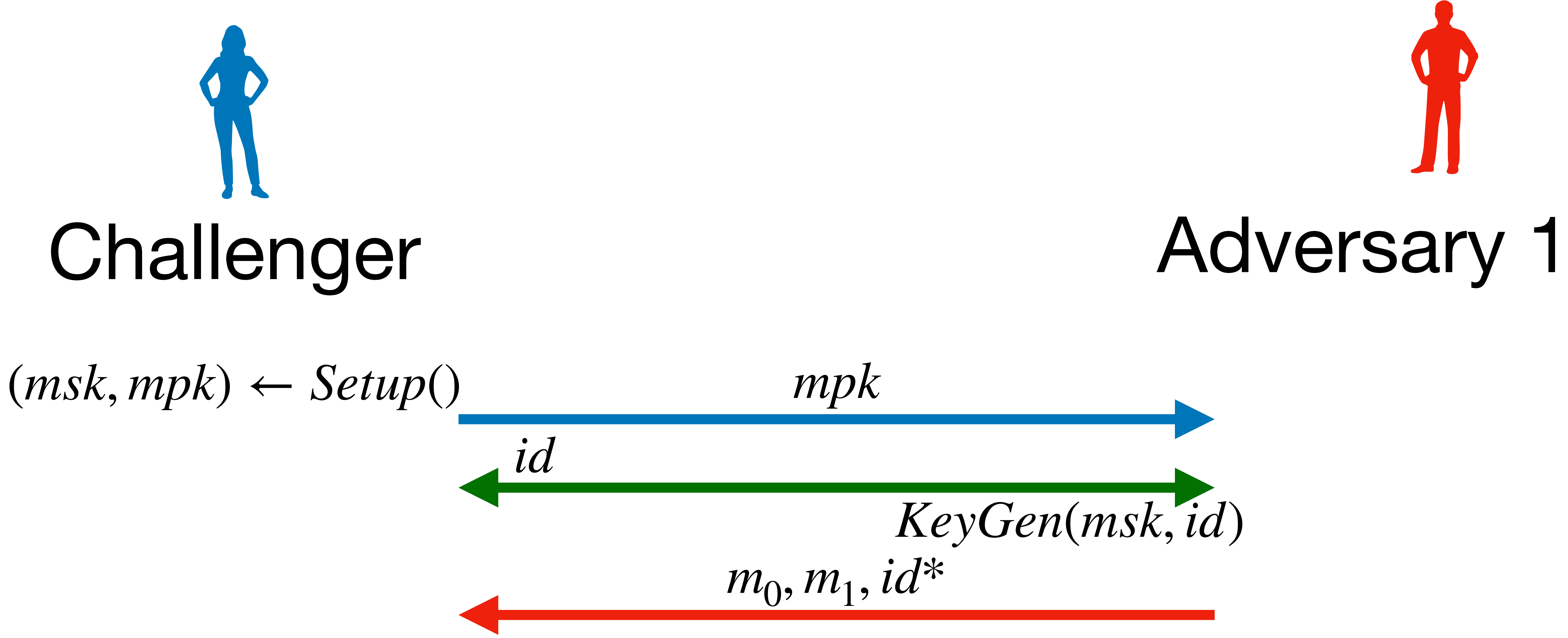
$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

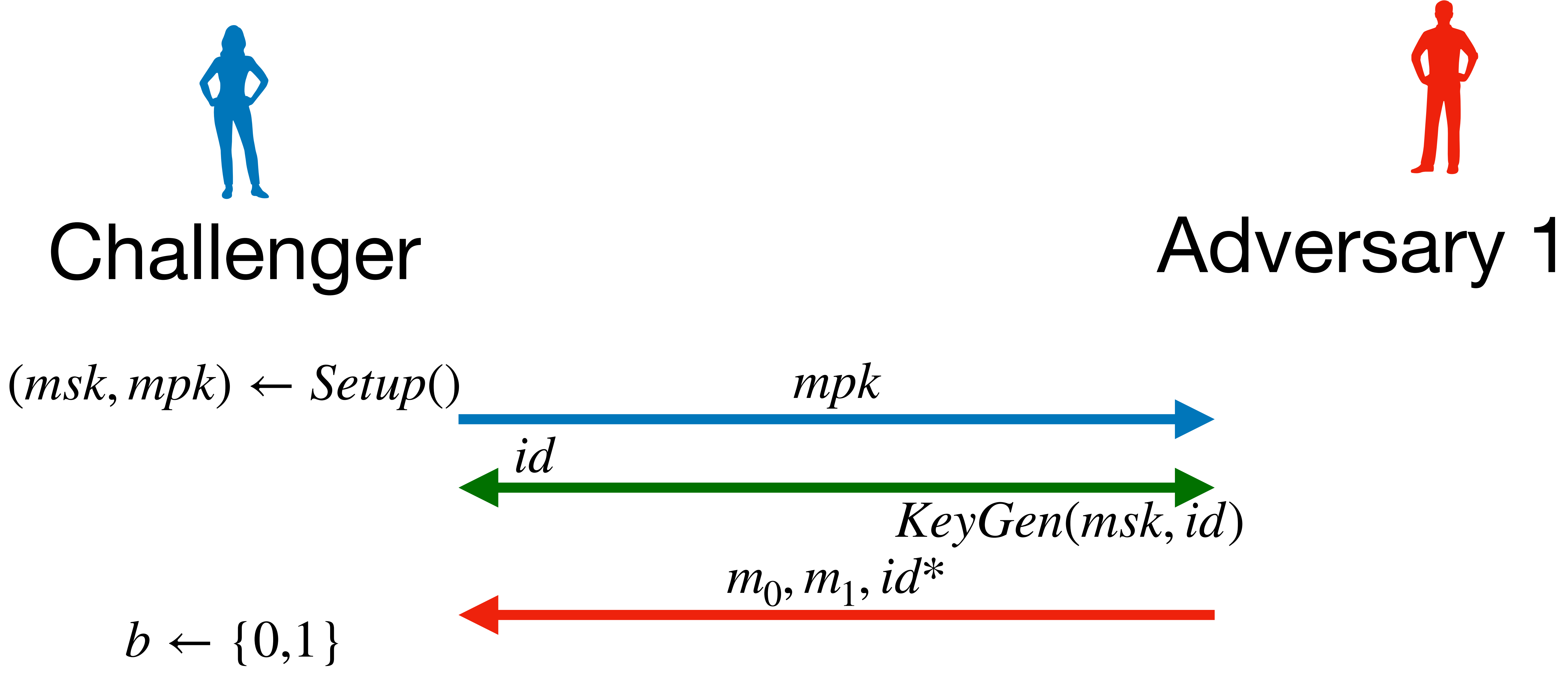
mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

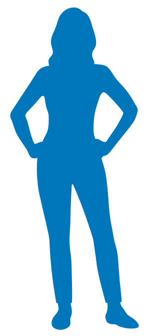
id

$KeyGen(msk, id)$

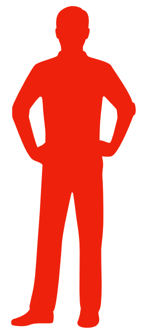
m_0, m_1, id^*

$b \leftarrow \{0,1\}$

Incompressible (IBE) Security



Challenger



Adversary 1

$(msk, mpk) \leftarrow Setup()$ $\xrightarrow{mpk}$

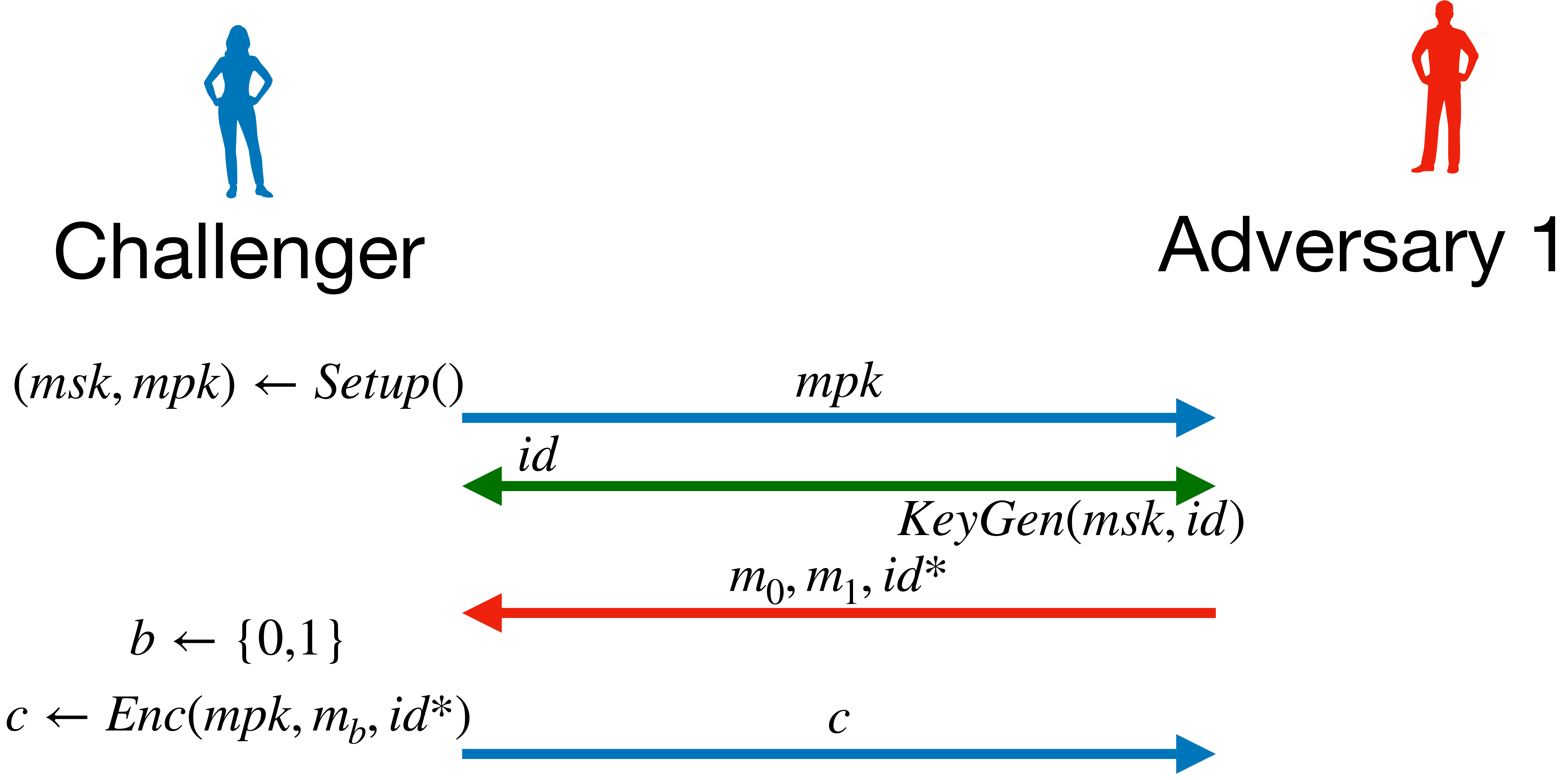
$\xleftarrow{id}$ $\xrightarrow{KeyGen(msk, id)}$

$\xleftarrow{m_0, m_1, id^*}$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$ $\xrightarrow{mpk}$

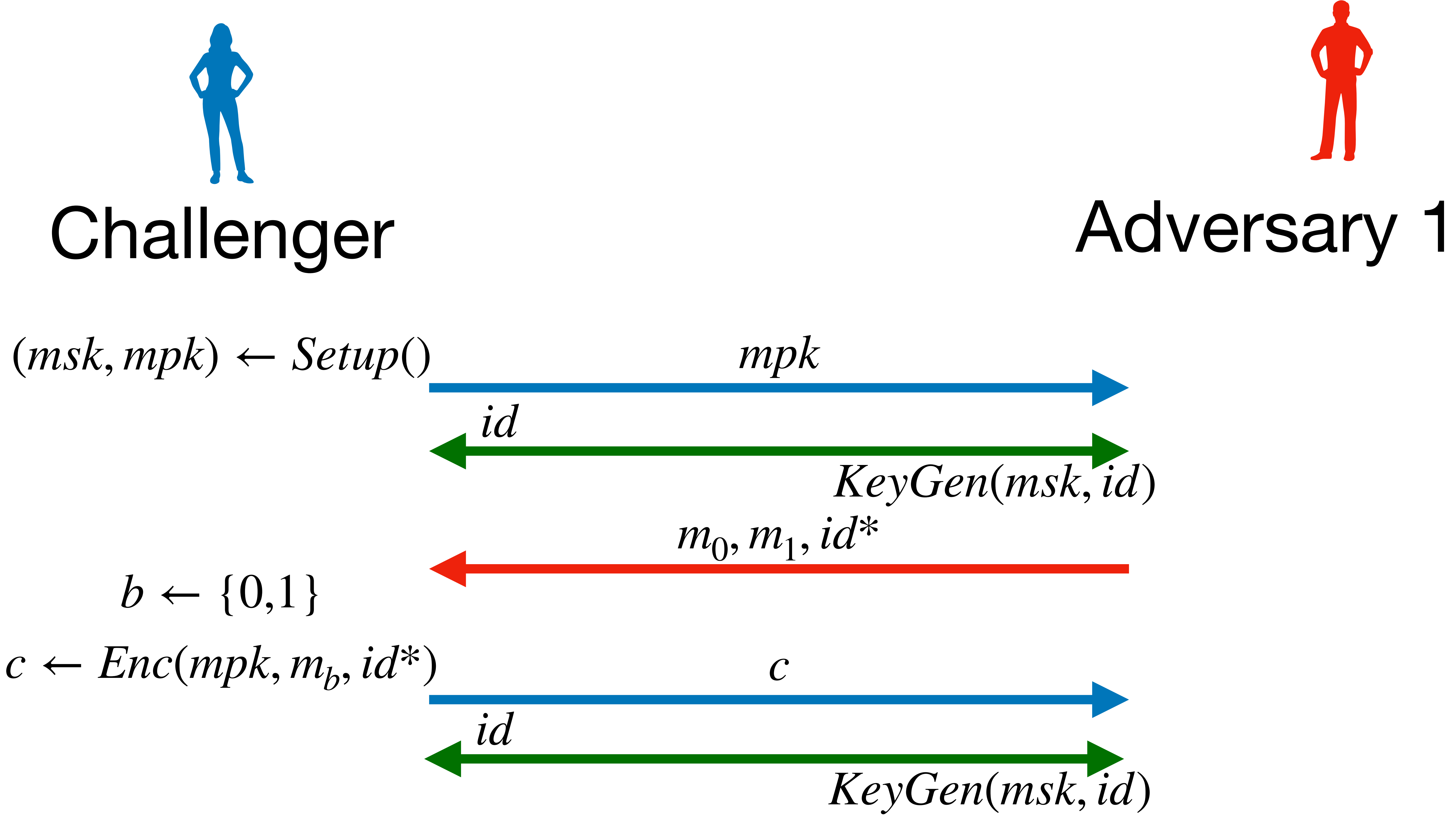
$\xleftarrow{id}$ $\xrightarrow{KeyGen(msk, id)}$

$\xleftarrow{m_0, m_1, id*}$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id*)$ $\xrightarrow{c}$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

