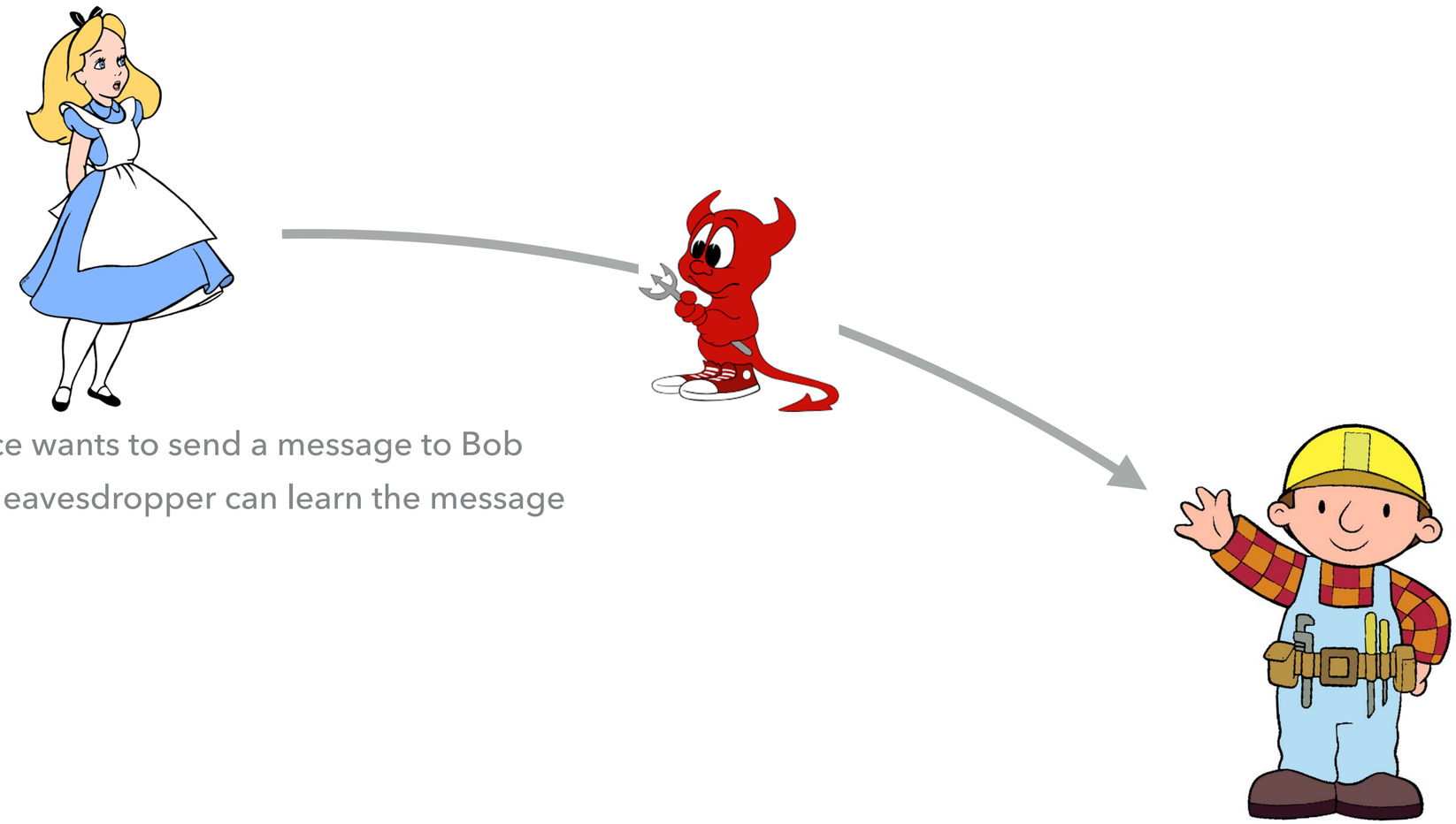
Incompressible Cryptography beyond Public Key Encryption

Mahesh Sreekumar Rajasree
IIT Delhi

Ongoing work with Rishab Goyal, Venkata Koppula, Aman Verma

SECRET KEY ENCRYPTION (SKE)