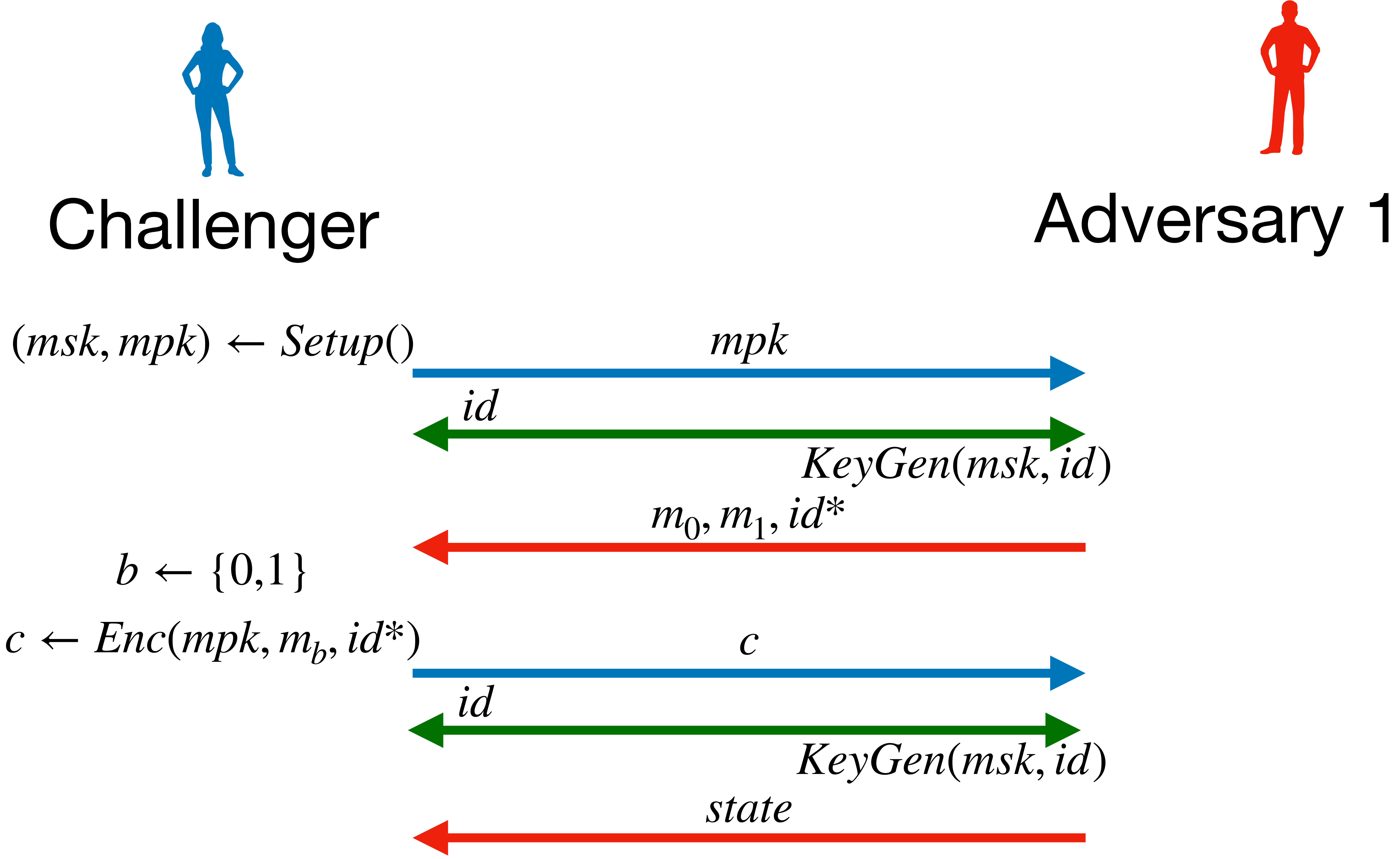
$c \leftarrow Enc(mpk, m_b, id^*)$

c

id

$KeyGen(msk, id)$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

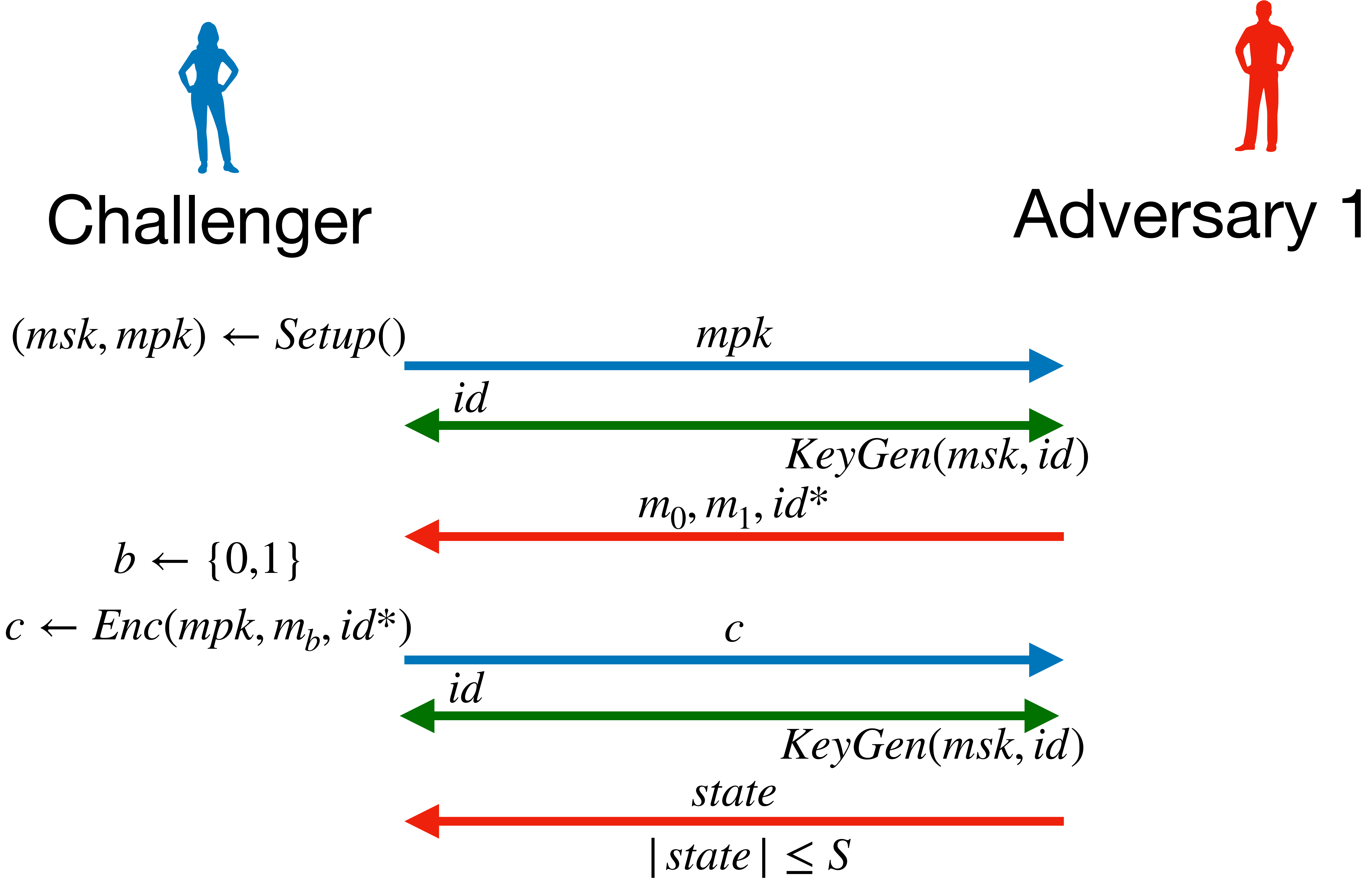
c

id

$KeyGen(msk, id)$

$state$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

c

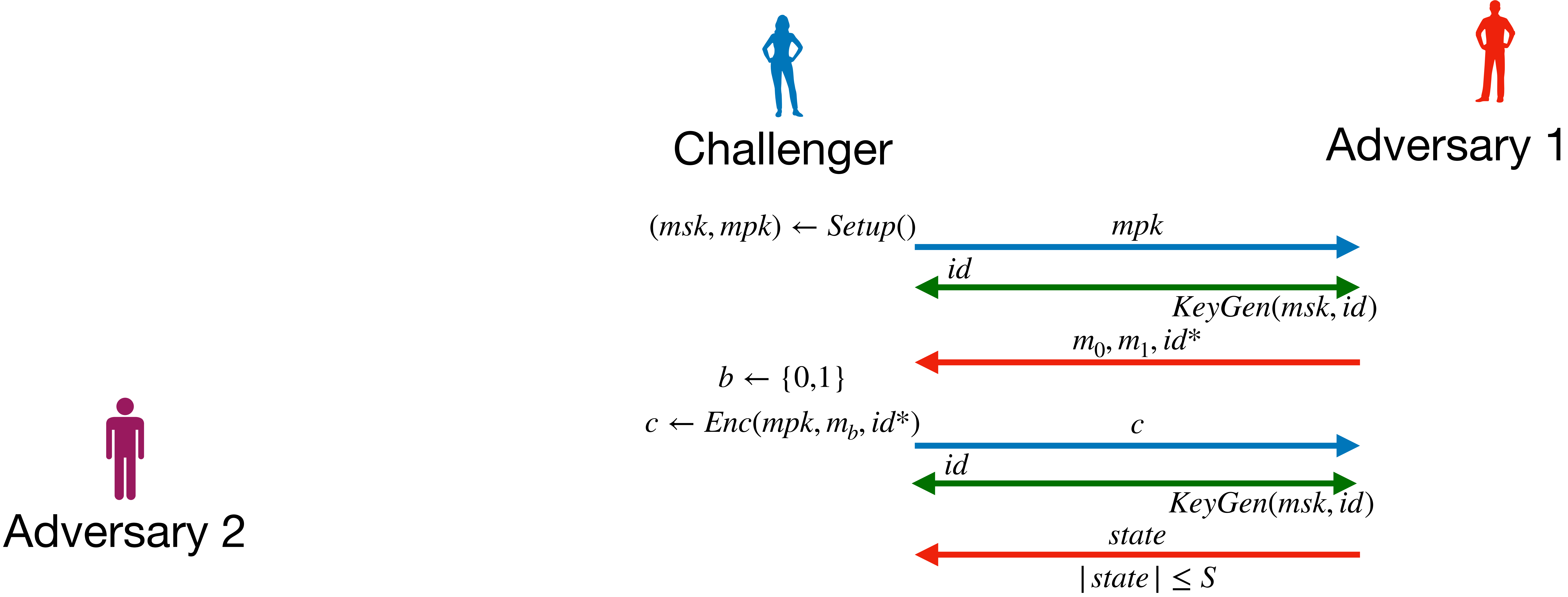
id

$KeyGen(msk, id)$

$state$

$|state| \leq S$

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

$m_0, m_1, id*$

$b \leftarrow \{0{,}1\}$

$c \leftarrow Enc(mpk, m_b, id*)$

c

id

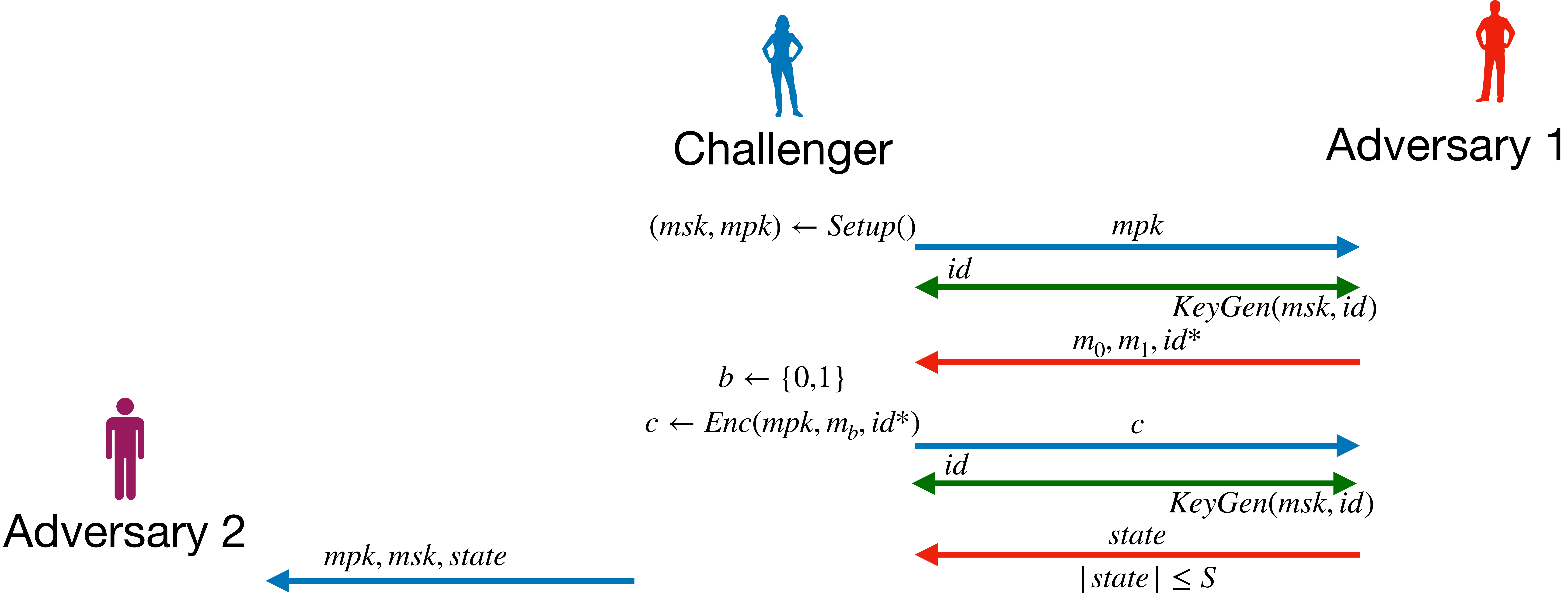
$KeyGen(msk, id)$

$state$

$|state| \leq S$

Adversary 2

Incompressible (IBE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

c

id

$KeyGen(msk, id)$

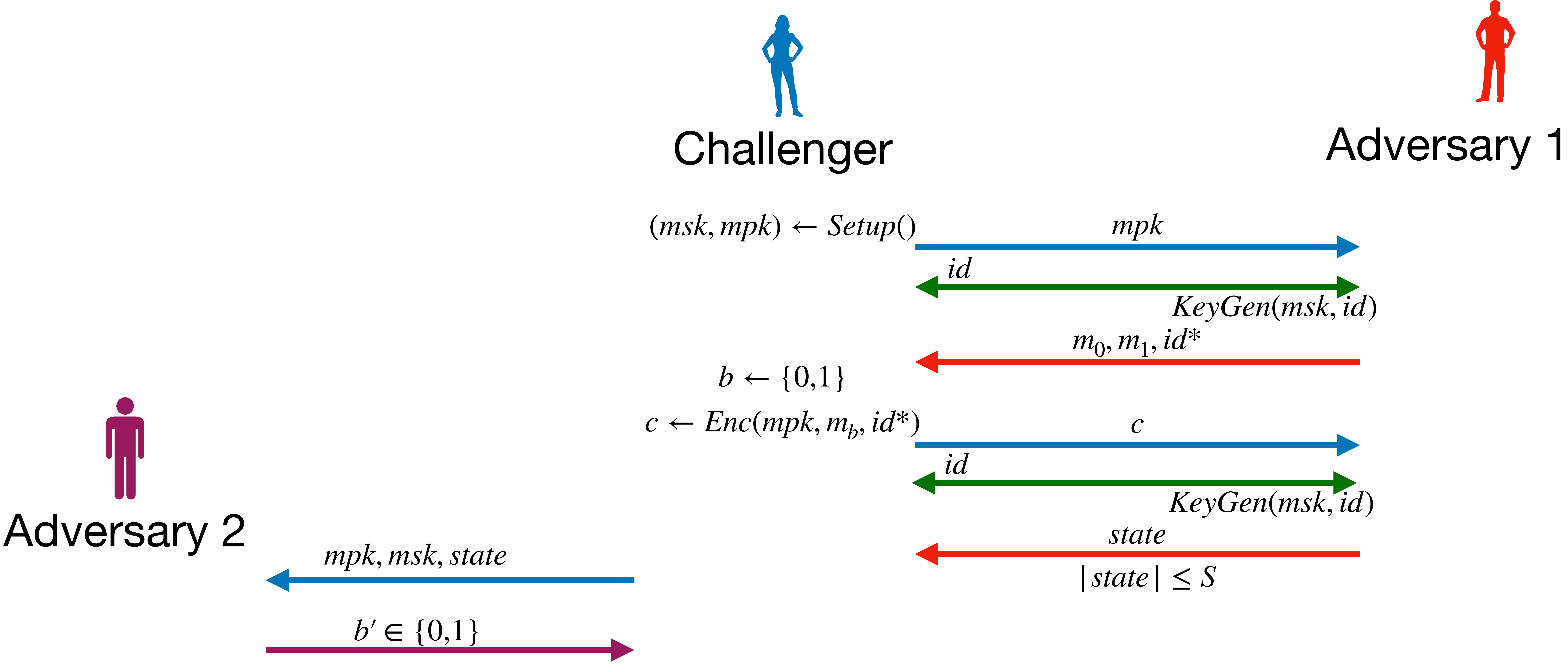
$state$

$|state| \leq S$

Adversary 2

$mpk, msk, state$

Incompressible (IBE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

c

id

$KeyGen(msk, id)$

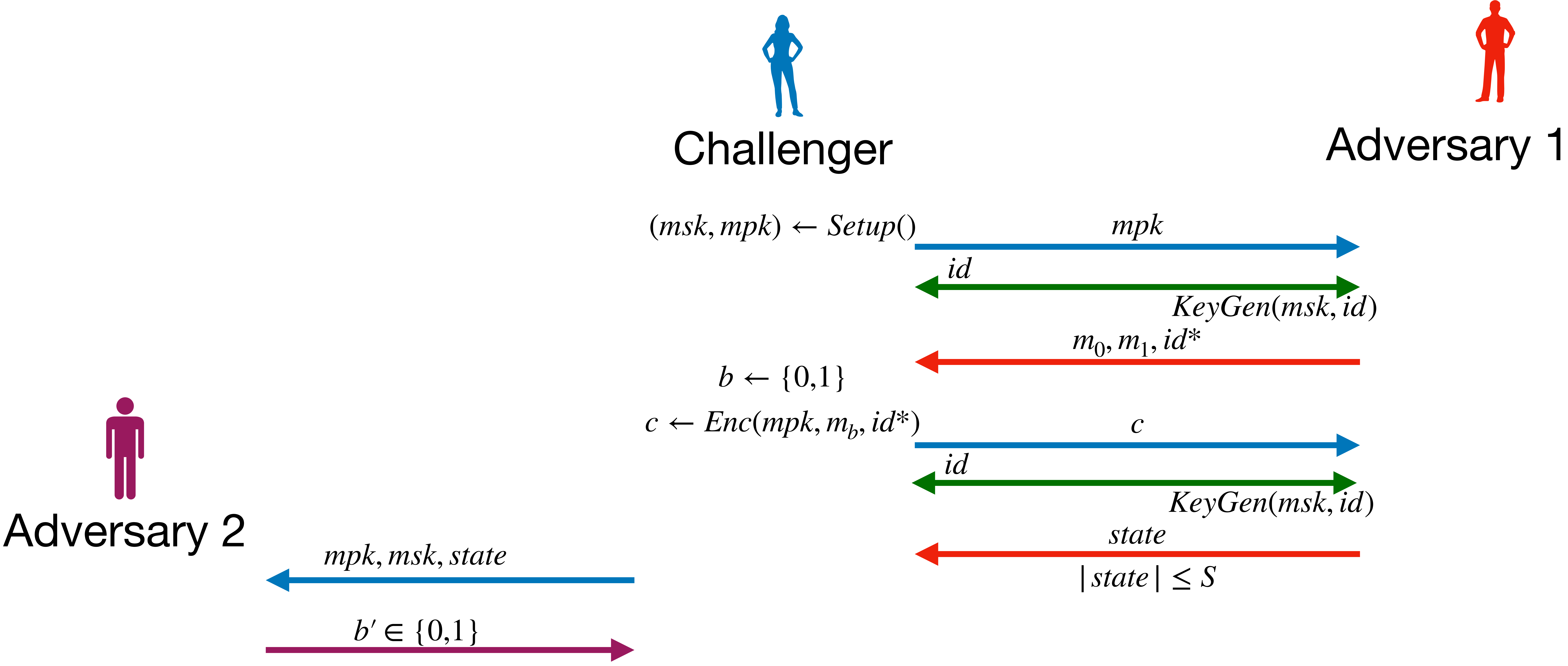
$state$

$|state| \leq S$

$mpk, msk, state$

$b' \in \{0,1\}$

Incompressible (IBE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

id

$KeyGen(msk, id)$

m_0, m_1, id^*

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b, id^*)$

c

id

$KeyGen(msk, id)$

$state$

$|state| \leq S$

$mpk, msk, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

Our Results

Our Results

- Gave an incompressible IBE scheme where second adversary gets sk_{id*}, i.e., the secret key for the target identity.

Our Results

- Gave an incompressible IBE scheme where second adversary gets sk_{id*}, i.e., the secret key for the target identity.

- Replace the PKE in the incompressible PKE construction with IBE.

Functional Encryption

Functional Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).

Functional Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m)$: Outputs ciphertext c.

Functional Encryption

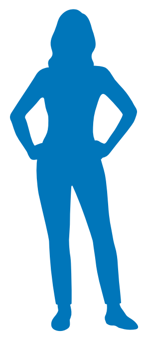
- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m)$: Outputs ciphertext c.
- $KeyGen(msk, f)$: Outputs secret key sk_f.

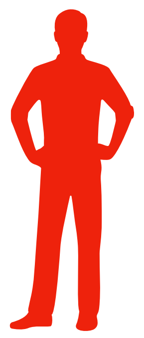
Functional Encryption

- $Setup()$: Outputs master public and secret key (mpk, msk).
- $Enc(mpk, m)$: Outputs ciphertext c.
- $KeyGen(msk, f)$: Outputs secret key sk_f.
- $Dec(sk_f, c)$: Outputs $f(m)$ or error.

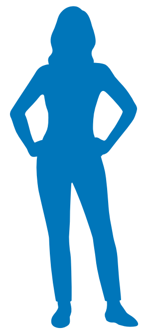
FE Security

FE Security

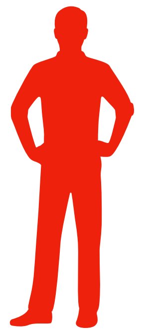




FE Security

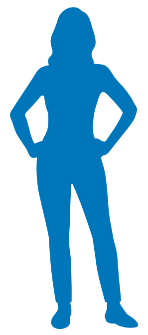


Challenger

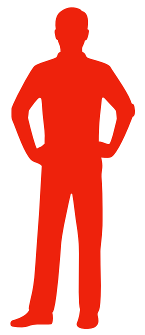


Adversary

FE Security



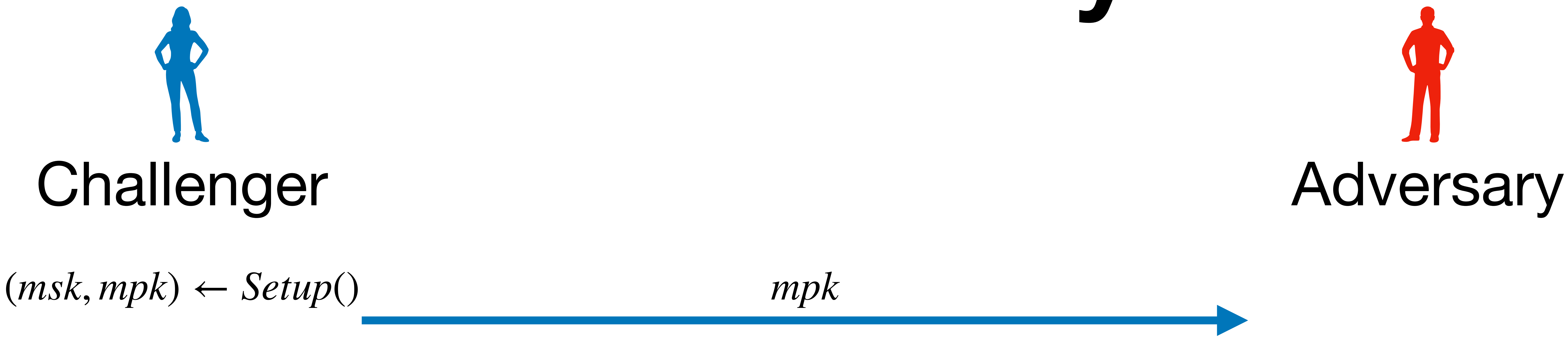
Challenger



Adversary

$(msk, mpk) \leftarrow Setup()$

FE Security



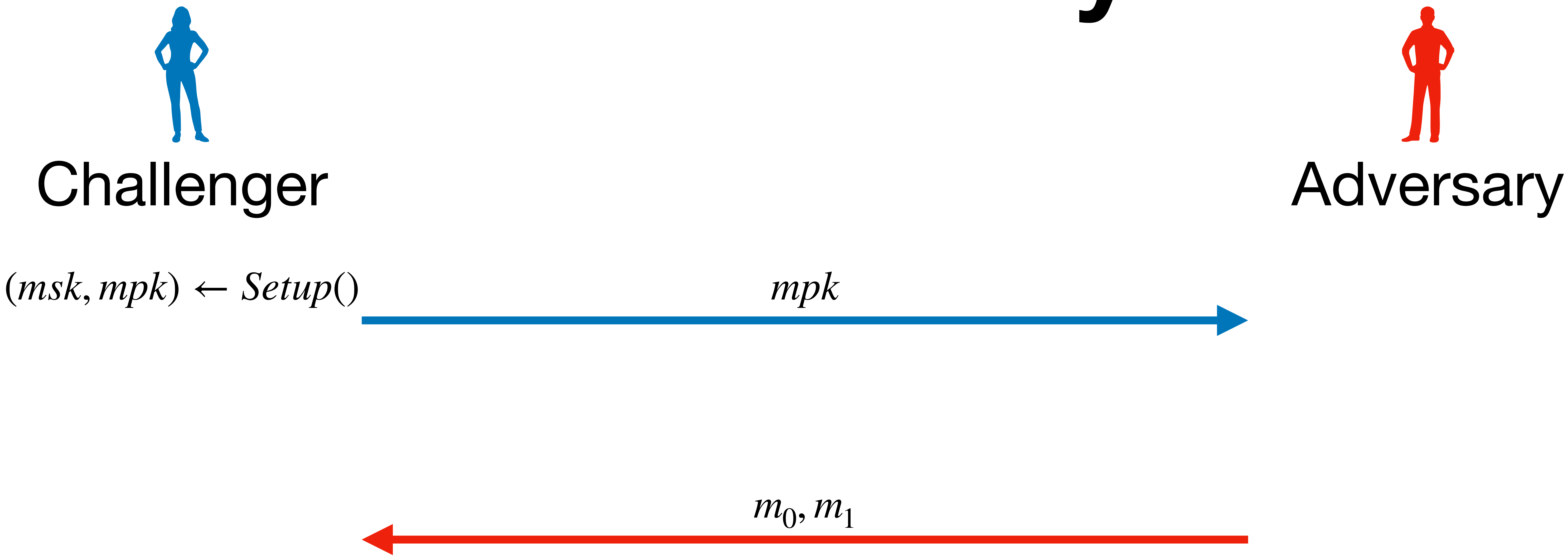
Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

FE Security



Challenger

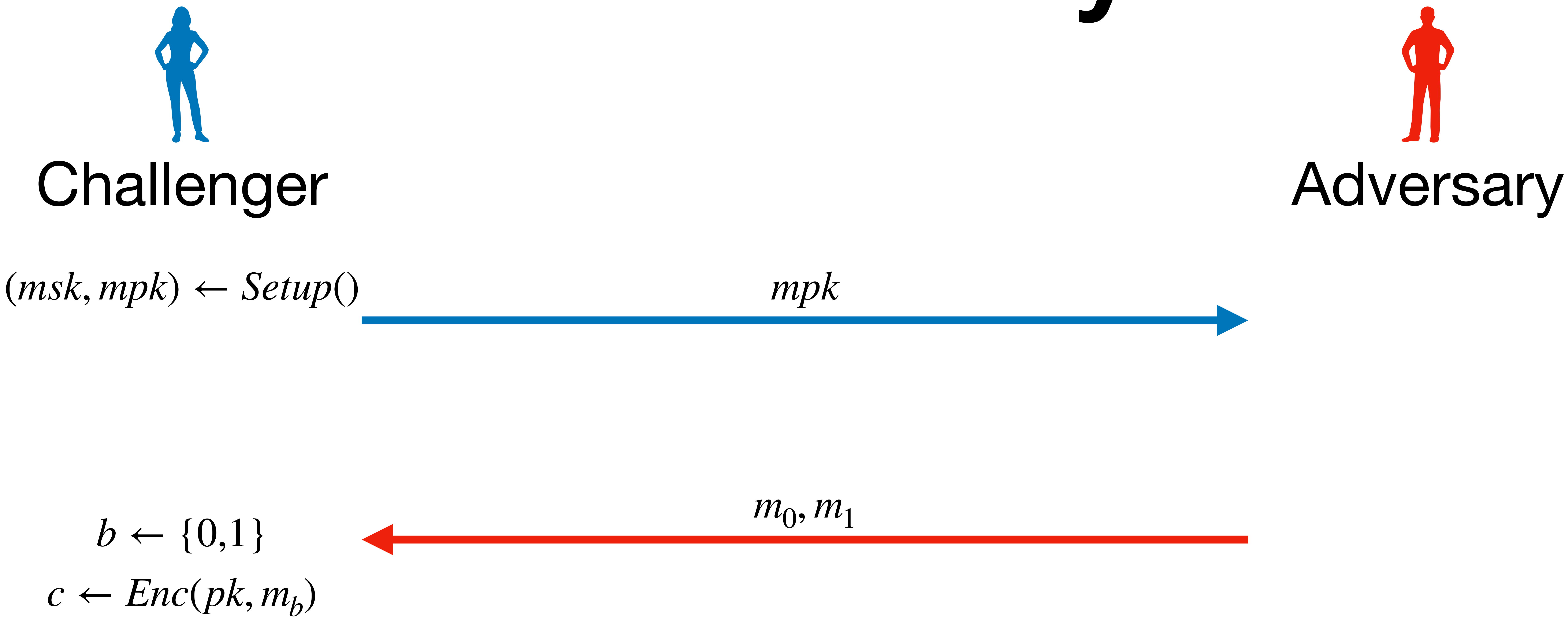
Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

m_0, m_1

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

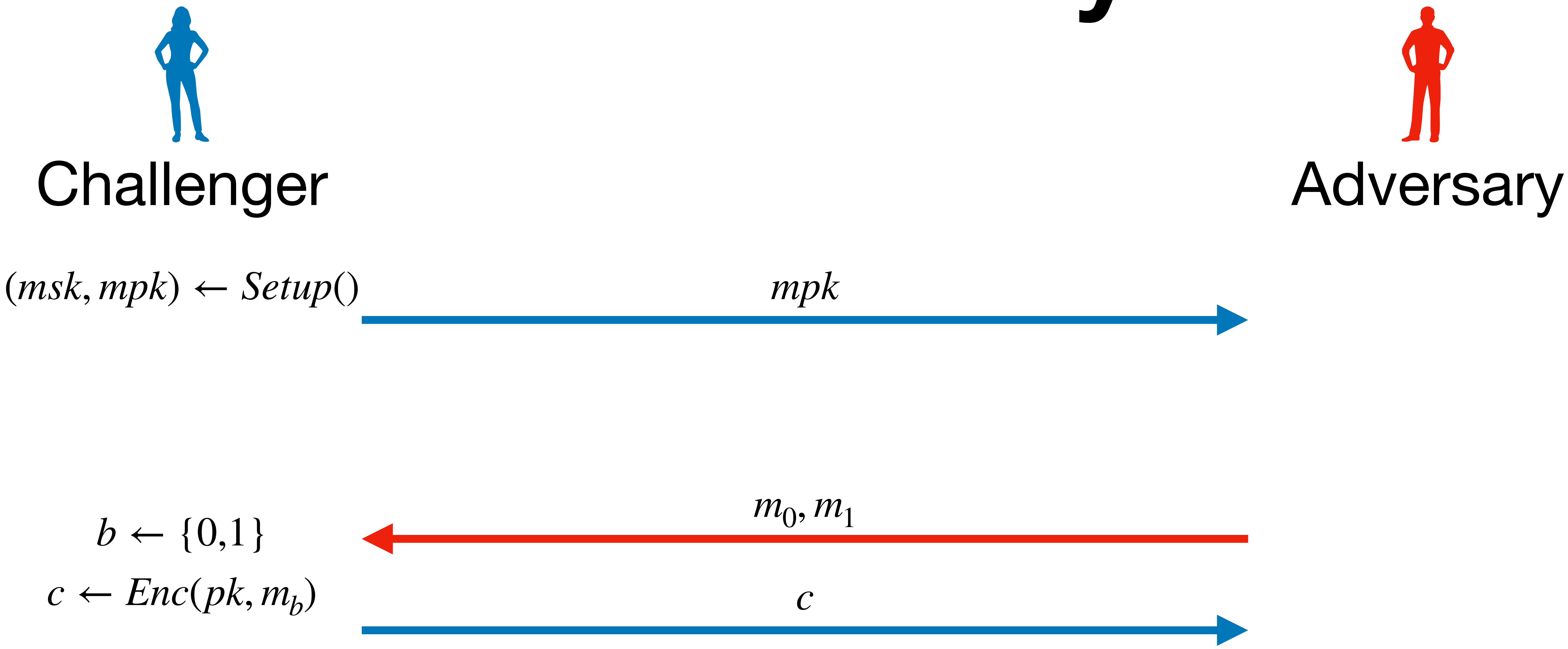
mpk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

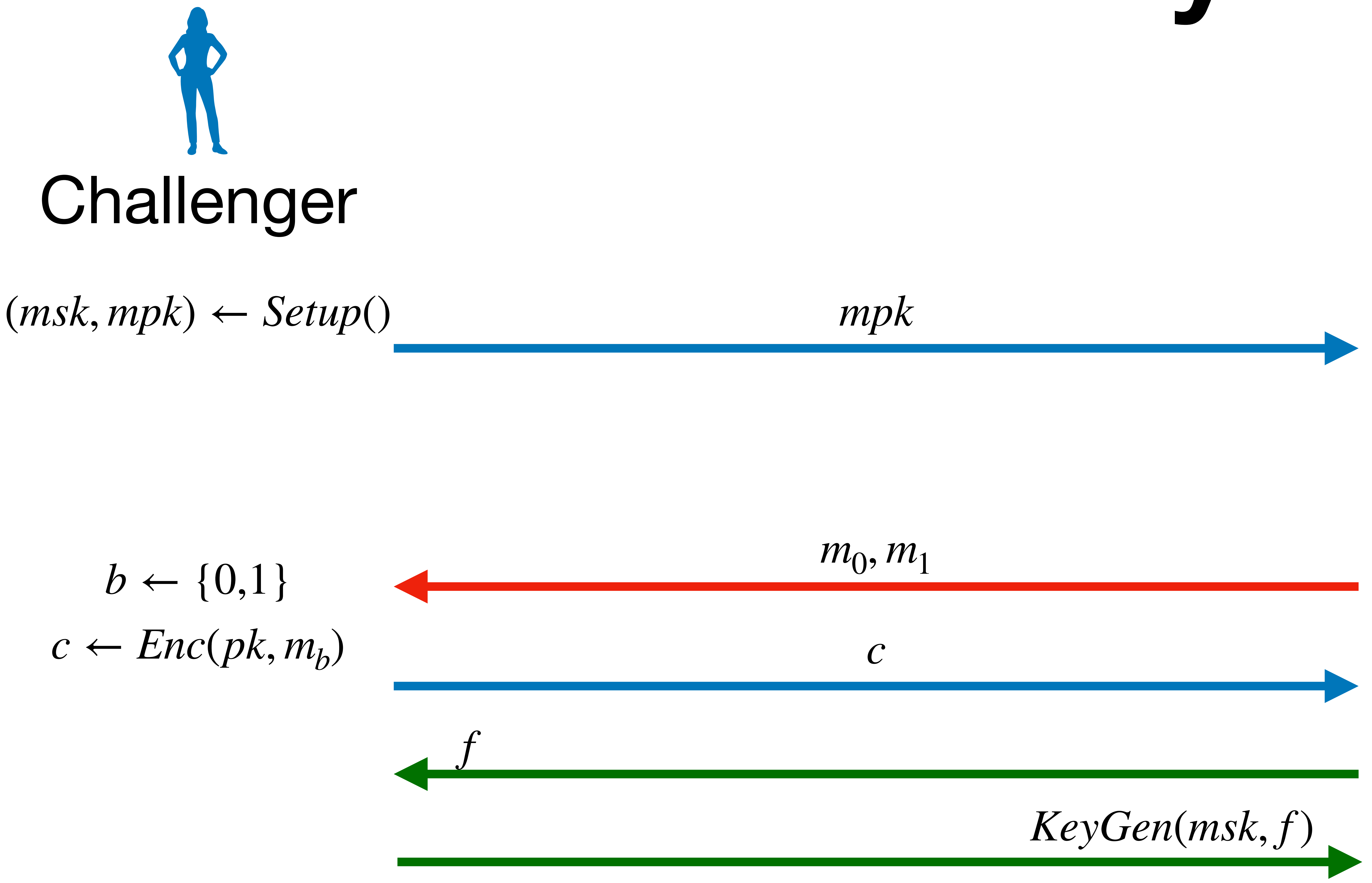
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

m_0, m_1

$b \leftarrow \{0,1\}$

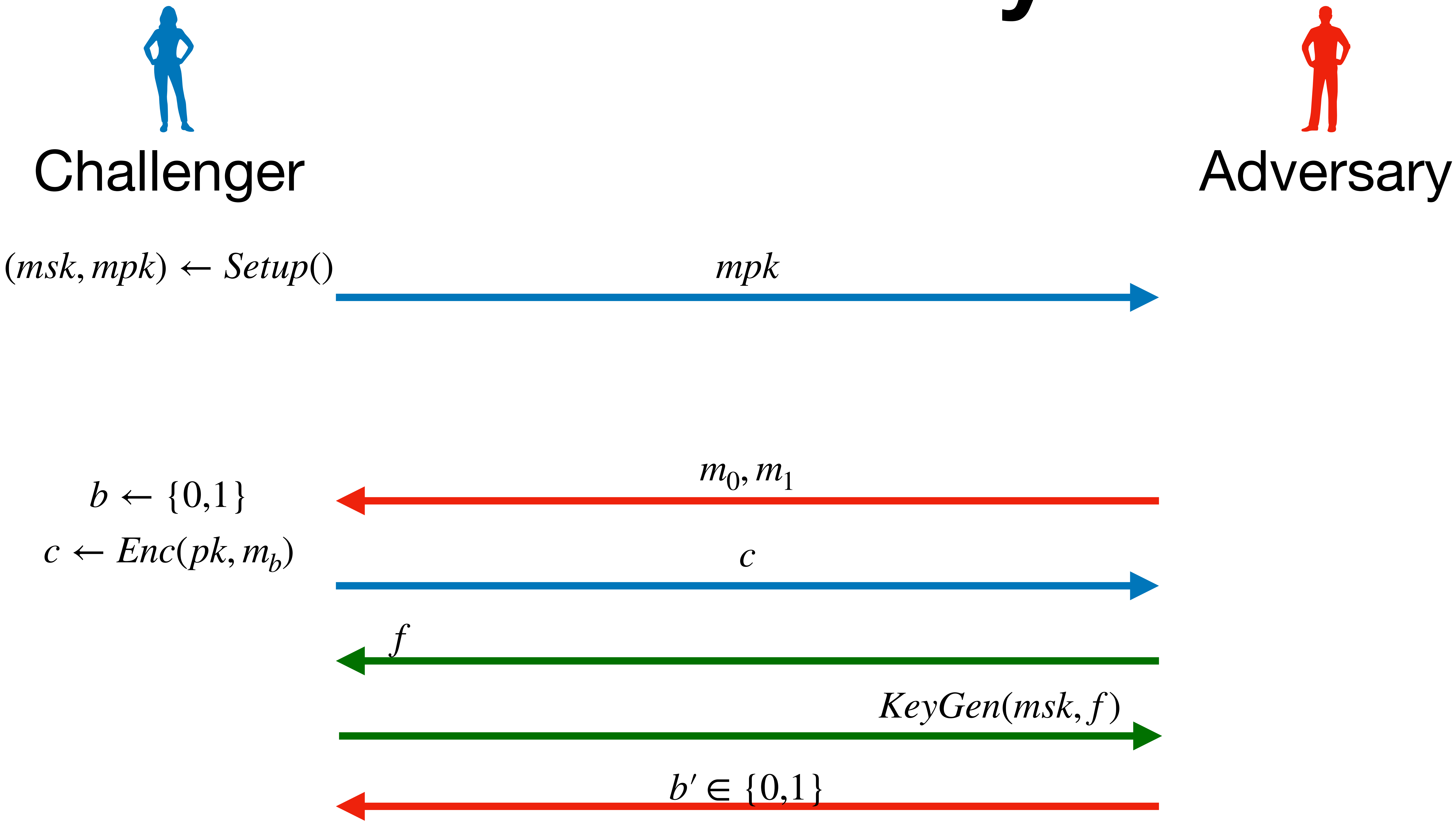
$c \leftarrow Enc(pk, m_b)$

c

f

$KeyGen(msk, f)$

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

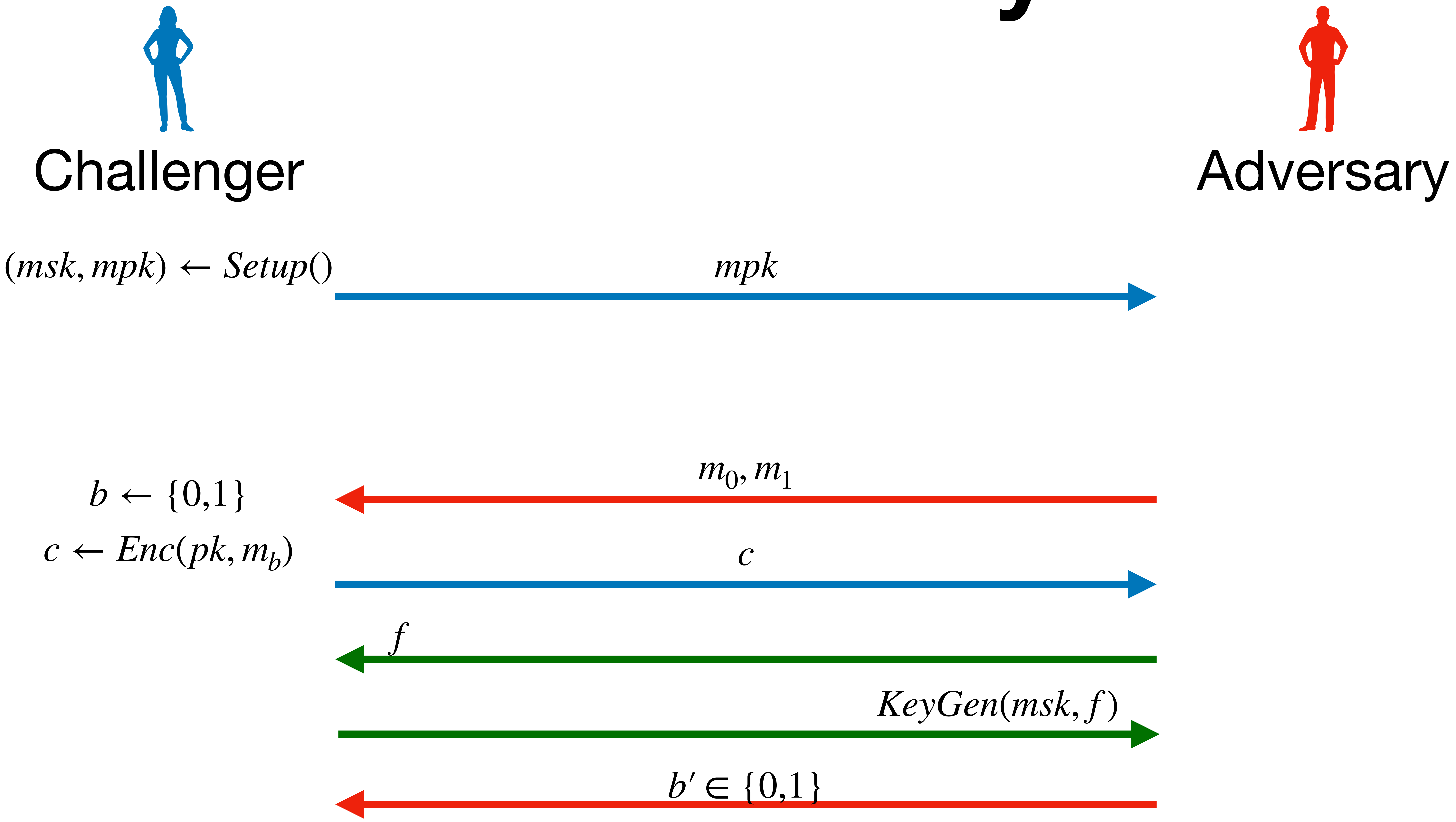
c

f

$KeyGen(msk, f)$

$b' \in \{0,1\}$

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

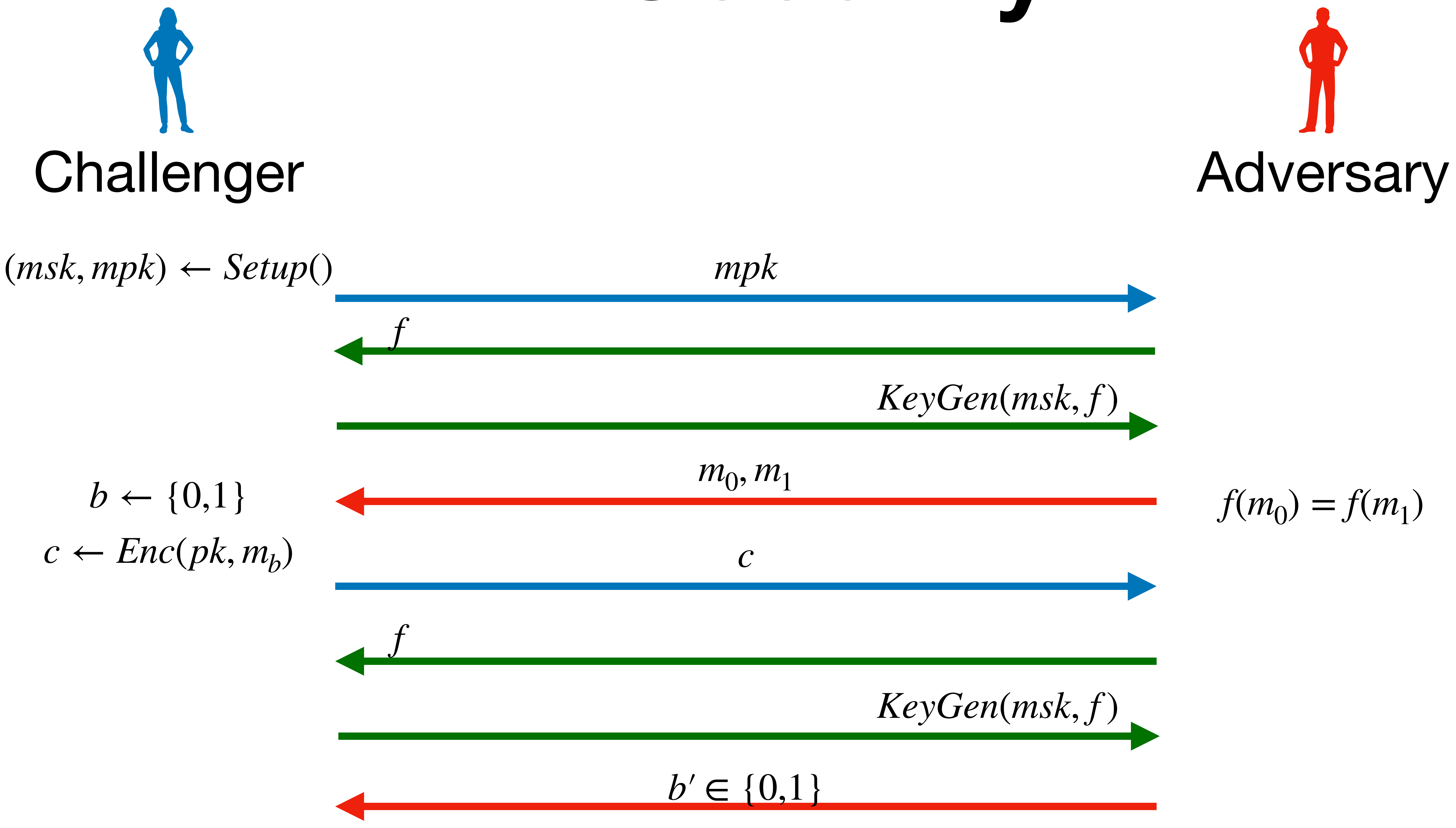
f

$KeyGen(msk, f)$

$b' \in \{0,1\}$

Adversary wins if $b = b'$

FE Security



Challenger

Adversary

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$f(m_0) = f(m_1)$

$c \leftarrow Enc(pk, m_b)$

c

f

$KeyGen(msk, f)$

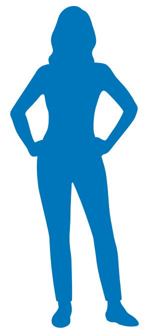
$b' \in \{0,1\}$

Adversary wins if $b = b'$

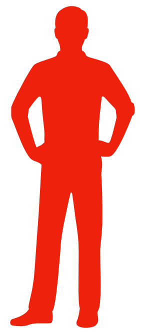
Strong Incompressible (FE) Security

Strong Incompressible (FE) Security

Strong Incompressible (FE) Security

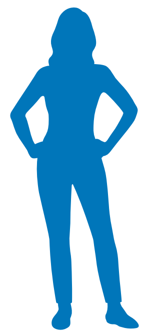


Challenger

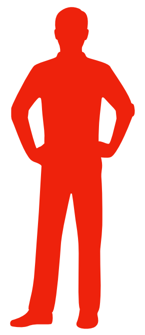


Adversary 1

Strong Incompressible (FE) Security



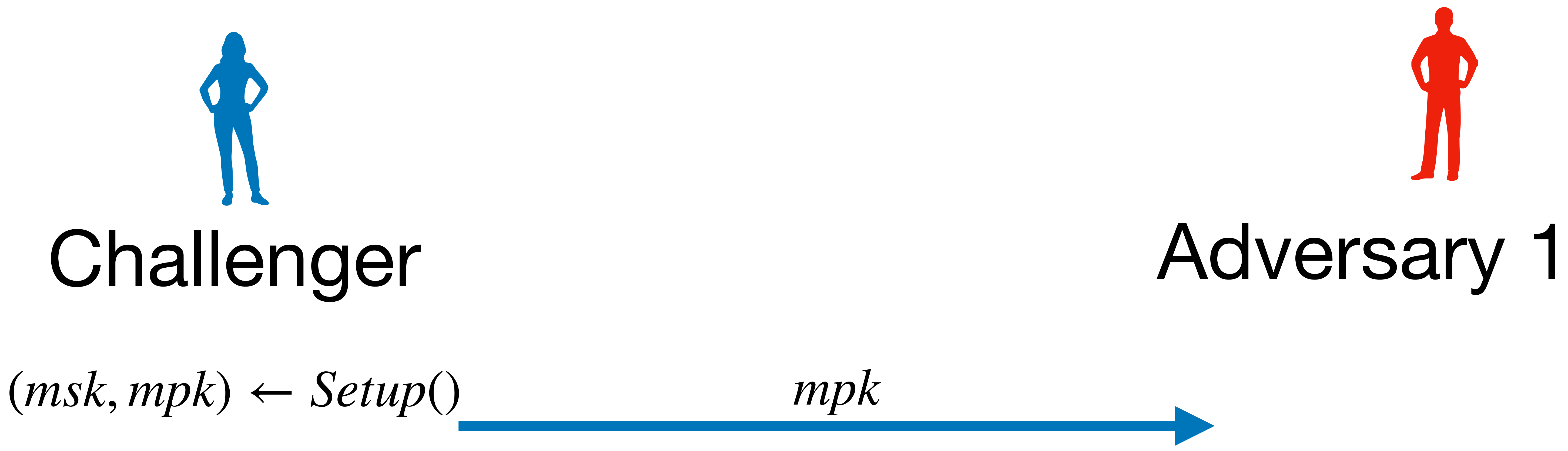
Challenger



Adversary 1

$$(msk, mpk) \leftarrow Setup()$$

Strong Incompressible (FE) Security



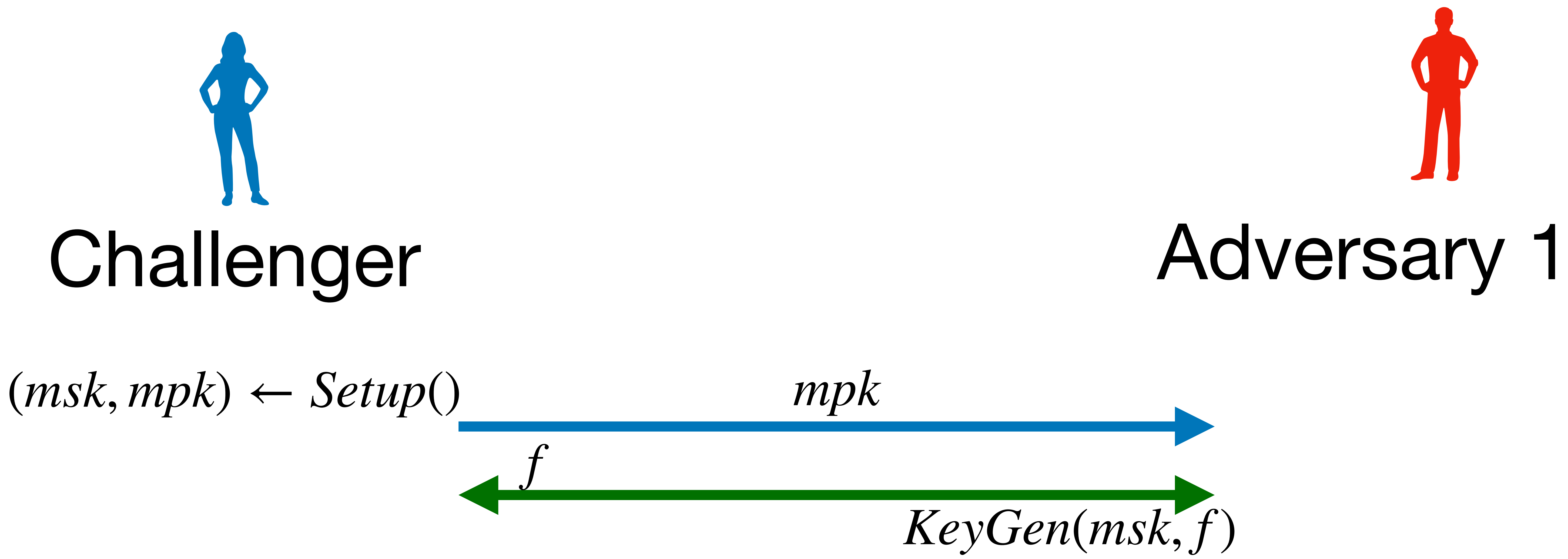
Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

$\xrightarrow{mpk}$

Strong Incompressible (FE) Security



Challenger

Adversary 1

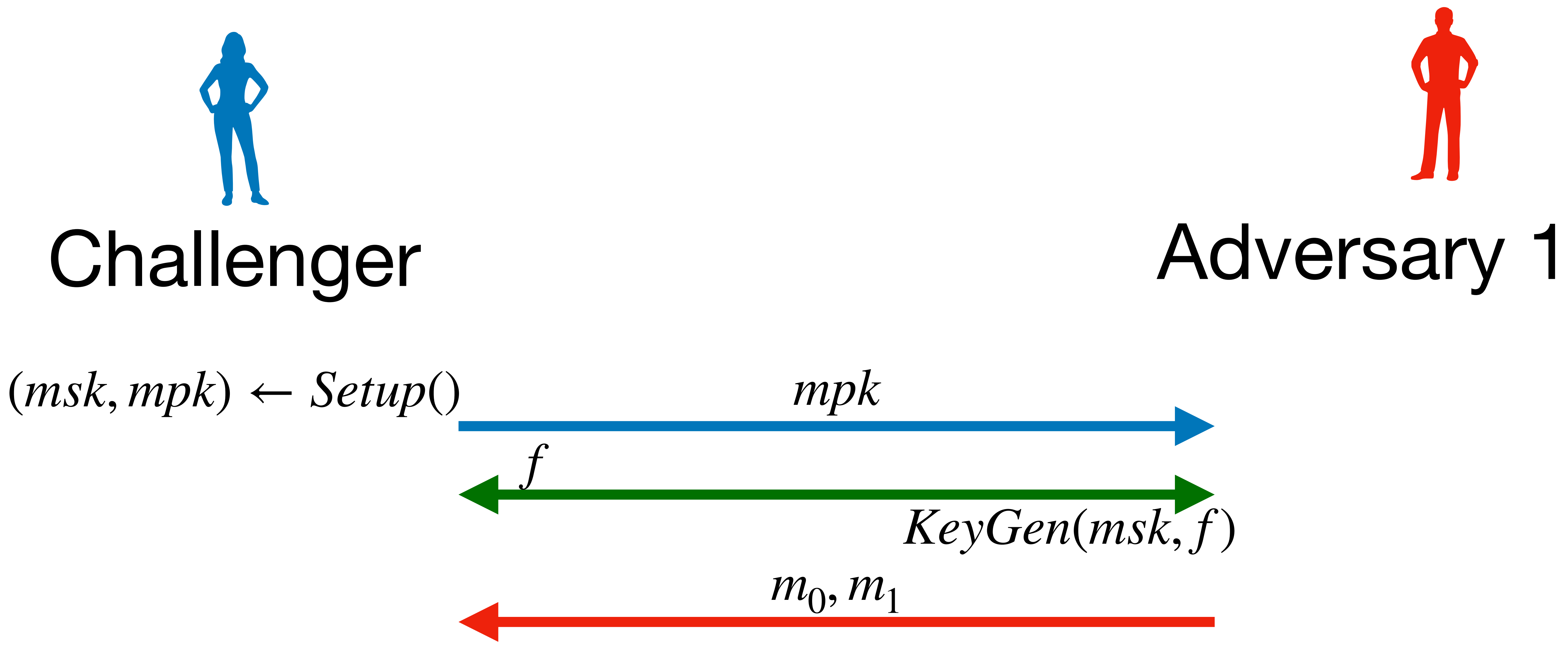
$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

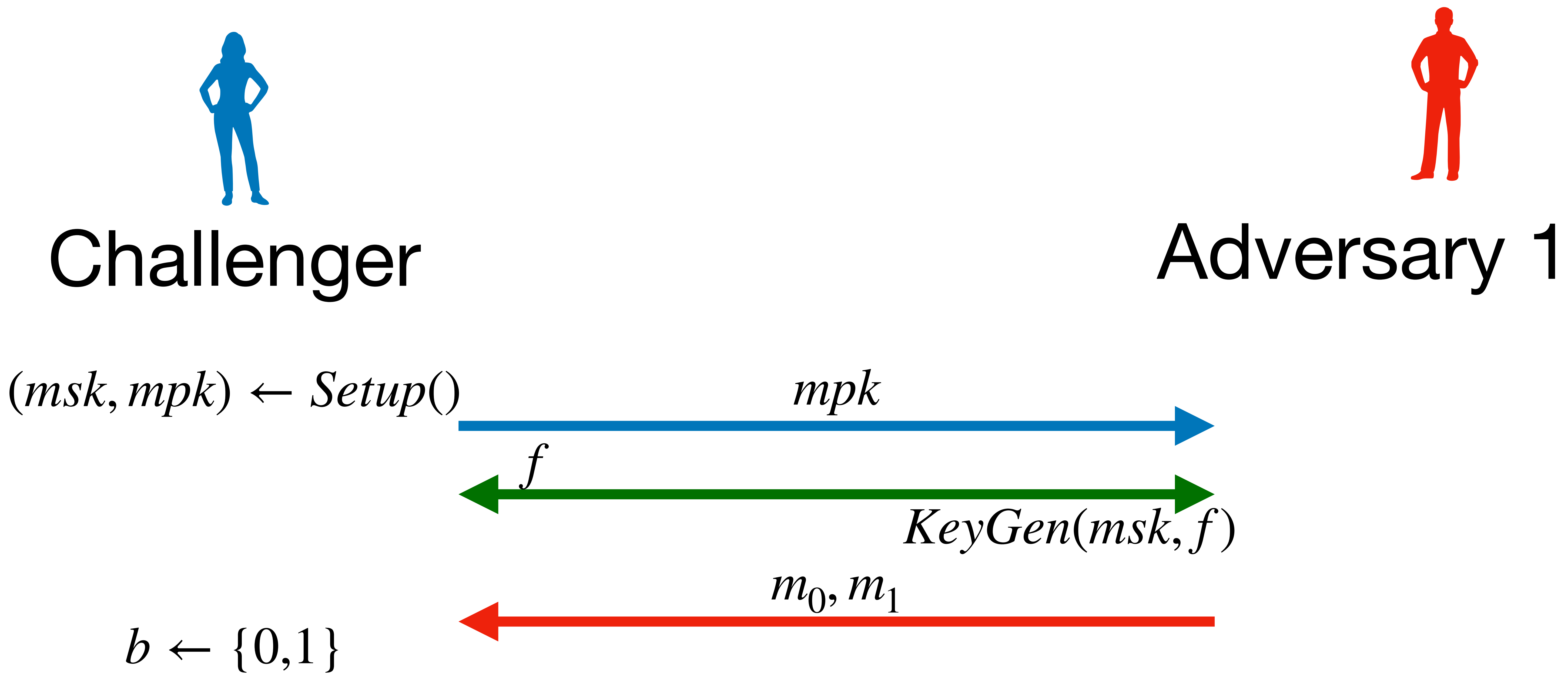
mpk

f

$KeyGen(msk, f)$

m_0, m_1

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

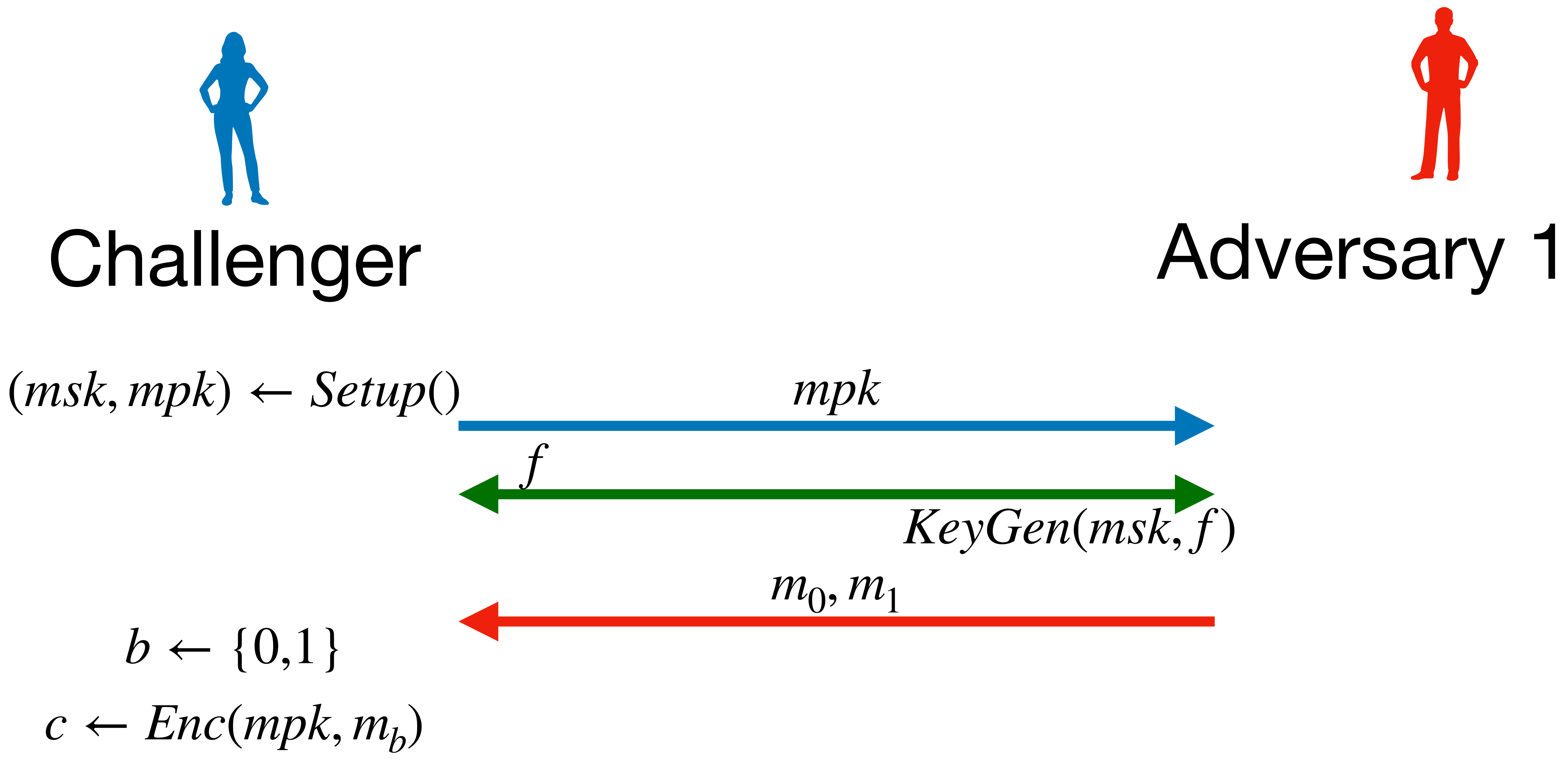
f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

Strong Incompressible (FE) Security

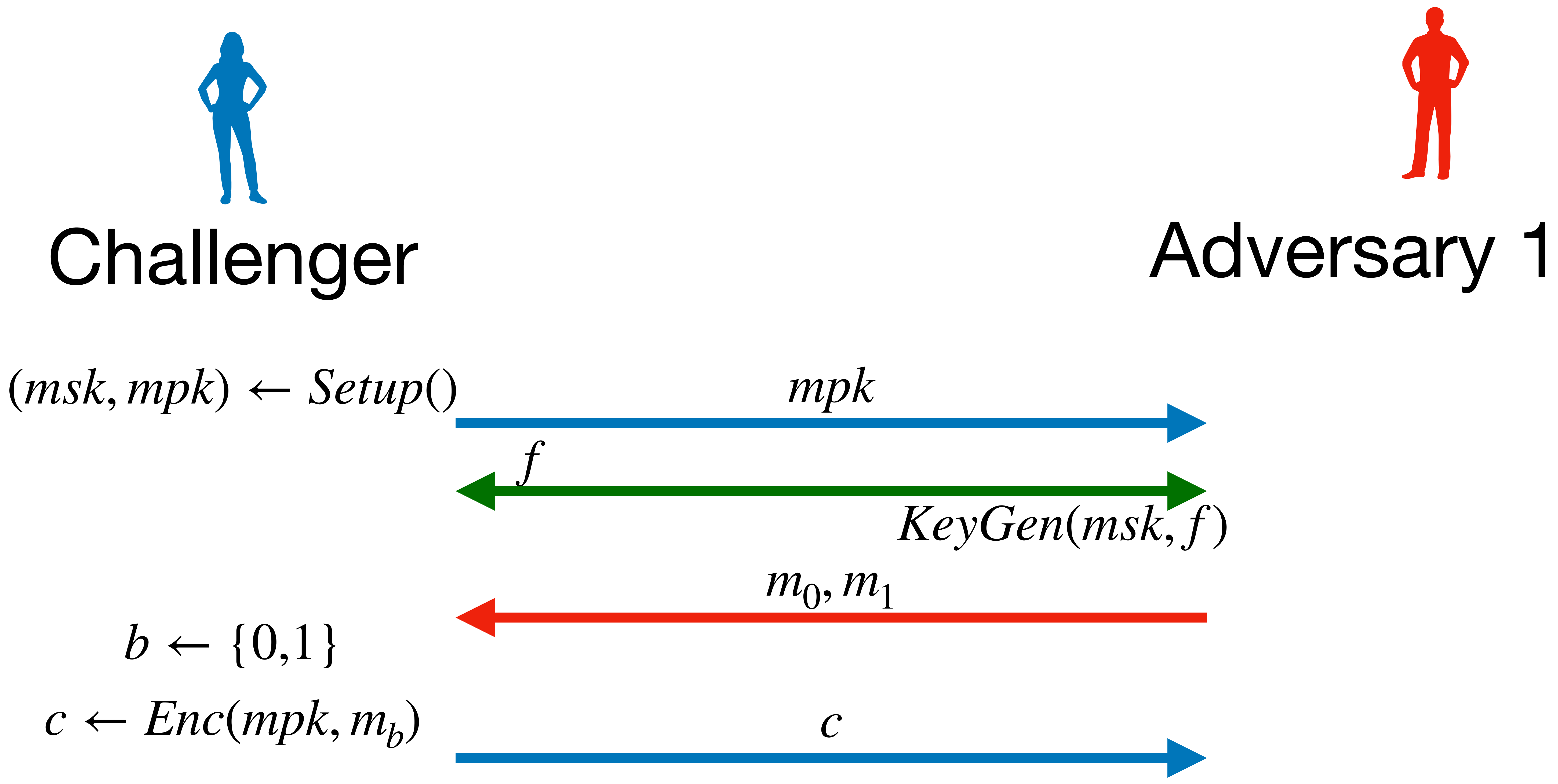


$(msk, mpk) \leftarrow Setup()$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

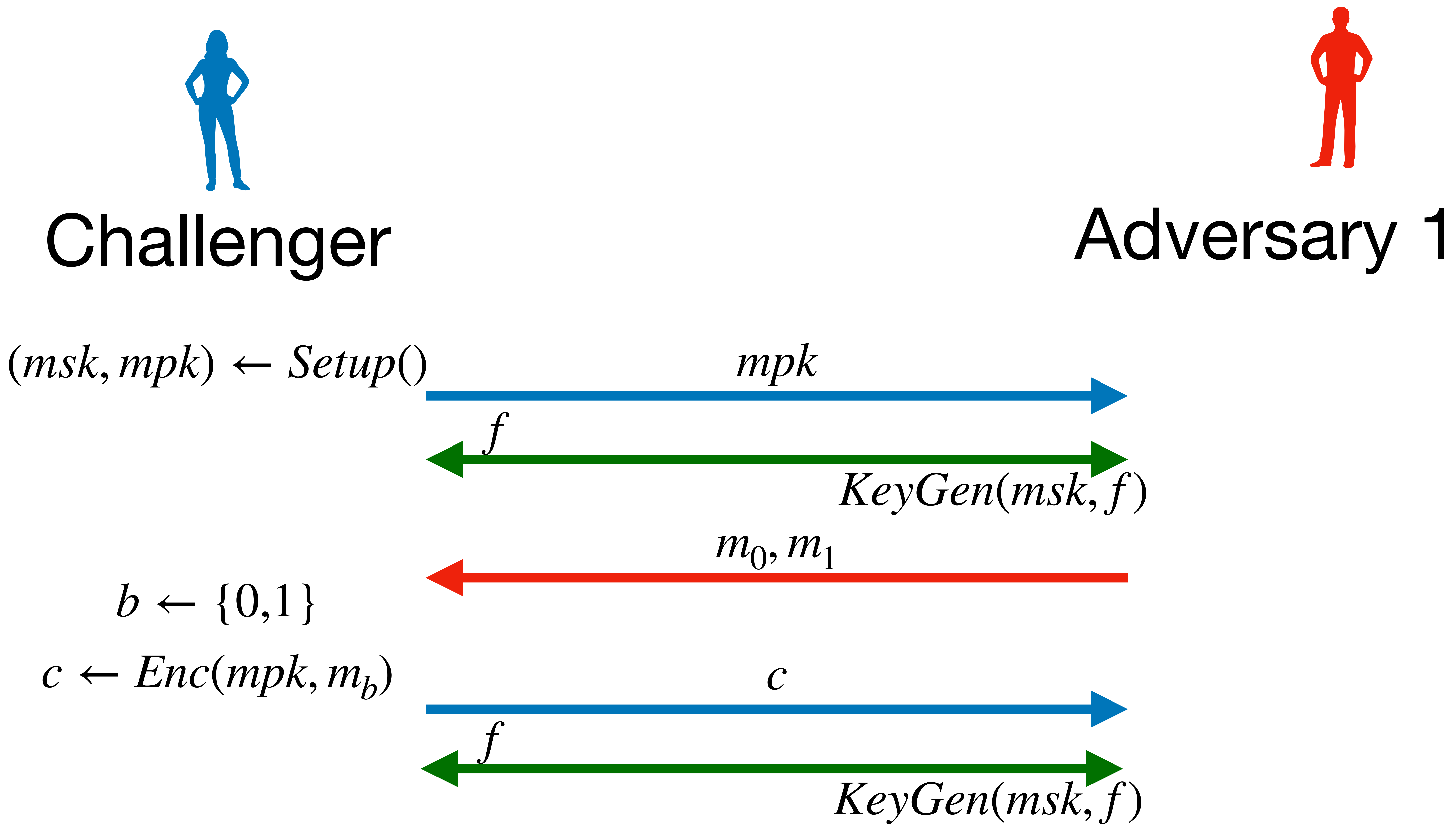
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

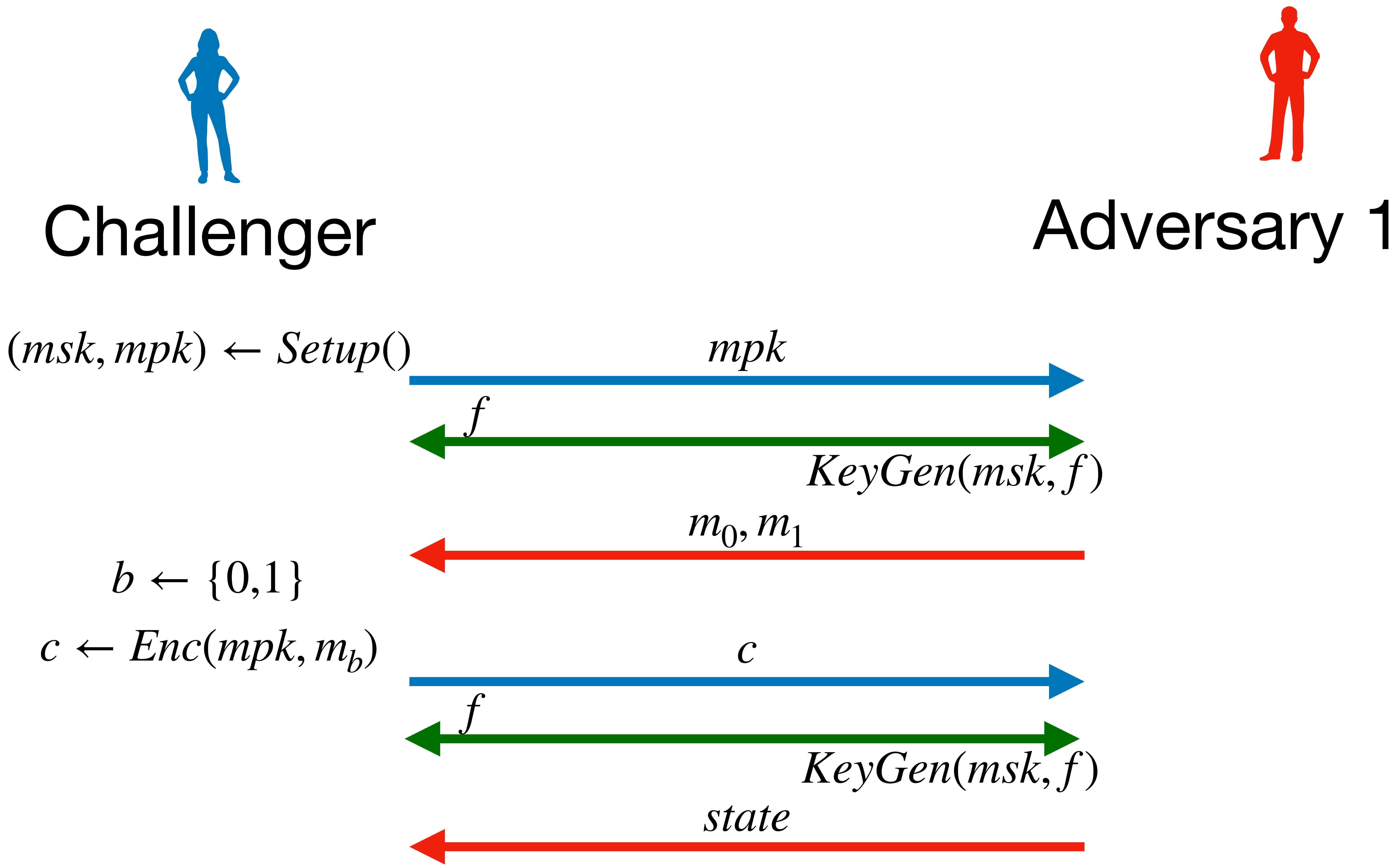
$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

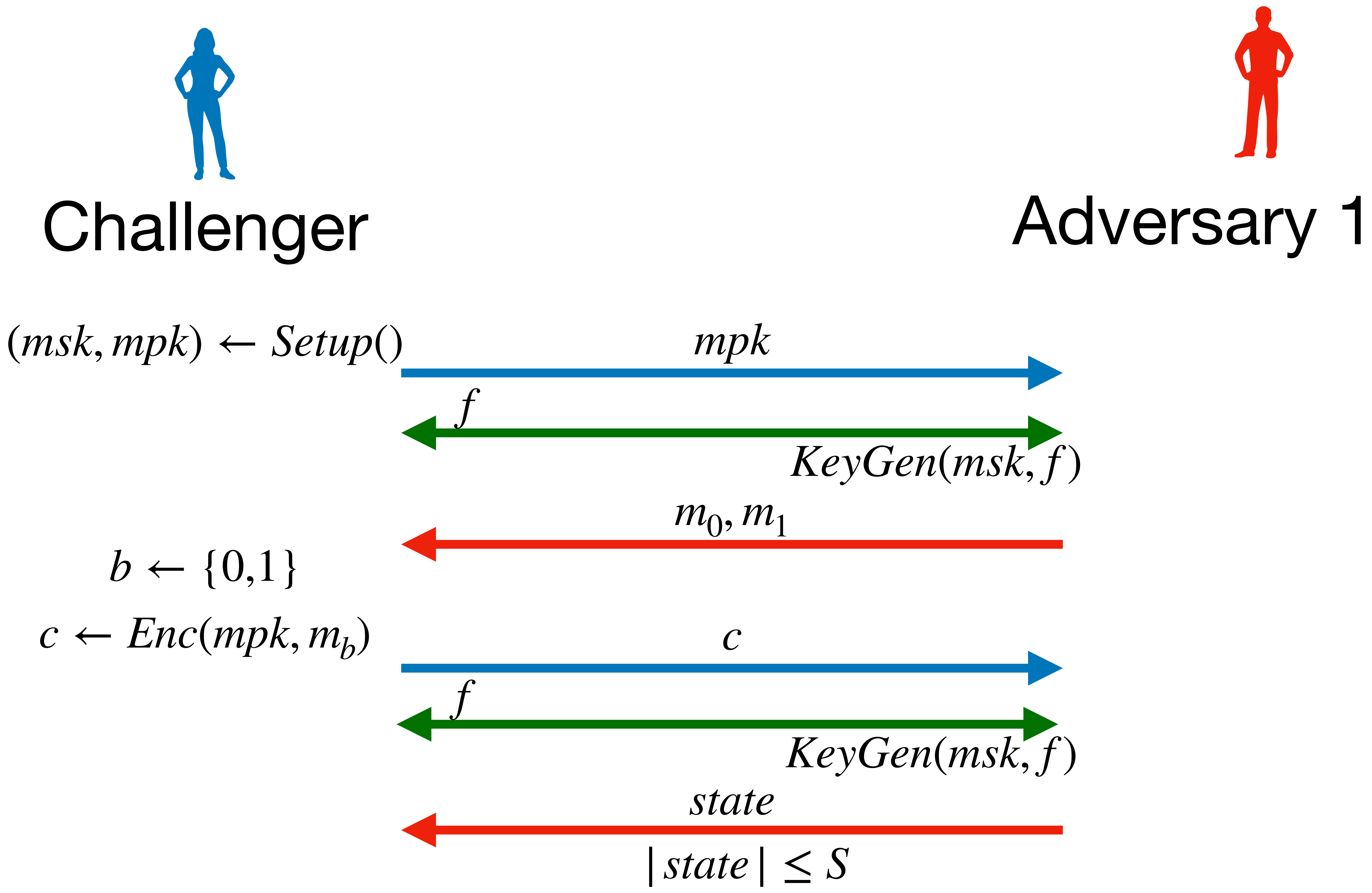
c

f

$KeyGen(msk, f)$

$state$

Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

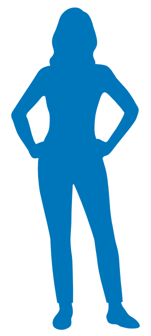
f

$KeyGen(msk, f)$

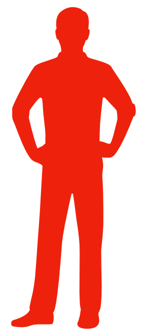
$state$

$|state| \leq S$

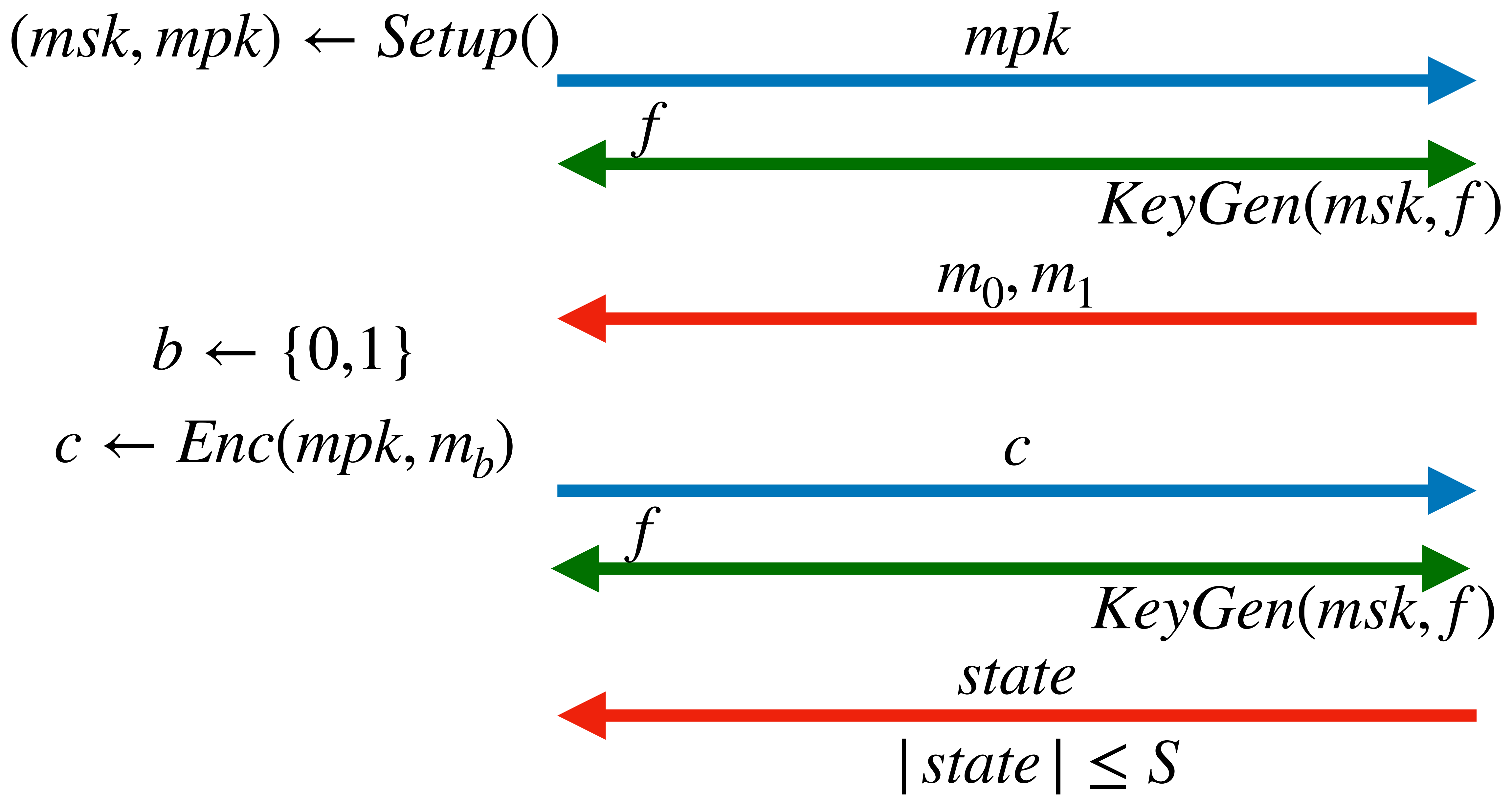
Strong Incompressible (FE) Security



Challenger



Adversary 1



$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

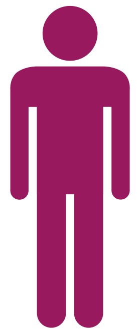
c

f

$KeyGen(msk, f)$

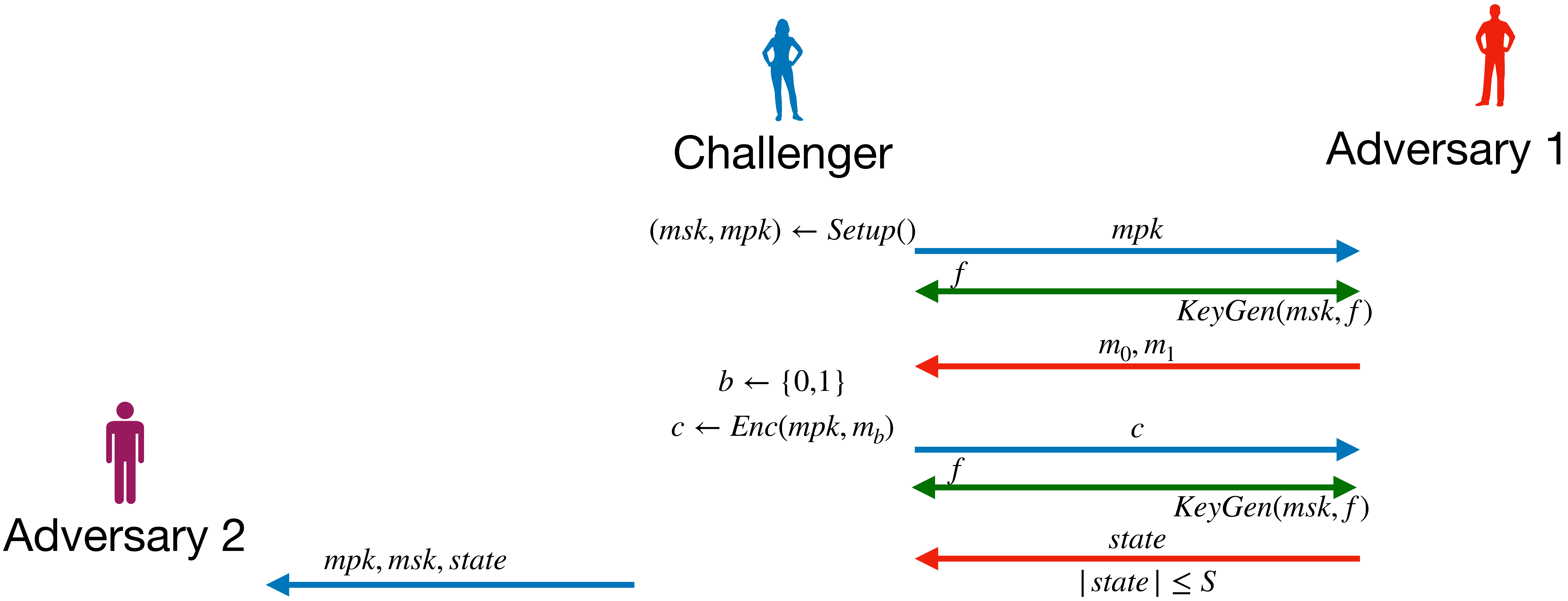
$state$

$|state| \leq S$



Adversary 2

Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

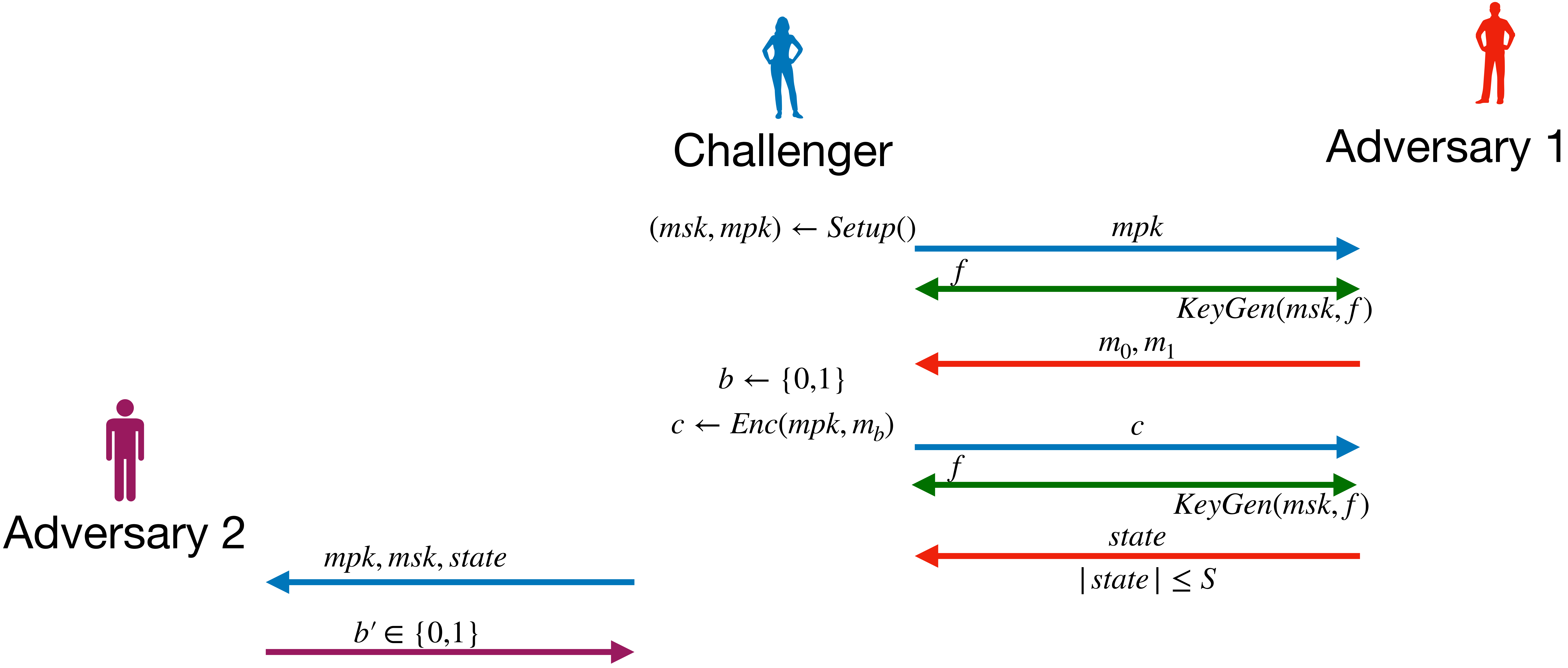
$KeyGen(msk, f)$

$state$

$|state| \leq S$

$mpk, msk, state$

Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

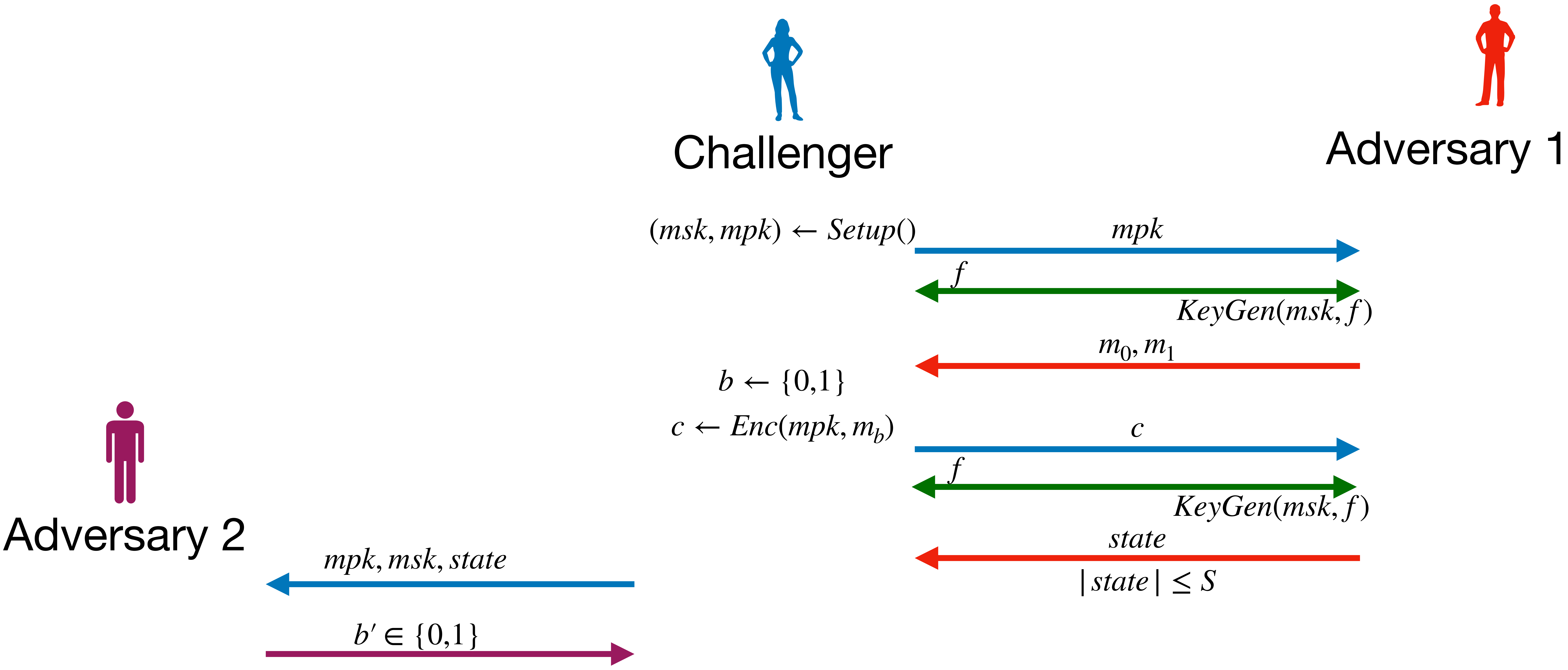
$state$

$|state| \leq S$

$mpk, msk, state$

$b' \in \{0,1\}$

Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

$mpk, msk, state$

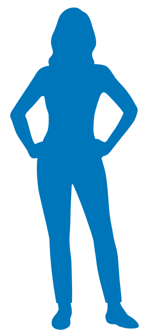
$b' \in \{0,1\}$

Adversaries wins if $b = b'$

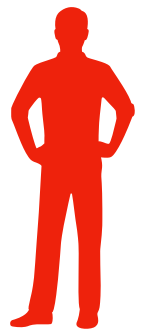
Semi-Strong Incompressible (FE) Security

Semi-Strong Incompressible (FE) Security

Semi-Strong Incompressible (FE) Security

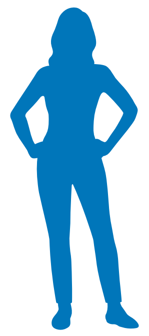


Challenger

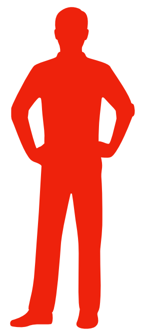


Adversary 1

Semi-Strong Incompressible (FE) Security



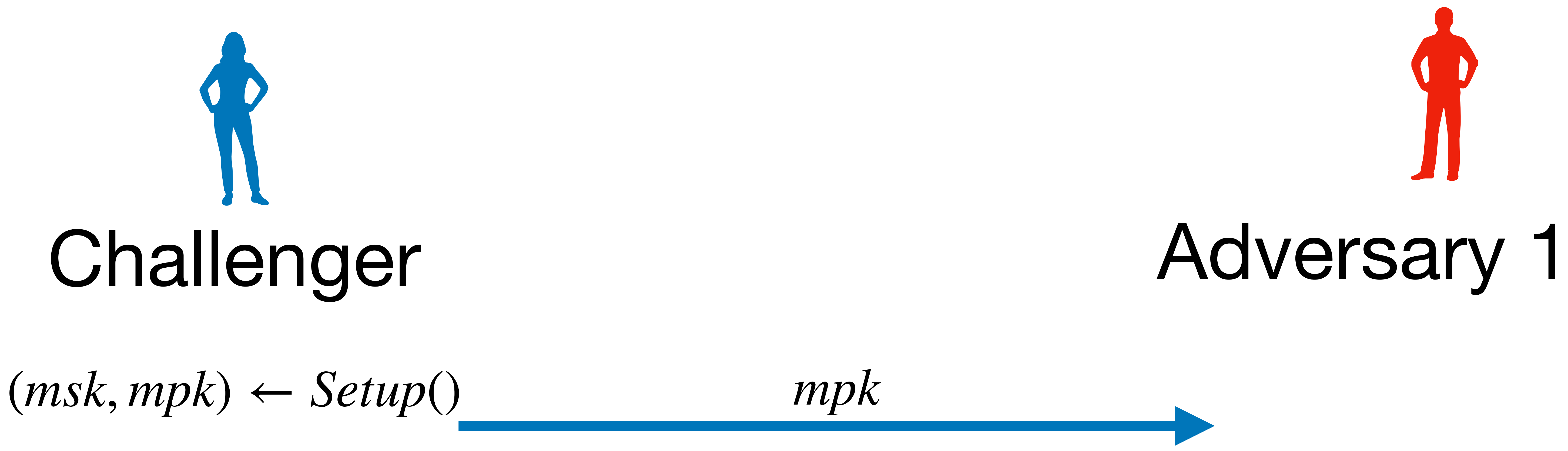
Challenger



Adversary 1

$$(msk, mpk) \leftarrow Setup()$$

Semi-Strong Incompressible (FE) Security



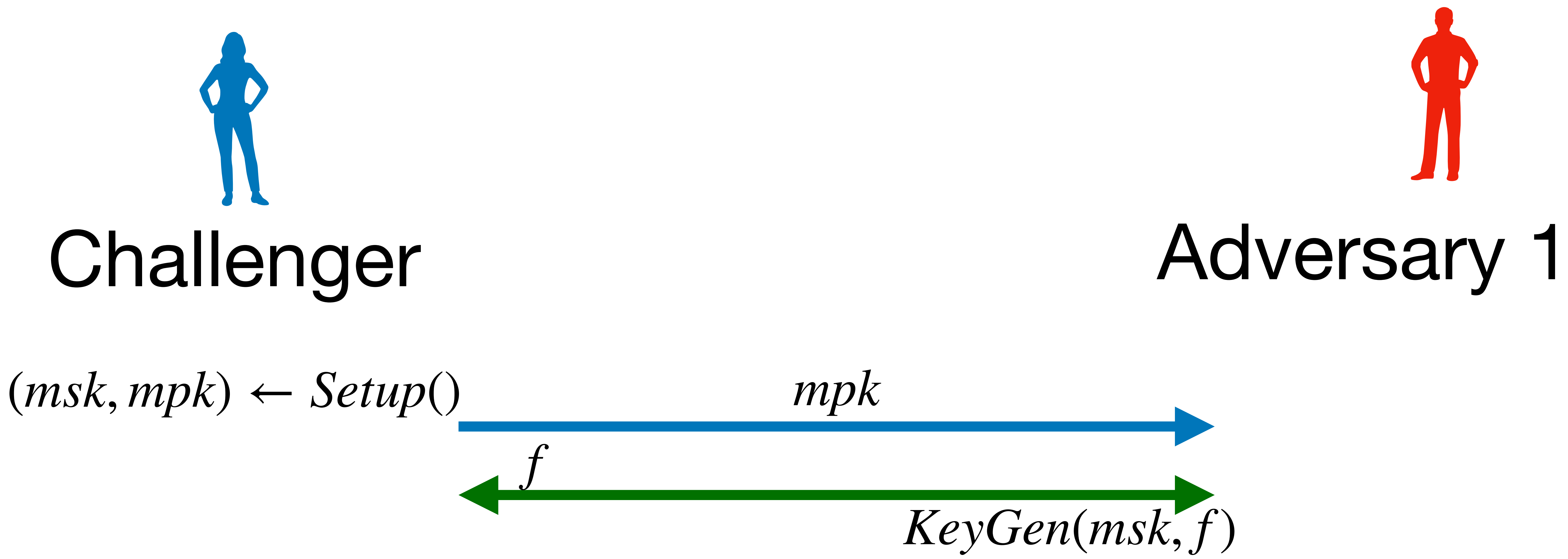
Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

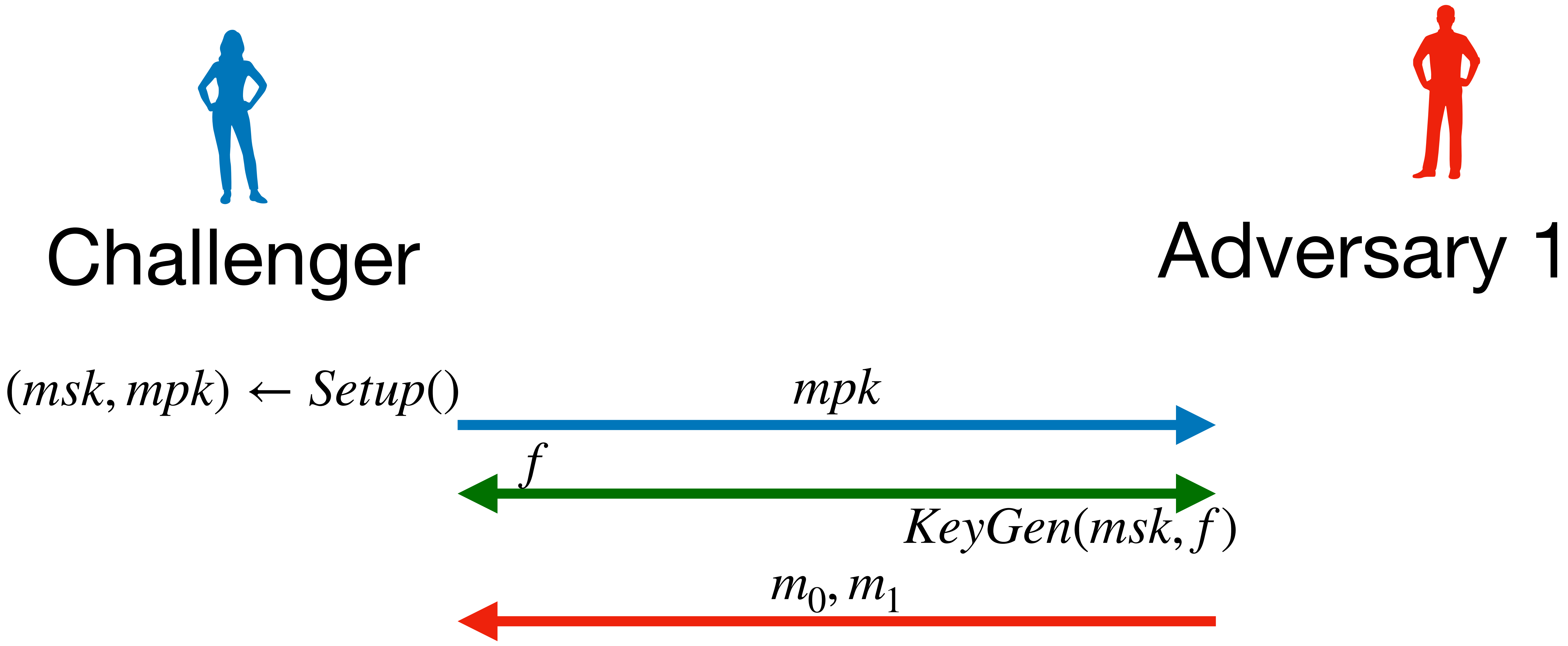
$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

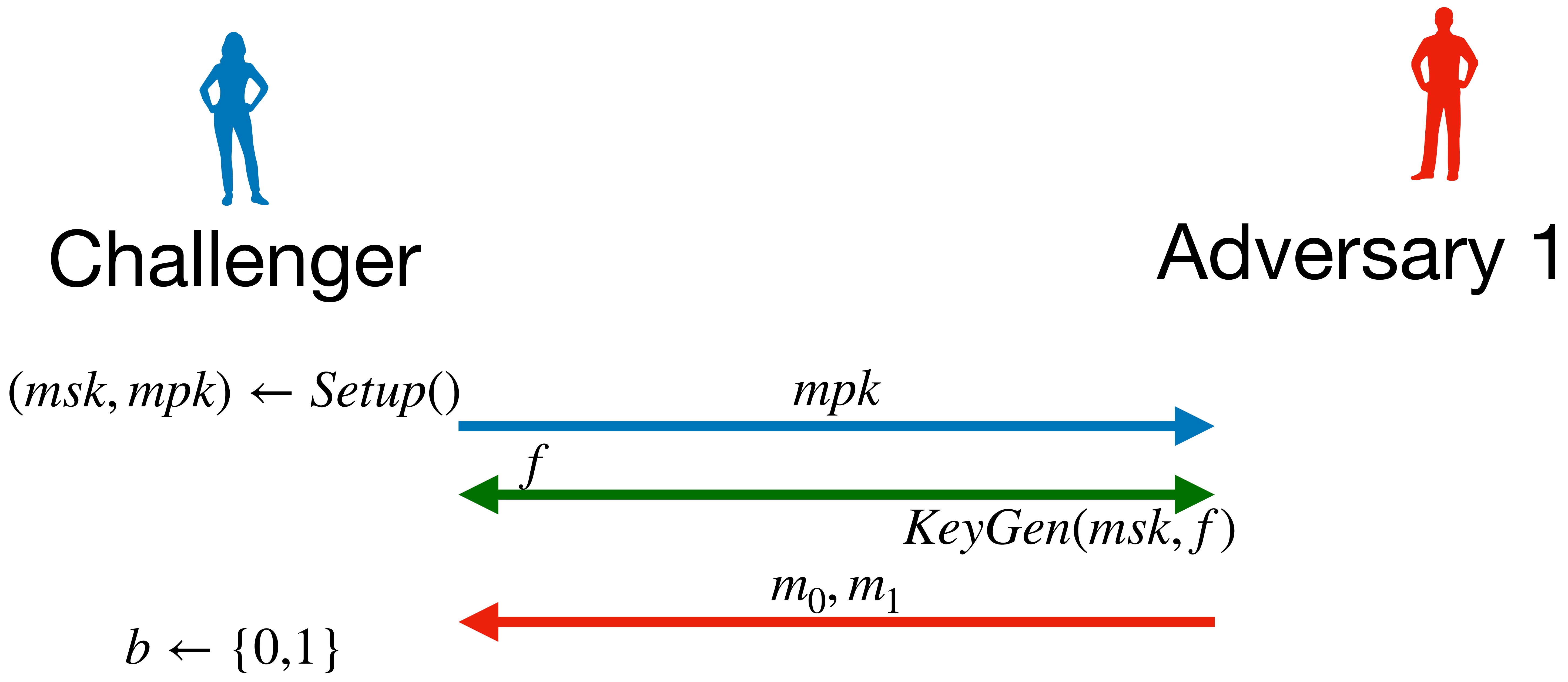
mpk

f

$KeyGen(msk, f)$

m_0, m_1

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

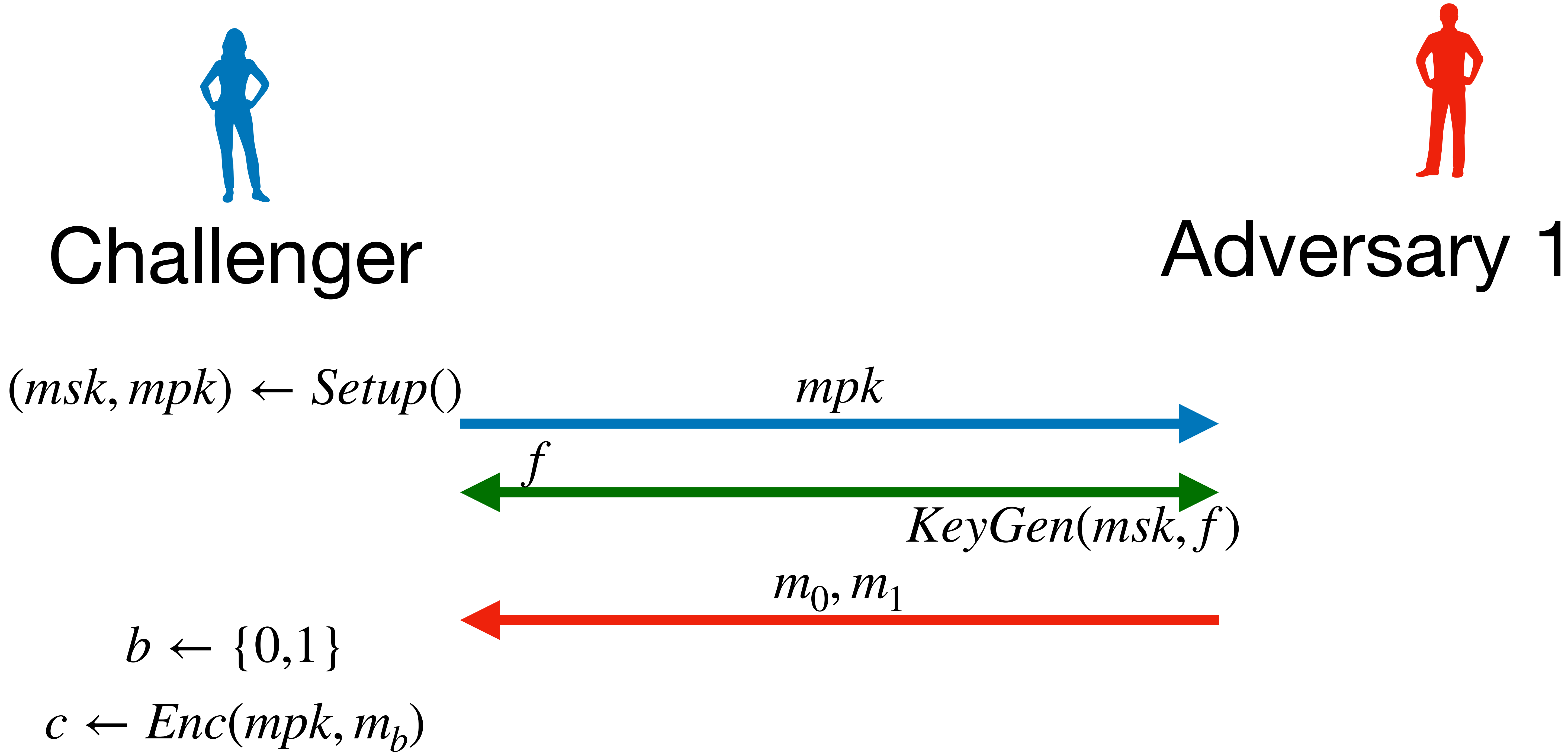
f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

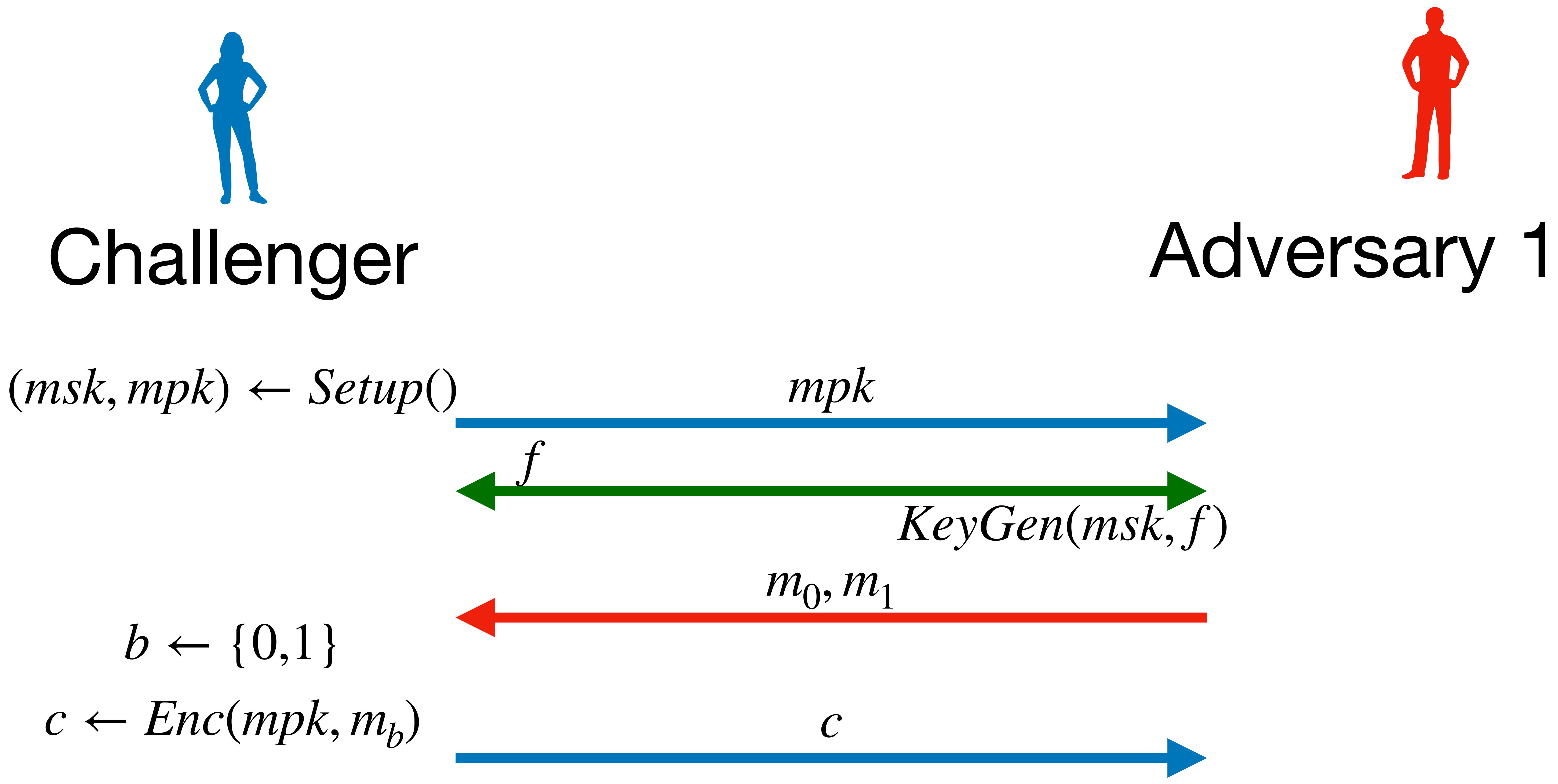
$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

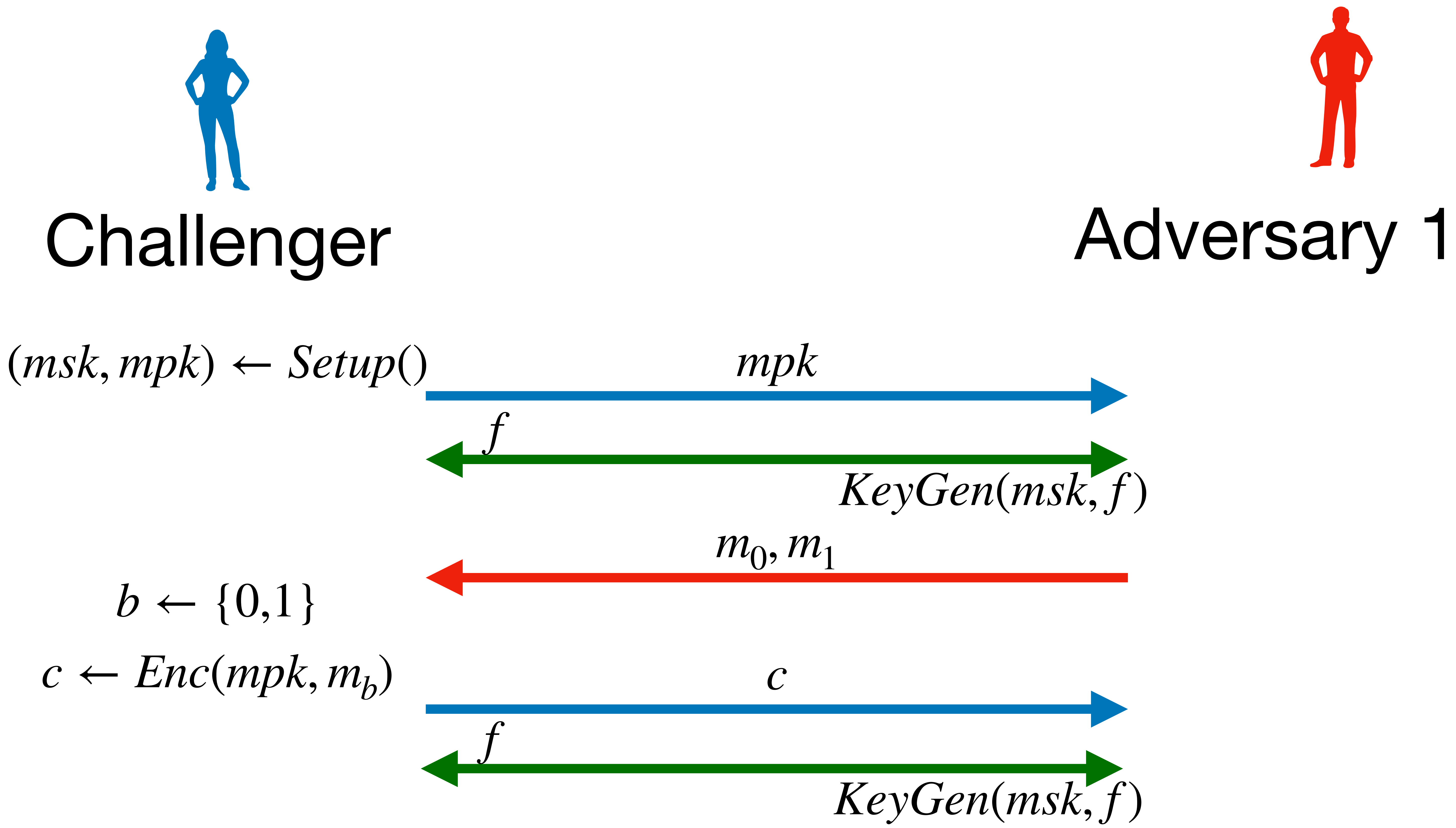
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

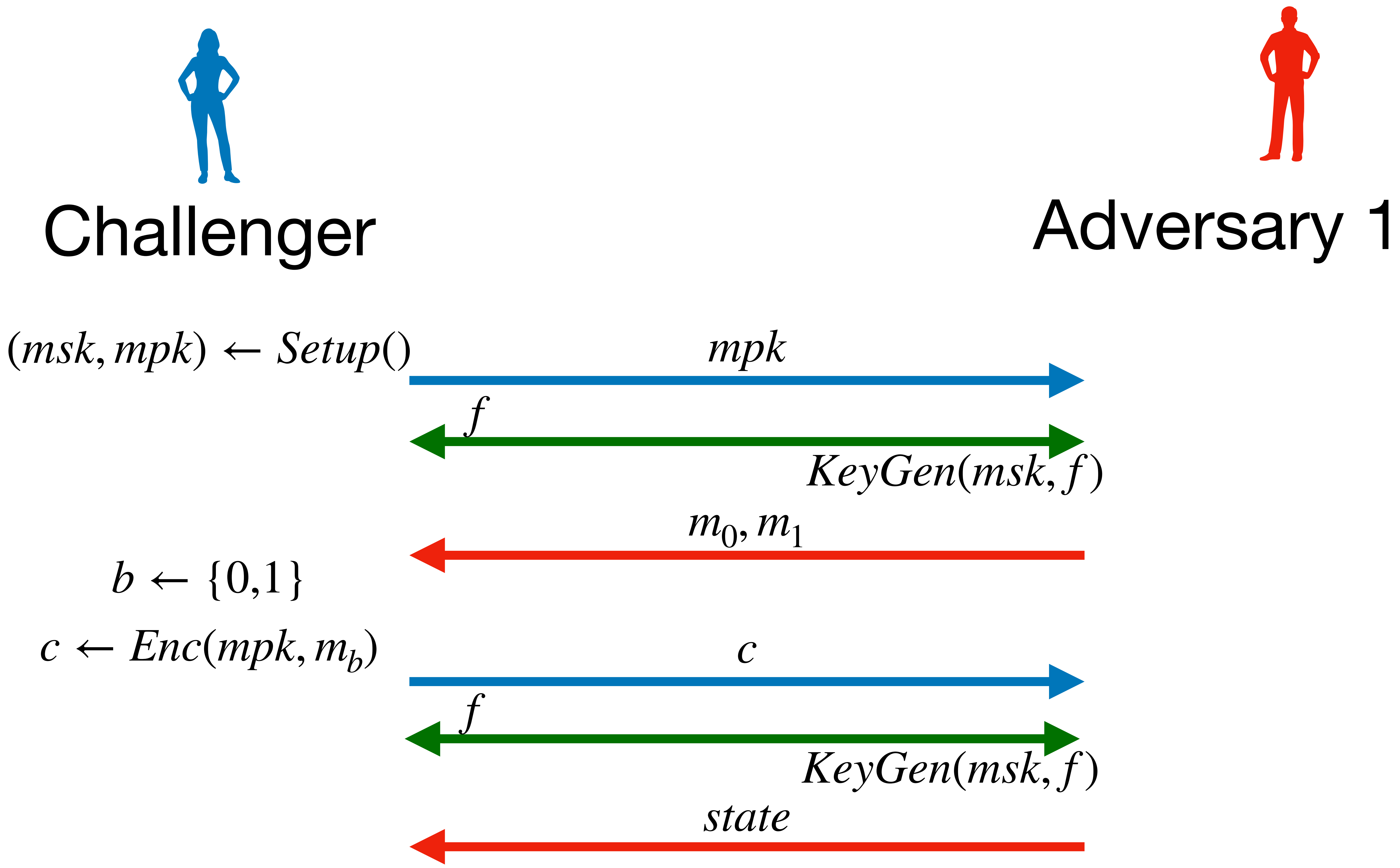
$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

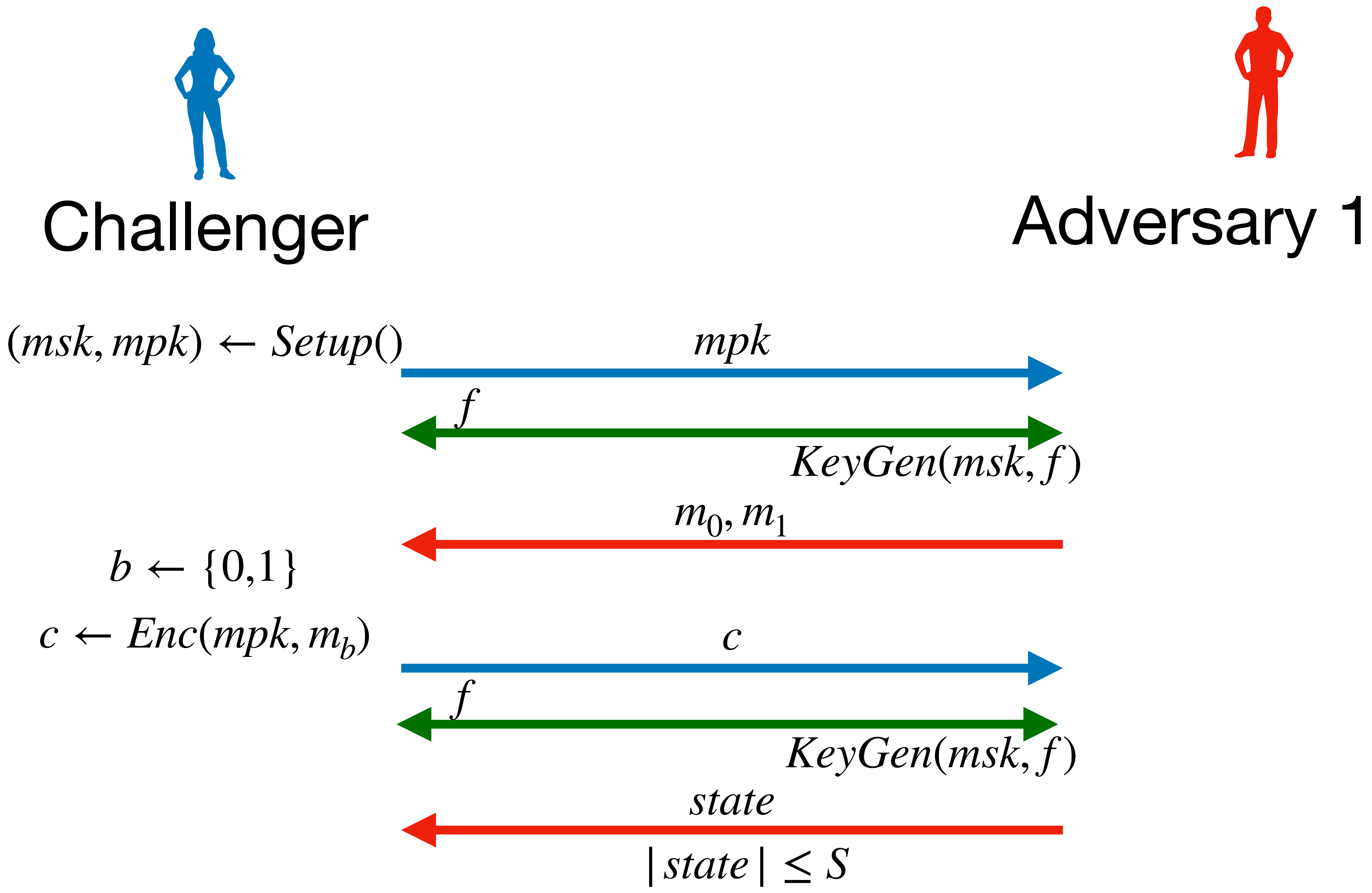
c

f

$KeyGen(msk, f)$

$state$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

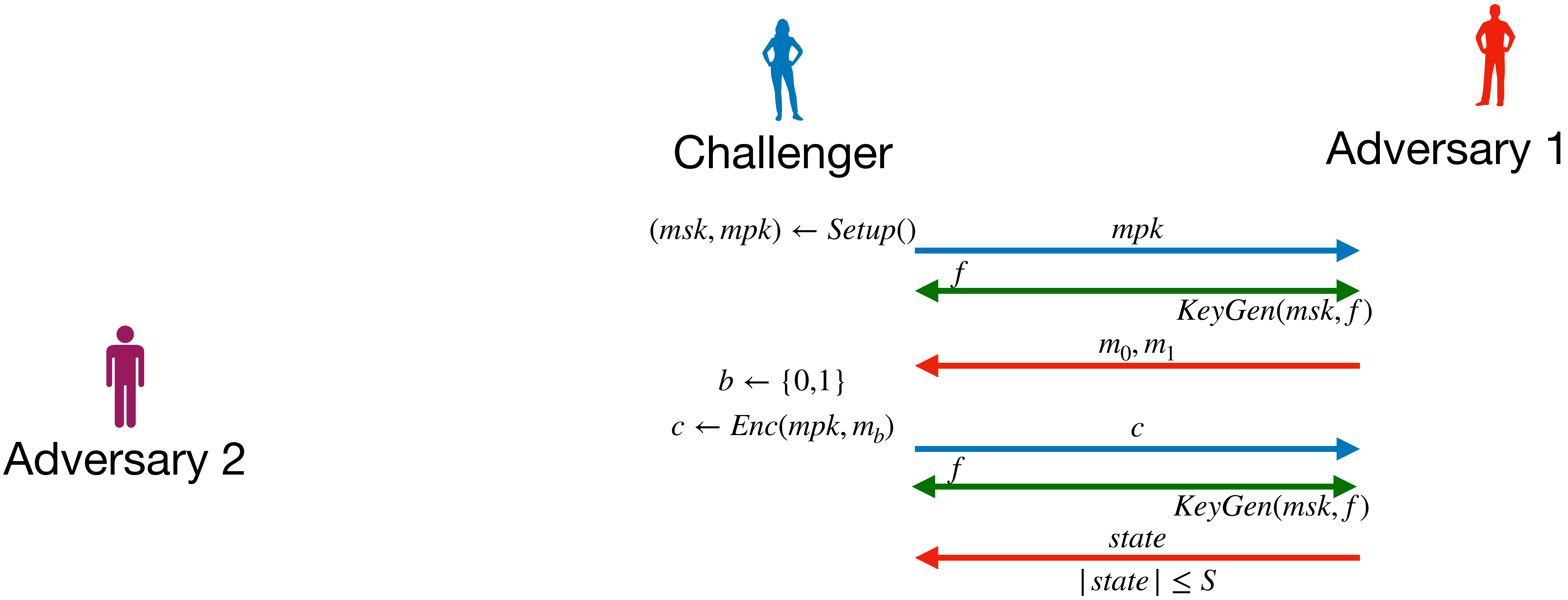
f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

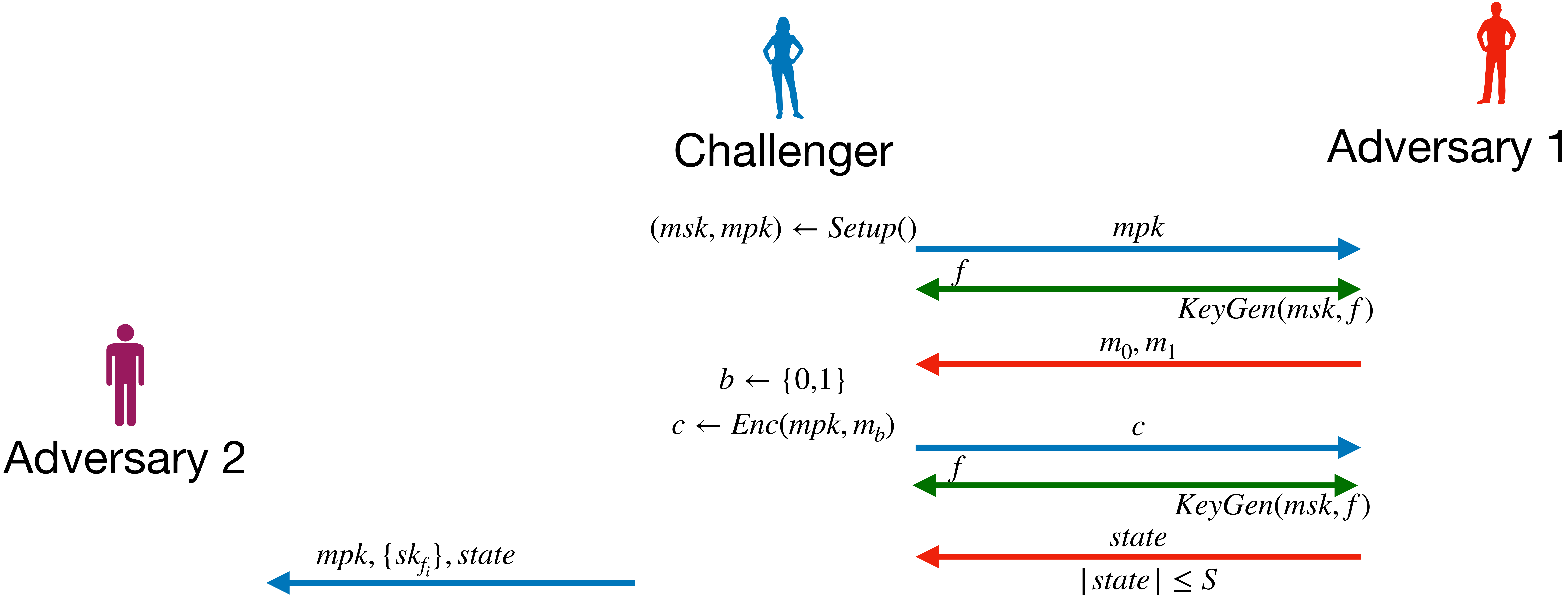
f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

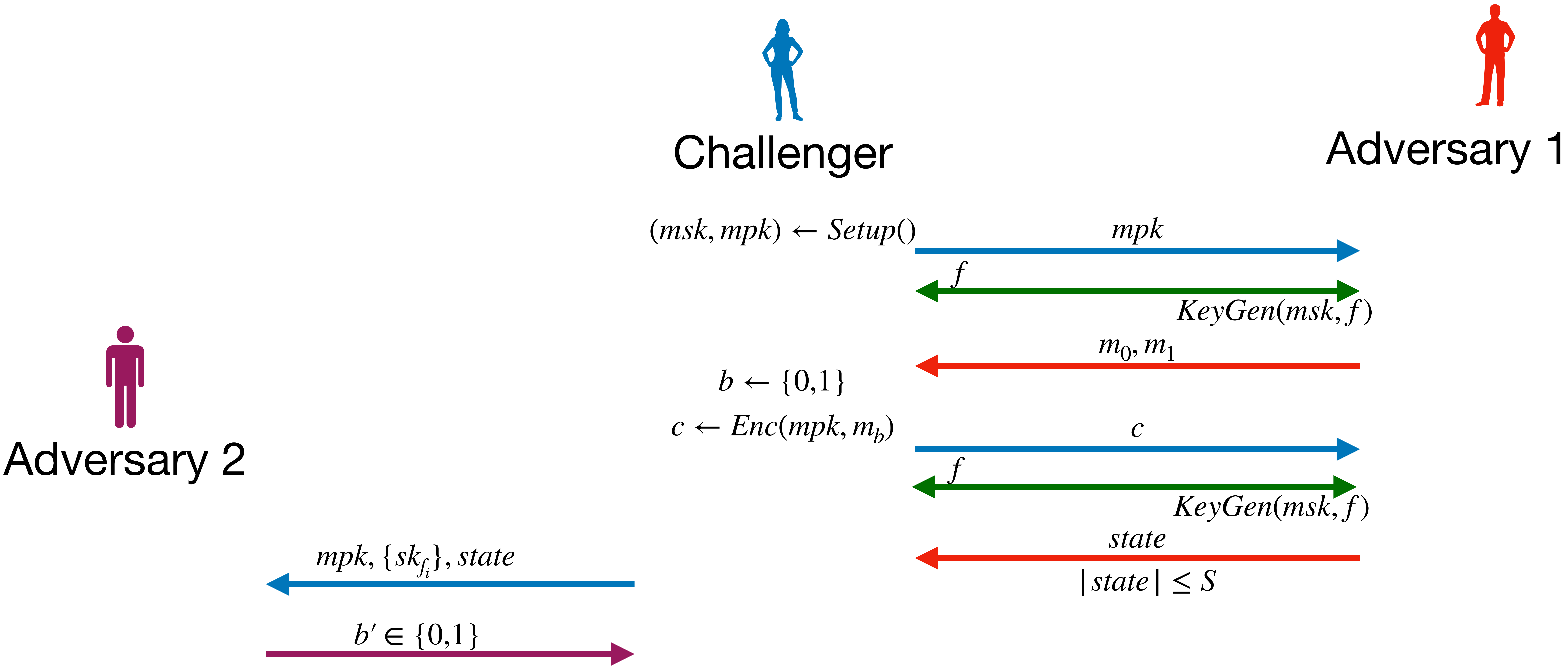
$KeyGen(msk, f)$

$state$

$|state| \leq S$

$mpk, \{sk_{f_i}\}, state$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

$state$

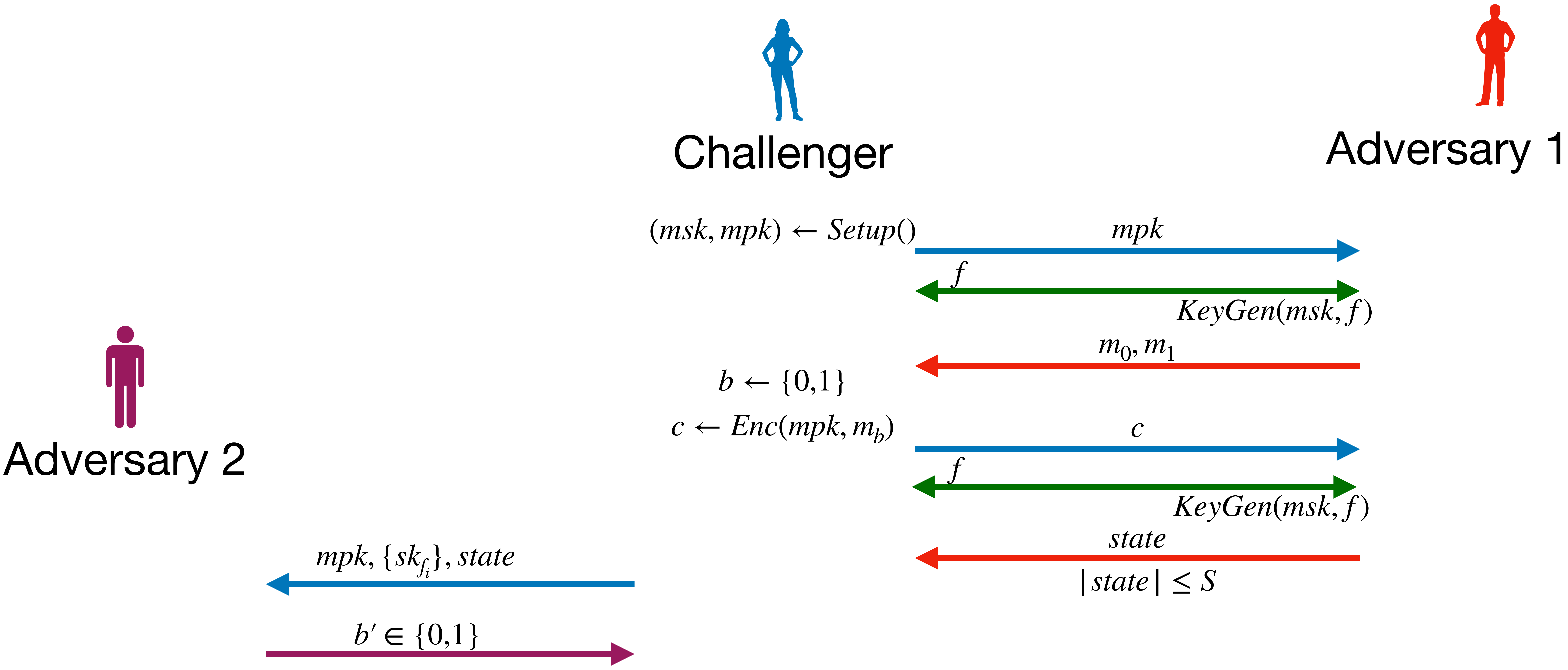
$|state| \leq S$

Adversary 2

$mpk, \{sk_{f_i}\}, state$

$b' \in \{0,1\}$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

$state$

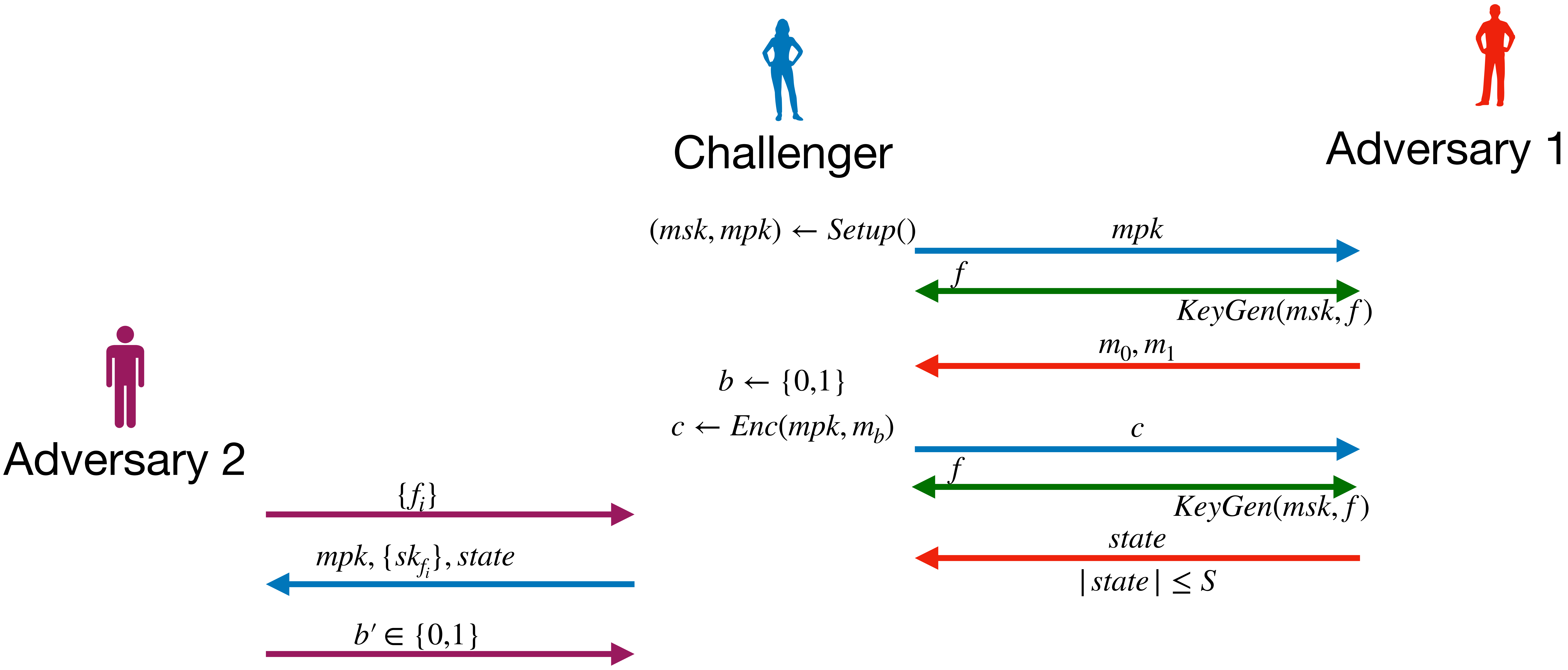
$|state| \leq S$

$mpk, \{sk_{f_i}\}, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

Semi-Strong Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

Adversary 2

$\{f_i\}$

$mpk, \{sk_{f_i}\}, state$

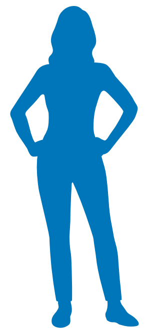
$b' \in \{0,1\}$

Adversaries wins if $b = b'$

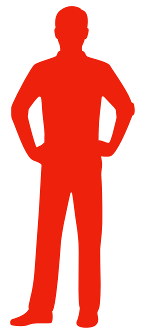
Regular Incompressible (FE) Security

Regular Incompressible (FE) Security

Regular Incompressible (FE) Security

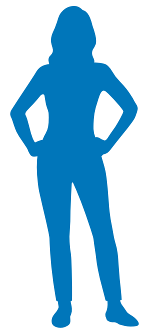


Challenger

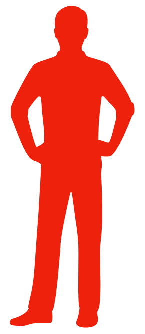


Adversary 1

Regular Incompressible (FE) Security



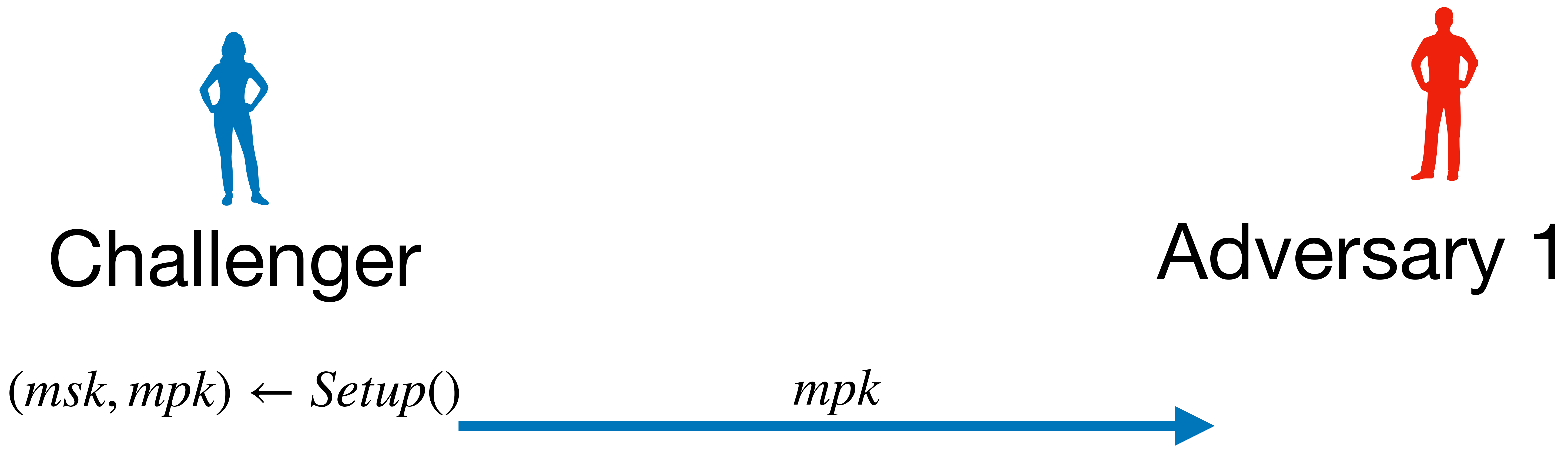
Challenger



Adversary 1

$$(msk, mpk) \leftarrow Setup()$$

Regular Incompressible (FE) Security

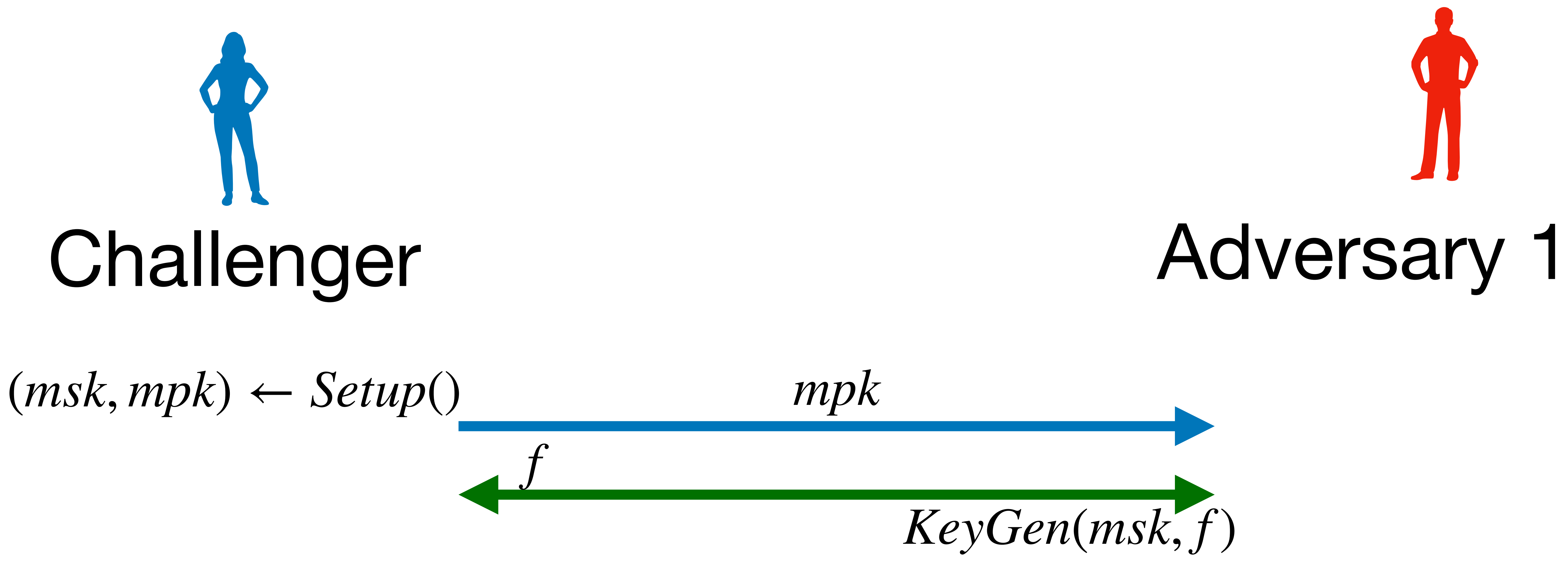


Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$ $\xrightarrow{mpk}$

Regular Incompressible (FE) Security



Challenger

Adversary 1

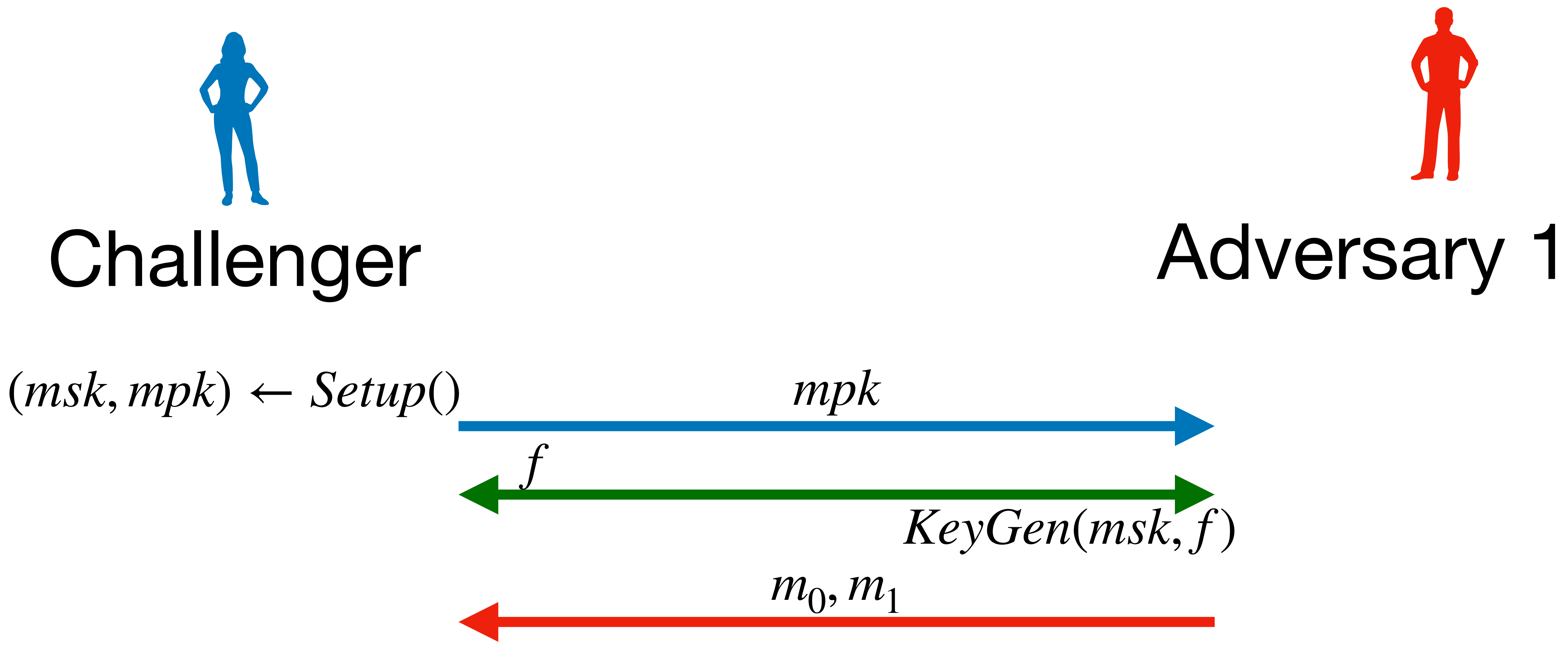
$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

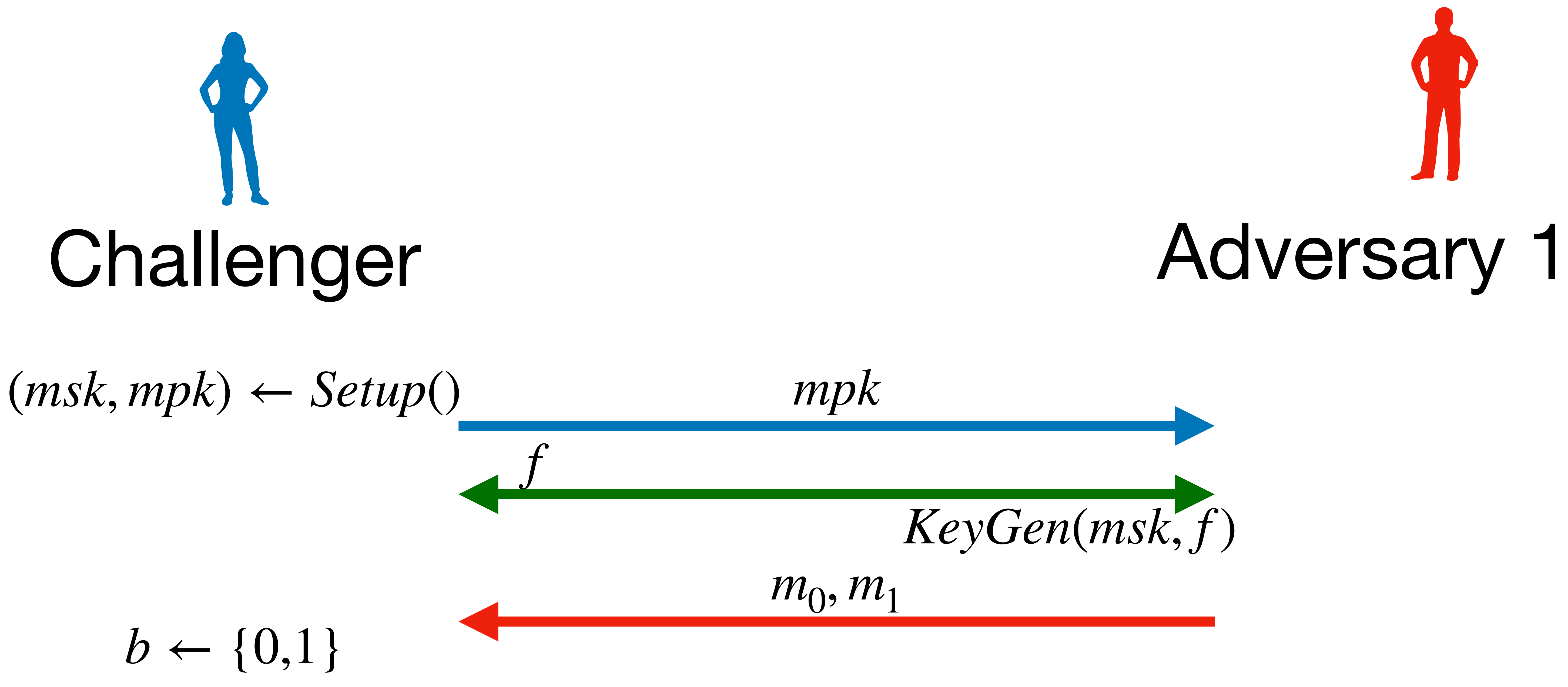
mpk

f

$KeyGen(msk, f)$

m_0, m_1

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

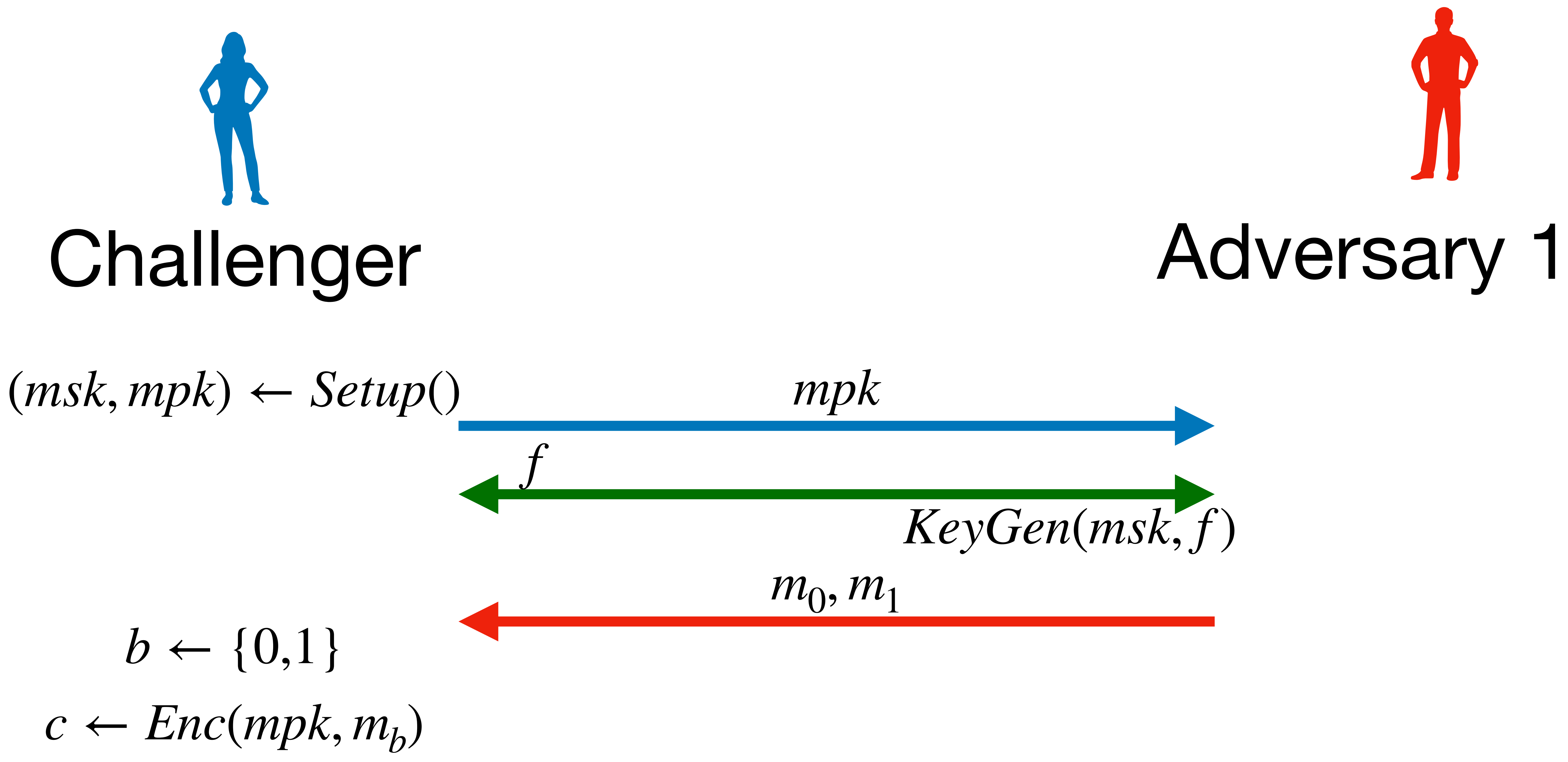
f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

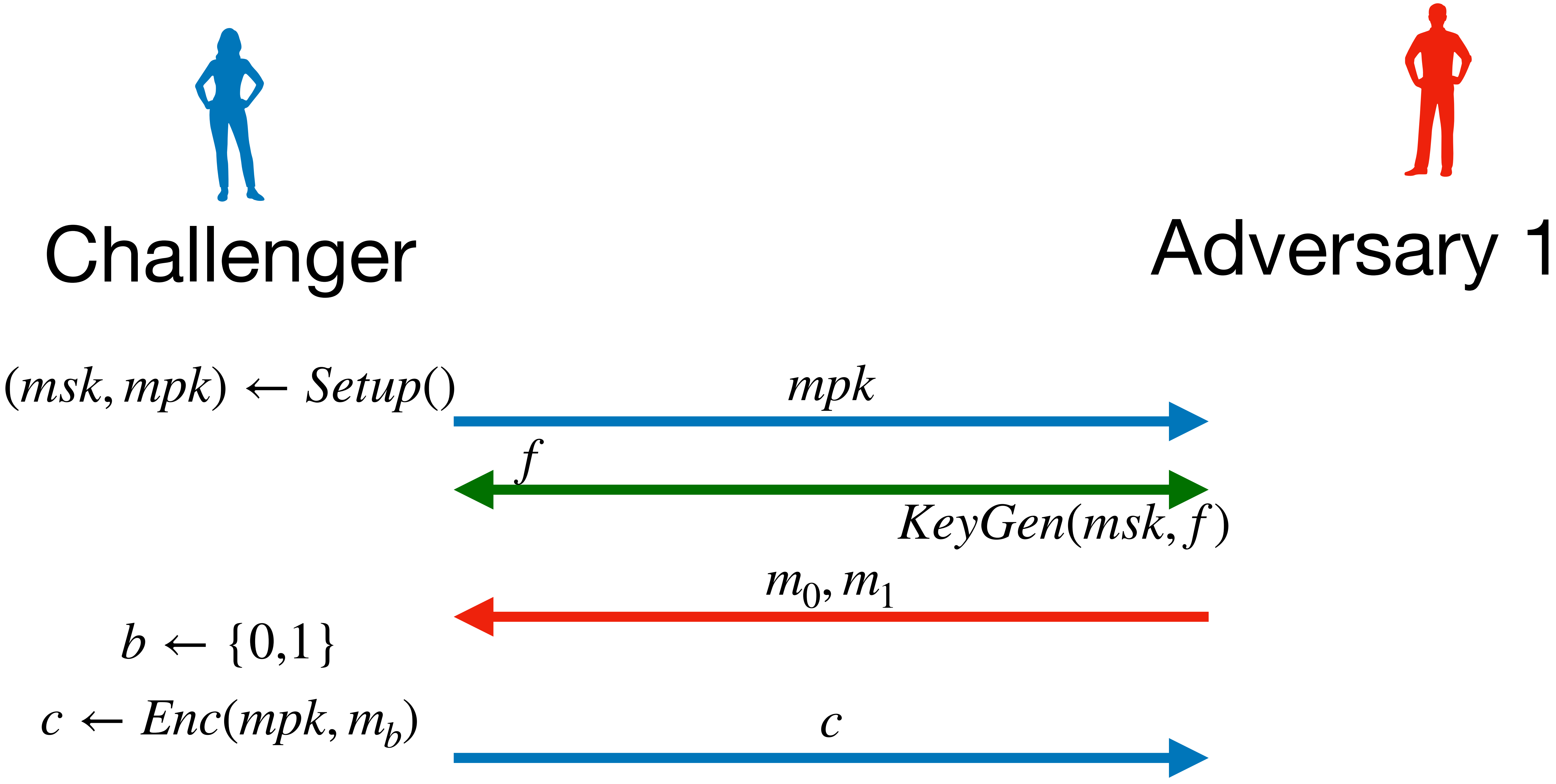
$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

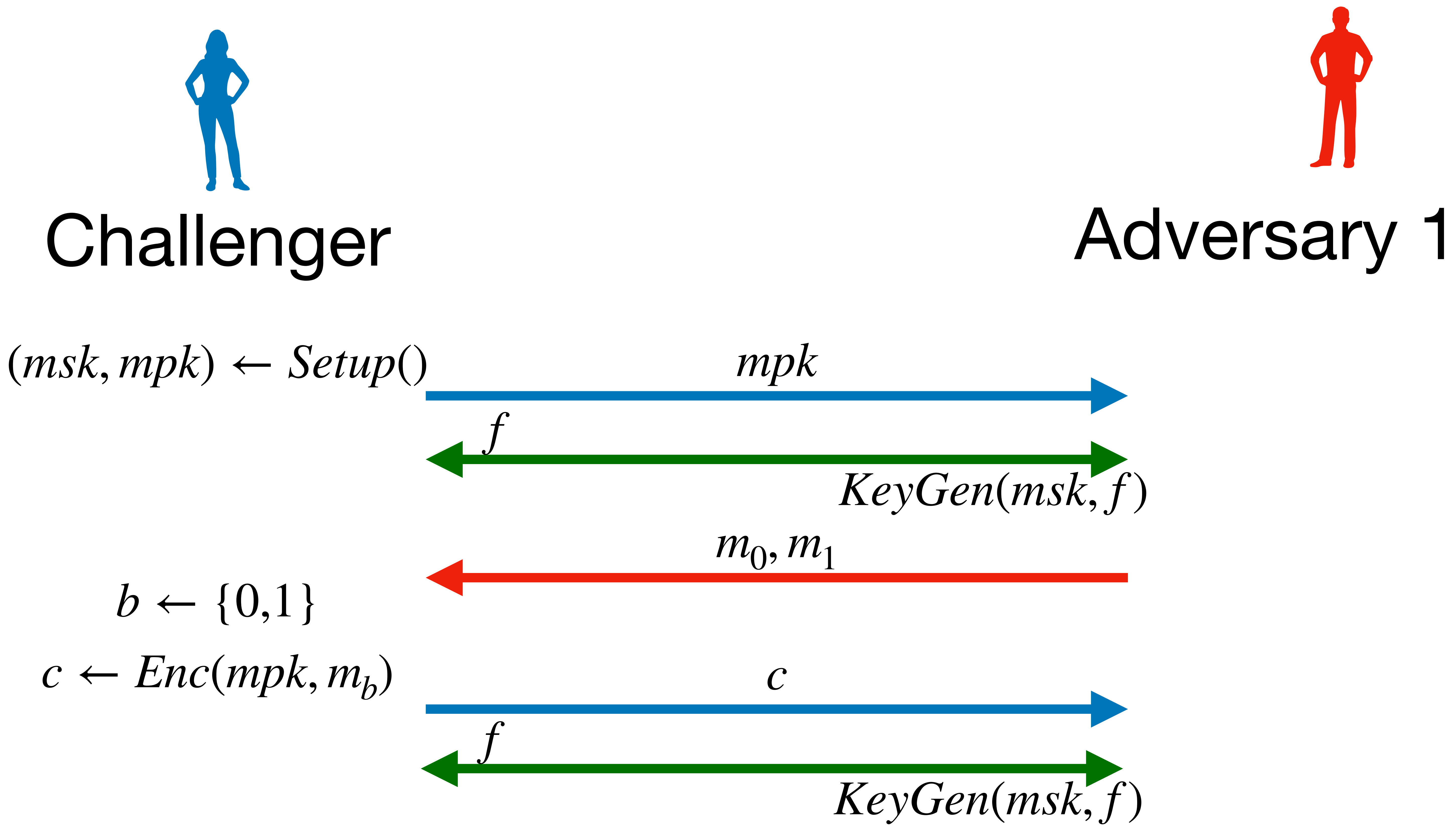
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

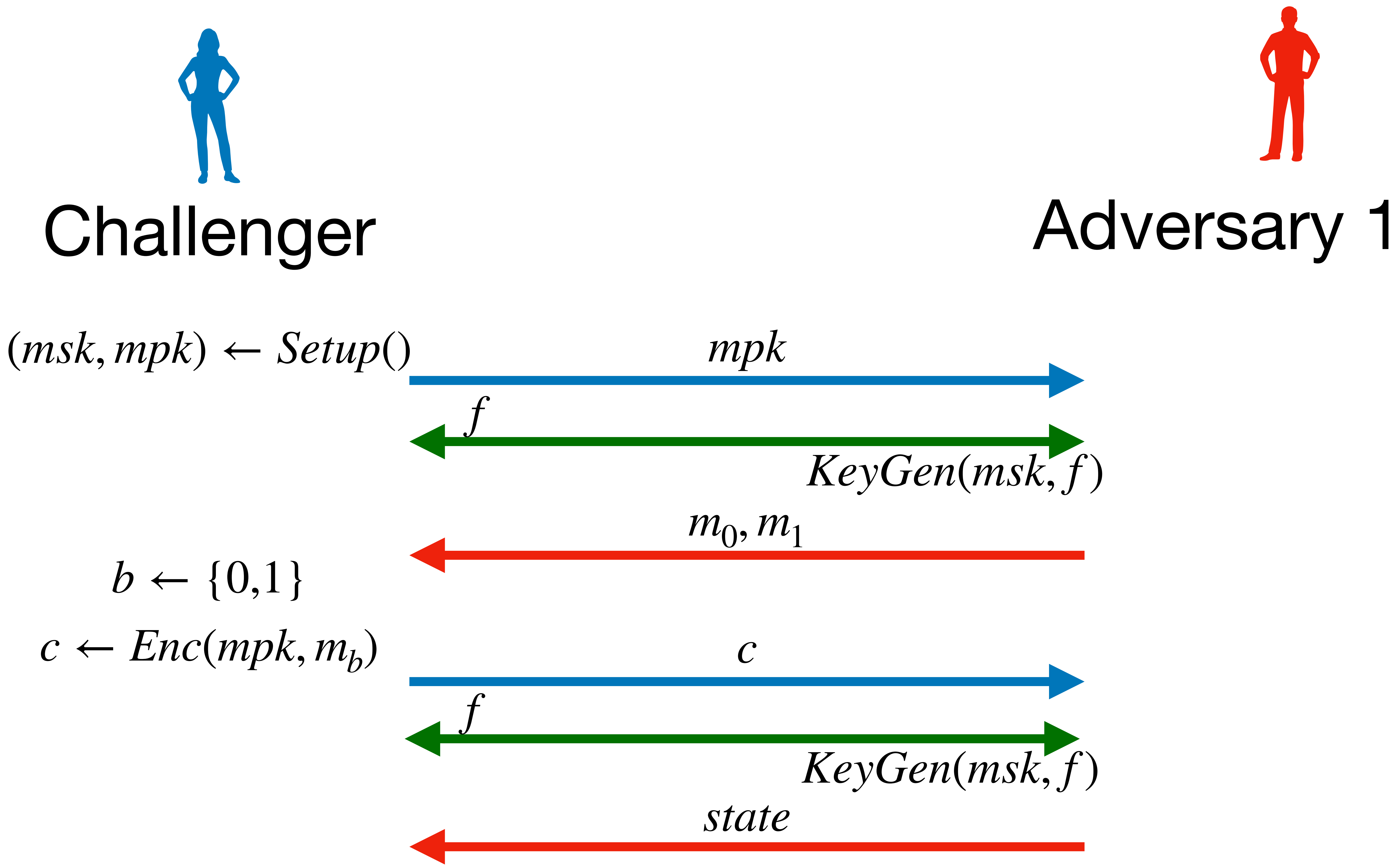
$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

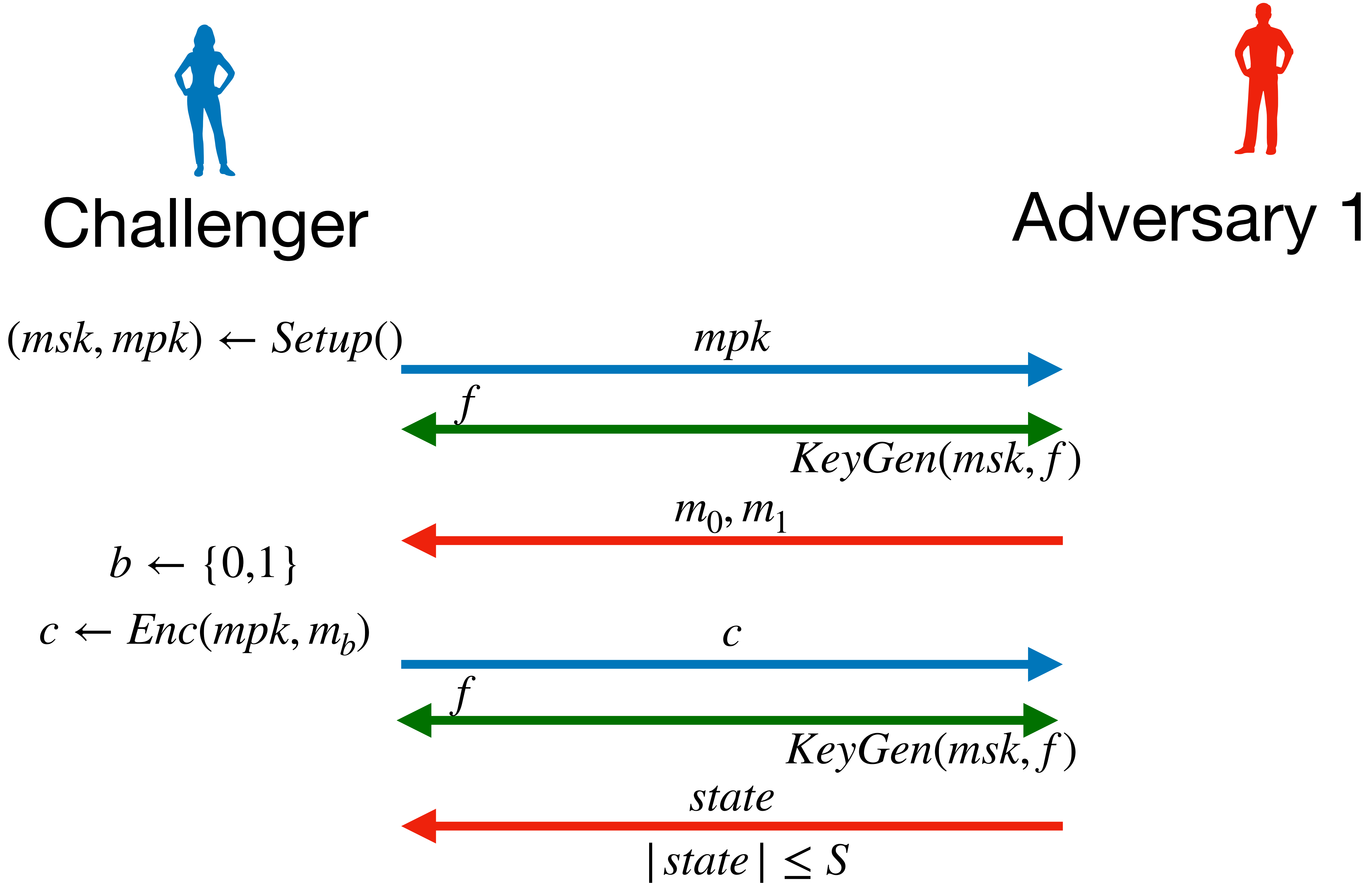
c

f

$KeyGen(msk, f)$

$state$

Regular Incompressible (FE) Security



Challenger

Adversary 1

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

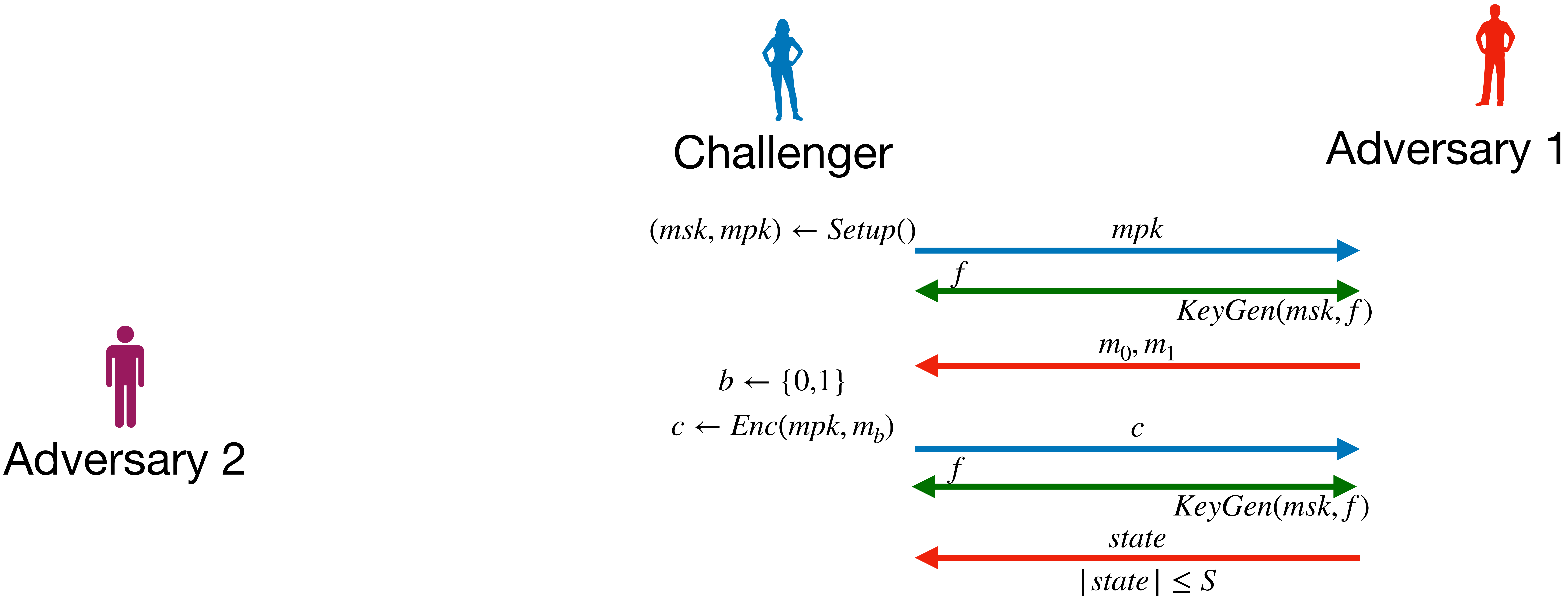
f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

Regular Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

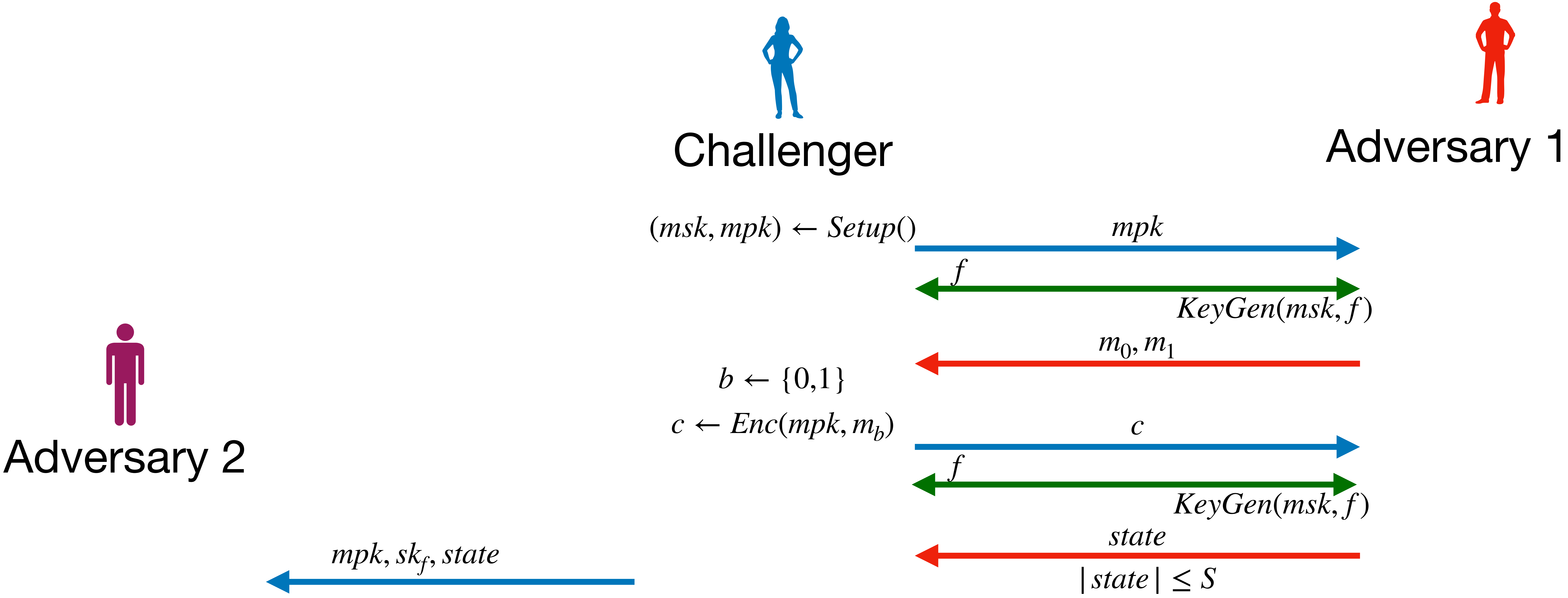
f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

Regular Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

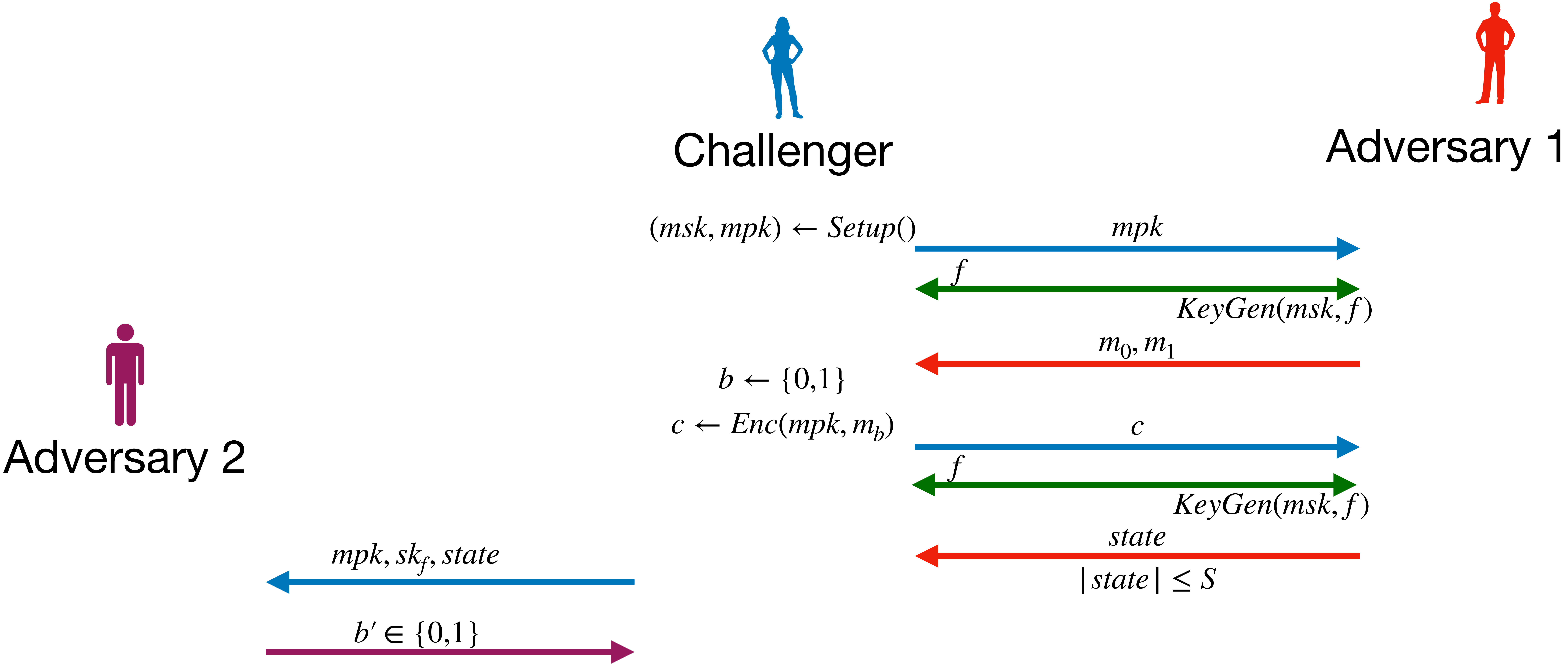
$KeyGen(msk, f)$

$state$

$|state| \leq S$

$mpk, sk_f, state$

Regular Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

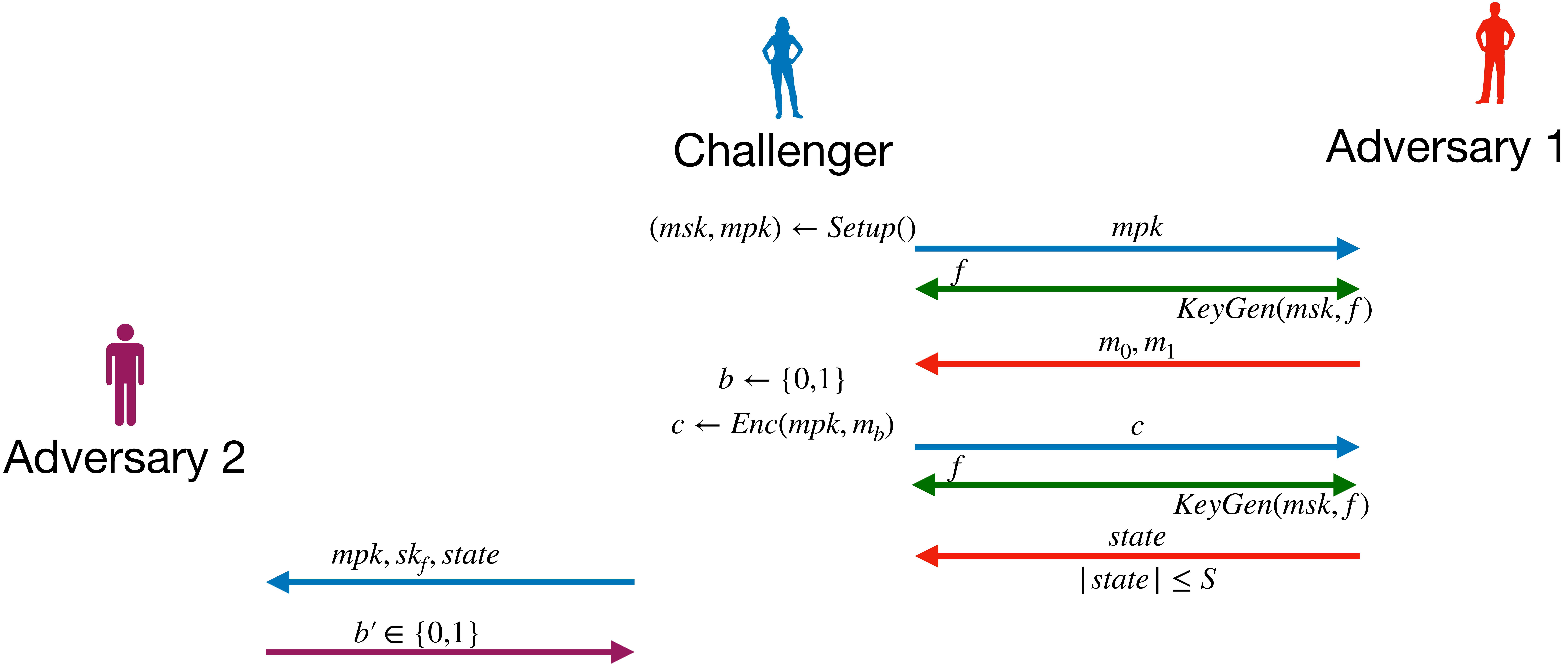
$state$

$|state| \leq S$

$mpk, sk_f, state$

$b' \in \{0,1\}$

Regular Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$ $\xrightarrow{mpk}$

f / $KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$ $\xrightarrow{c}$

f / $KeyGen(msk, f)$

$state$

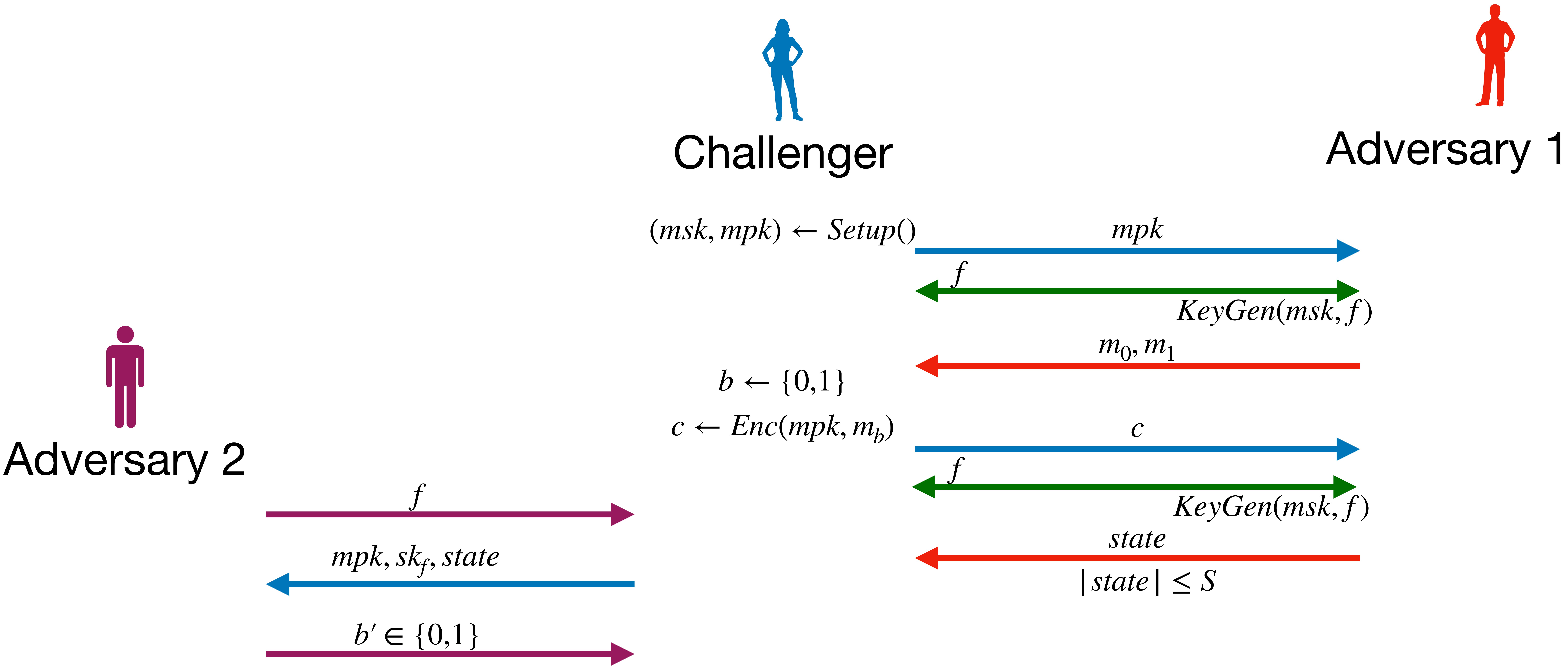
$|state| \leq S$

$mpk, sk_f, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

Regular Incompressible (FE) Security



Challenger

Adversary 1

Adversary 2

$(msk, mpk) \leftarrow Setup()$

mpk

f

$KeyGen(msk, f)$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(mpk, m_b)$

c

f

$KeyGen(msk, f)$

$state$

$|state| \leq S$

f

$mpk, sk_f, state$

$b' \in \{0,1\}$

Adversaries wins if $b = b'$

Our Results

Our Results

- Gave an incompressible FE scheme where second adversary can ask for polynomially many distinguishing keys.

Our Results

- Gave an incompressible FE scheme where second adversary can ask for polynomially many distinguishing keys.
- Construction is based on “Trojan Horse” technique.

Conclusion

Conclusion

Conclusion

- Discussed different security notion for encryption schemes.

Conclusion

- Discussed different security notion for encryption schemes.
- Focussed on constructions for incompressible SKE and PKE.

Conclusion

- Discussed different security notion for encryption schemes.
- Focussed on constructions for incompressible SKE and PKE.
- Looked at incompressible IBE & FE security definition.

Conclusion

- Discussed different security notion for encryption schemes.
- Focussed on constructions for incompressible SKE and PKE.
- Looked at incompressible IBE & FE security definition.
- Open problem : Is it possible to define incompressible version of other primitive and give a construction?

Thank You

Bounded Storage Model [Mau92]

Bounded Storage Model [Mau92]

- Adversary’s memory is bounded but not time. Honest parties communicate lot of information that the adversary cannot store them.

Bounded Storage Model [Mau92]

- Adversary’s memory is bounded but not time. Honest parties communicate lot of information that the adversary cannot store them.

- Unconditional proofs of security. Security still hold even if adversary get more memory in the later stage.

Bounded Storage Model [Mau92]

- Adversary’s memory is bounded but not time. Honest parties communicate lot of information that the adversary cannot store them.
- Unconditional proofs of security. Security still hold even if adversary get more memory in the later stage.
- Tools used are Birthday attacks and space lower bounds [Raz,FOCS17].

Bounded Storage Model [Mau92]

- Adversary’s memory is bounded but not time. Honest parties communicate lot of information that the adversary cannot store them.

- Unconditional proofs of security. Security still hold even if adversary get more memory in the later stage.

- Tools used are Birthday attacks and space lower bounds [Raz,FOCS17].

- CPA encryption [CM97,AR99,Raz17,GZ19]

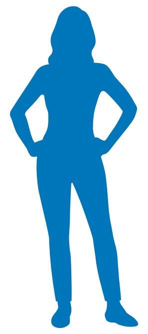
Bounded Storage Model [Mau92]

- Adversary’s memory is bounded but not time. Honest parties communicate lot of information that the adversary cannot store them.
- Unconditional proofs of security. Security still hold even if adversary get more memory in the later stage.
- Tools used are Birthday attacks and space lower bounds [Raz,FOCS17].
- CPA encryption [CM97,AR99,Raz17,GZ19]
- Key agreement [CM97,GZ19,DQW21], Commitment [DLN15,GZ19], etc.

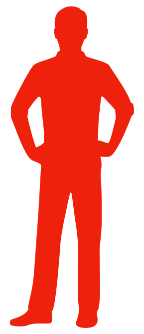
Standard Security (CCA)

Standard Security (CCA)

Standard Security (CCA)

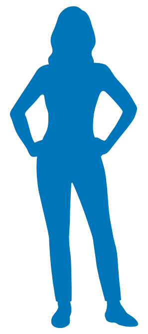


Challenger

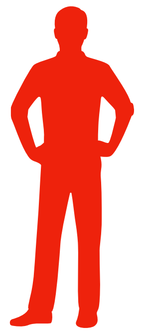


Adversary

Standard Security (CCA)



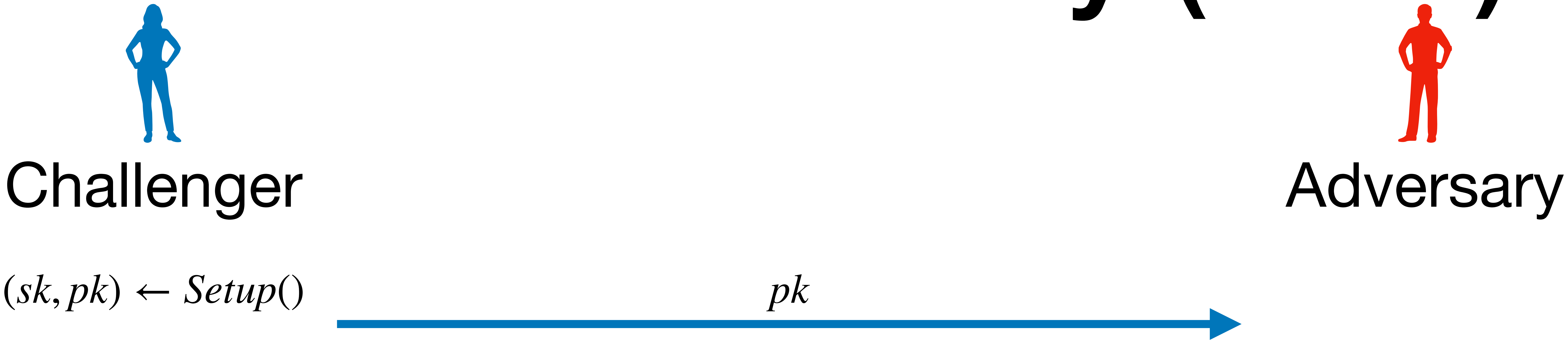
Challenger



Adversary

$(sk, pk) \leftarrow Setup()$

Standard Security (CCA)



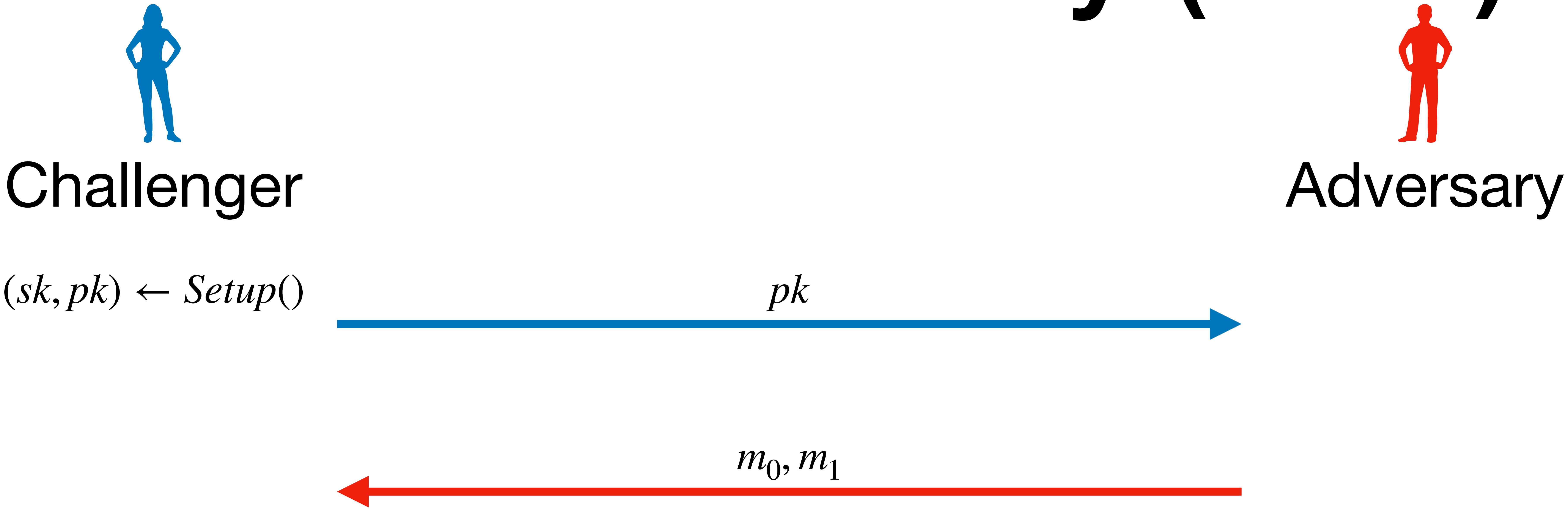
Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

Standard Security (CCA)



Challenger

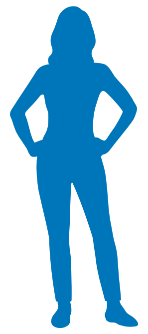
Adversary

$(sk, pk) \leftarrow Setup()$

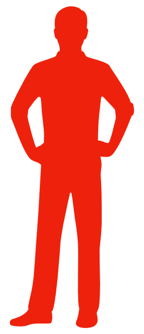
pk

m_0, m_1

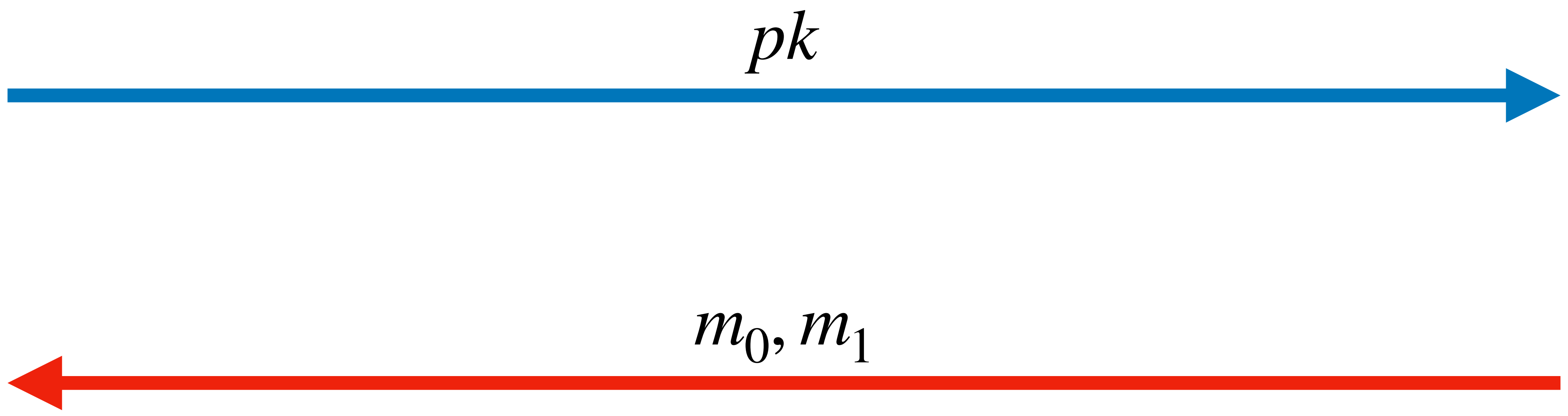
Standard Security (CCA)



Challenger



Adversary



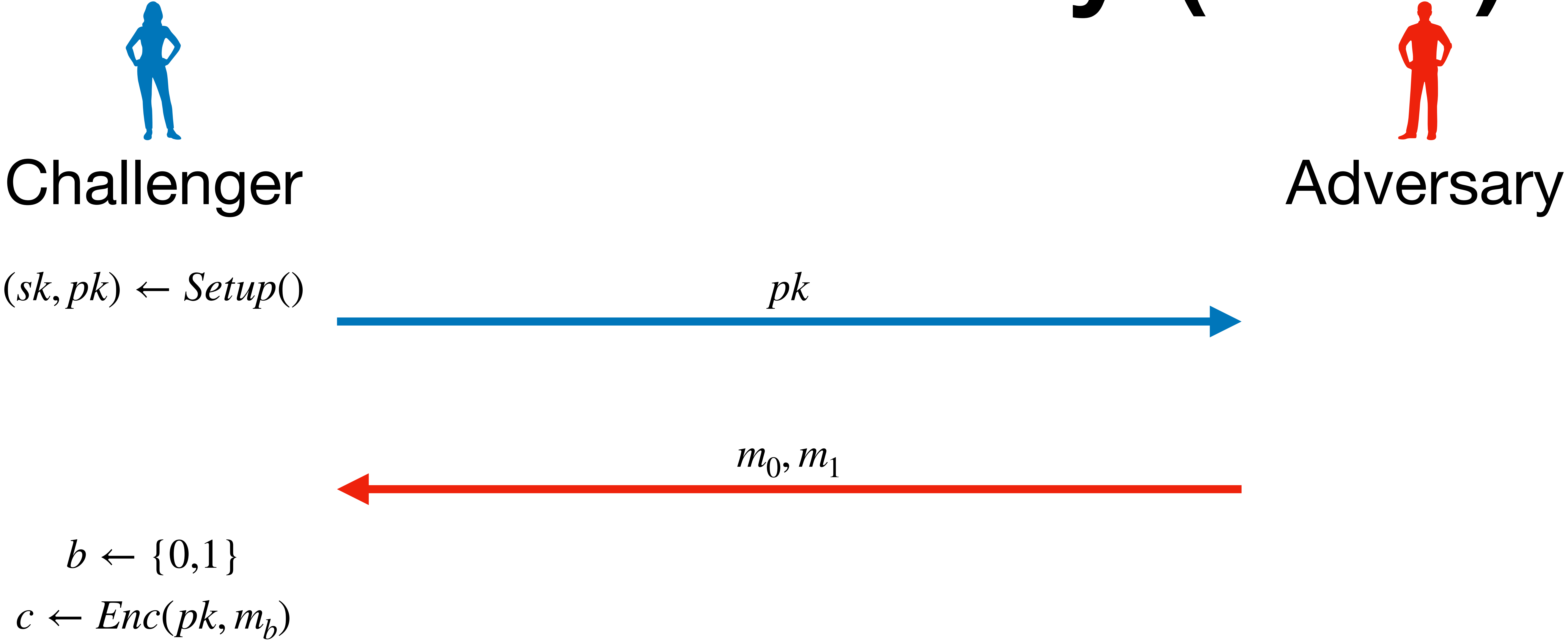
$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

$b \leftarrow \{0,1\}$

Standard Security (CCA)



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

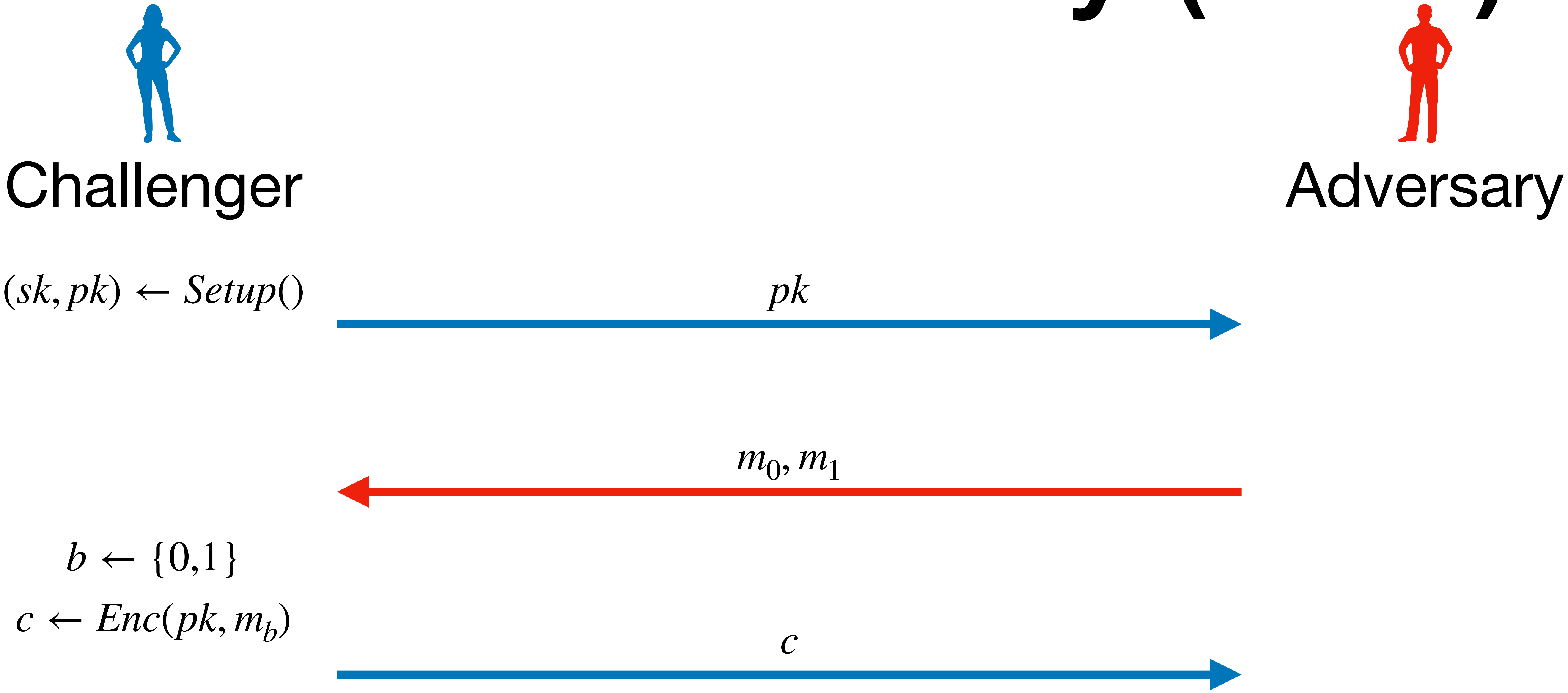
$\xrightarrow{pk}$

$\xleftarrow{m_0, m_1}$

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

Standard Security (CCA)



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

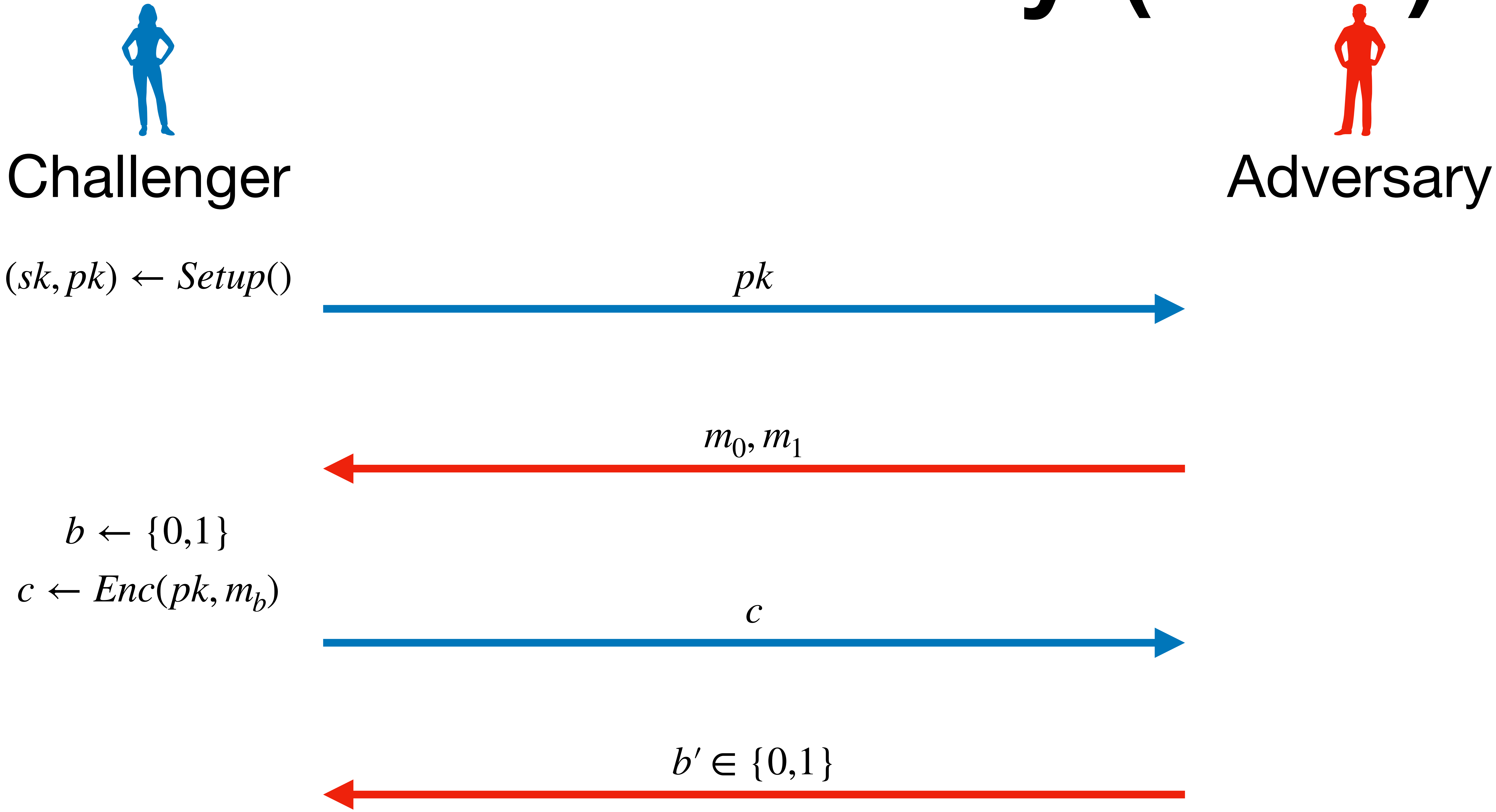
m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

Standard Security (CCA)



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

m_0, m_1

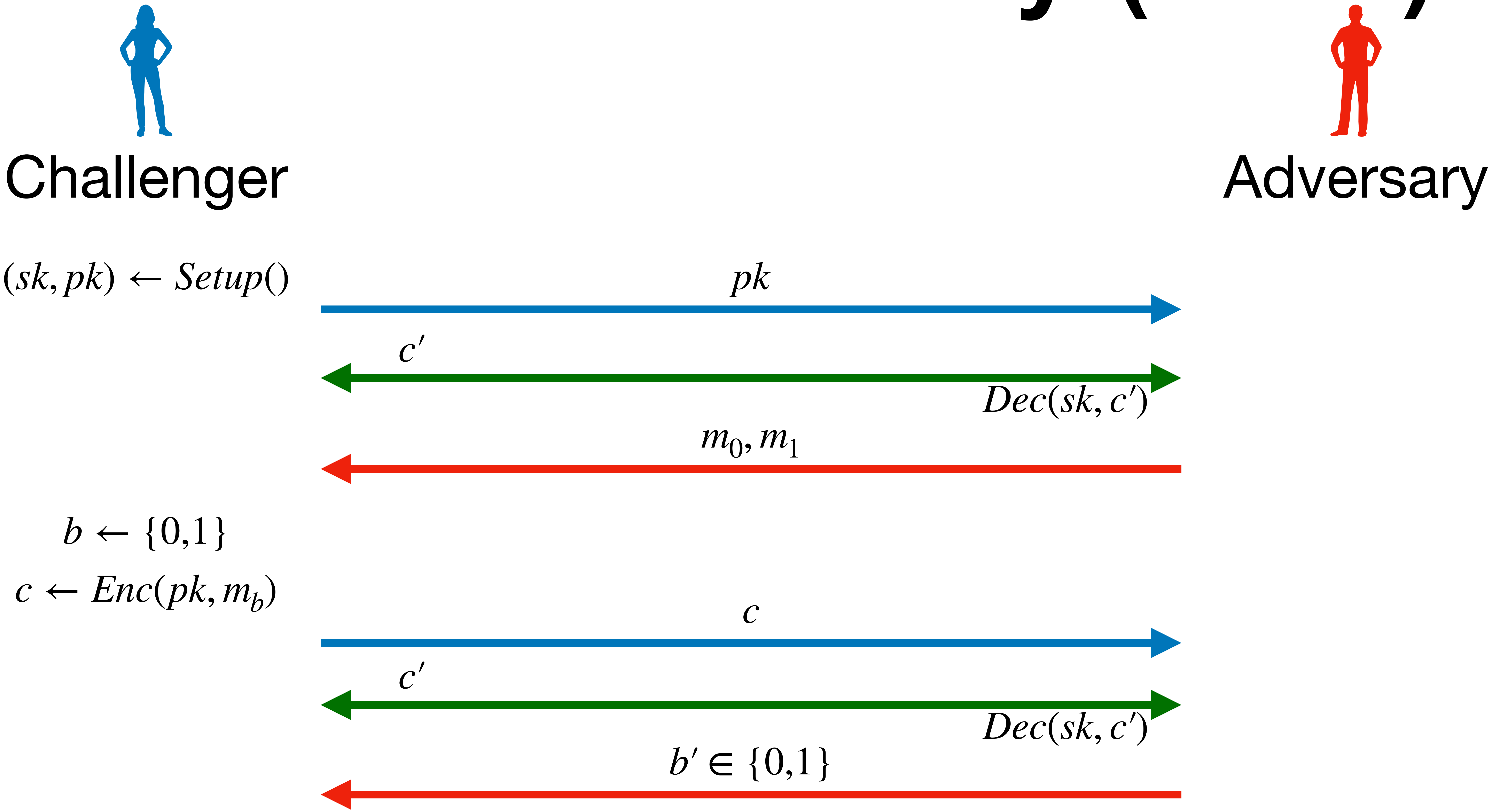
$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

$b' \in \{0,1\}$

Standard Security (CCA)



Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

c'

$Dec(sk, c')$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

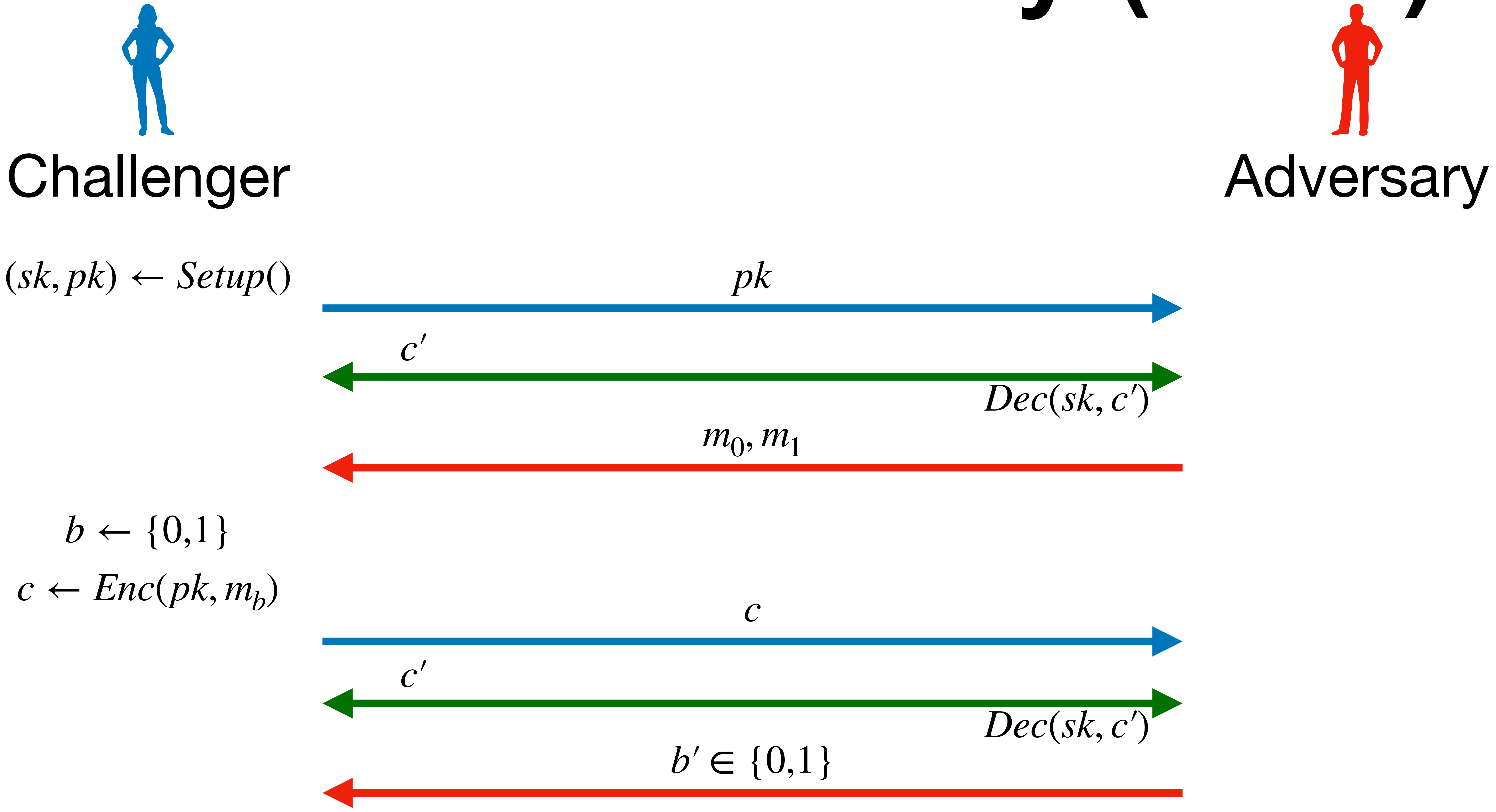
c

c'

$Dec(sk, c')$

$b' \in \{0,1\}$

Standard Security (CCA)


Challenger

Adversary

$(sk, pk) \leftarrow Setup()$

pk

c'

$Dec(sk, c')$

m_0, m_1

$b \leftarrow \{0,1\}$

$c \leftarrow Enc(pk, m_b)$

c

c'

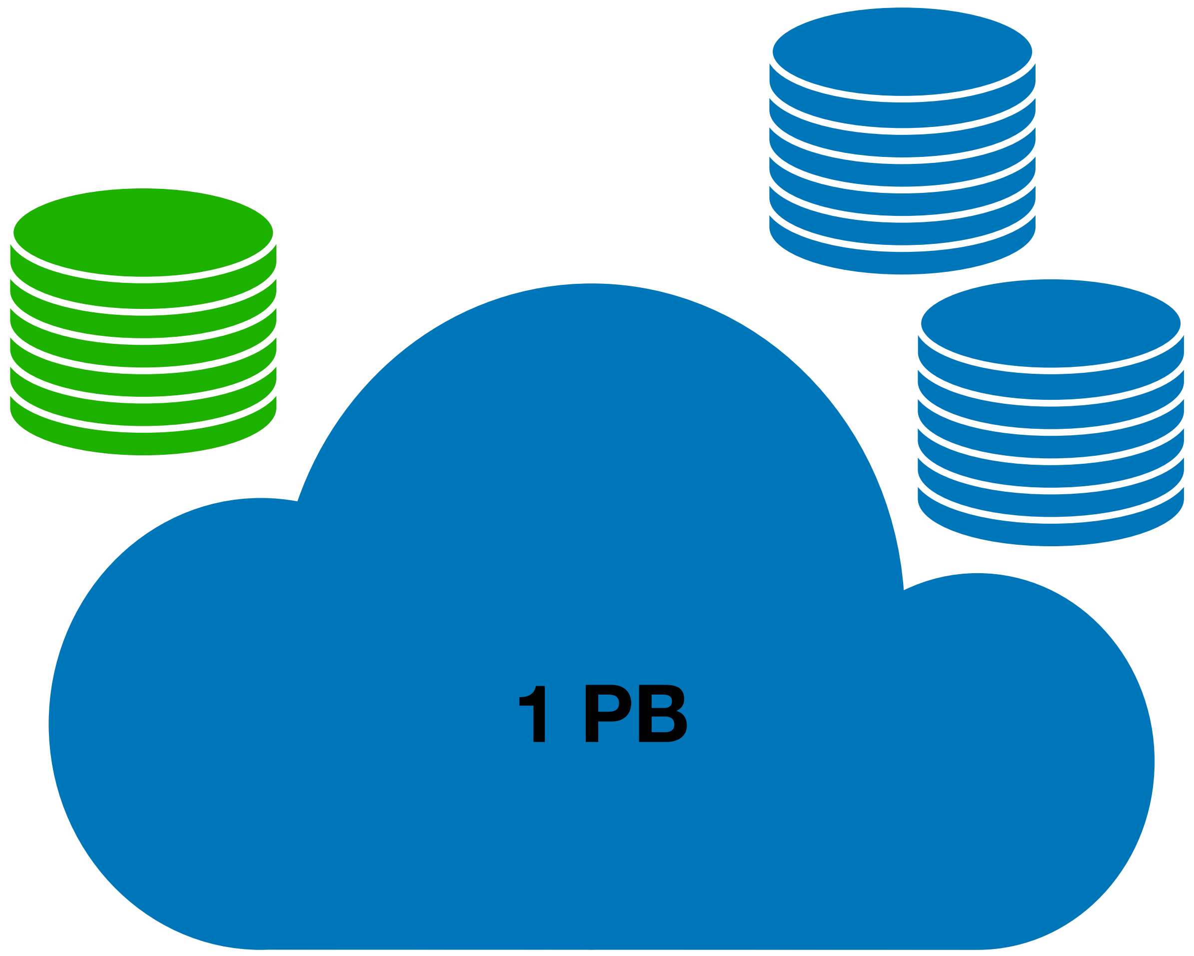
$Dec(sk, c')$

$b' \in \{0,1\}$

Adversary wins if $b = b'$

A Real Life Scenario

A Real Life Scenario



A Real Life Scenario

A Real Life Scenario

A Real Life Scenario

A Real Life Scenario

A Real Life Scenario

Rate 1 Incompressible SKE

Rate 1 Incompressible SKE

- Due to Guan et al.

Rate 1 Incompressible SKE

- Due to Guan et al.
- Uses incompressible encoding - no adversary can decode a compressed version of an encoding.

Rate 1 Incompressible SKE

- Due to Guan et al.
- Uses incompressible encoding - no adversary can decode a compressed version of an encoding.
- $Enc(sk = (crs, k), m)$:
  Compute $c_0 = Encode(crs, PRG(s) \oplus m)$.
  Set $c_1 = Ext(c_0; k) \oplus s$.
  Return (c_0, c_1).