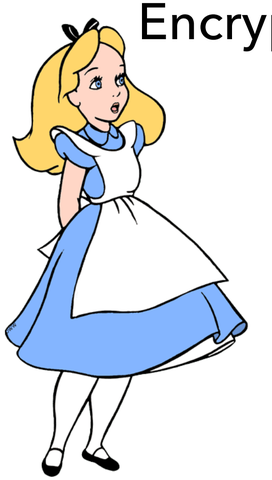
SECRET KEY ENCRYPTION (SKE)



Alice wants to send a message to Bob
s.t. no eavesdropper can learn the message

SECRET KEY ENCRYPTION (SKE)

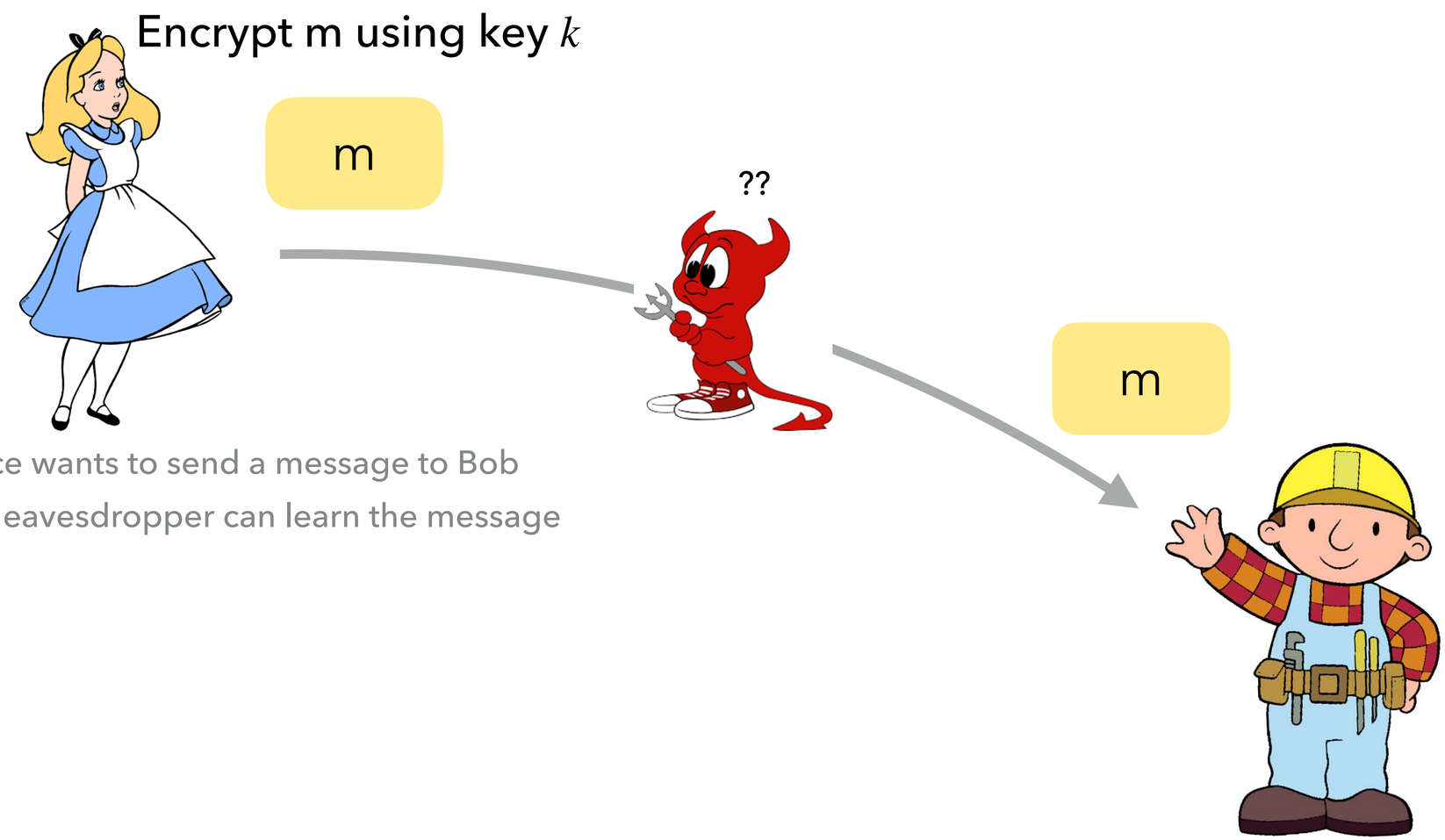
Encrypt m using key k



m

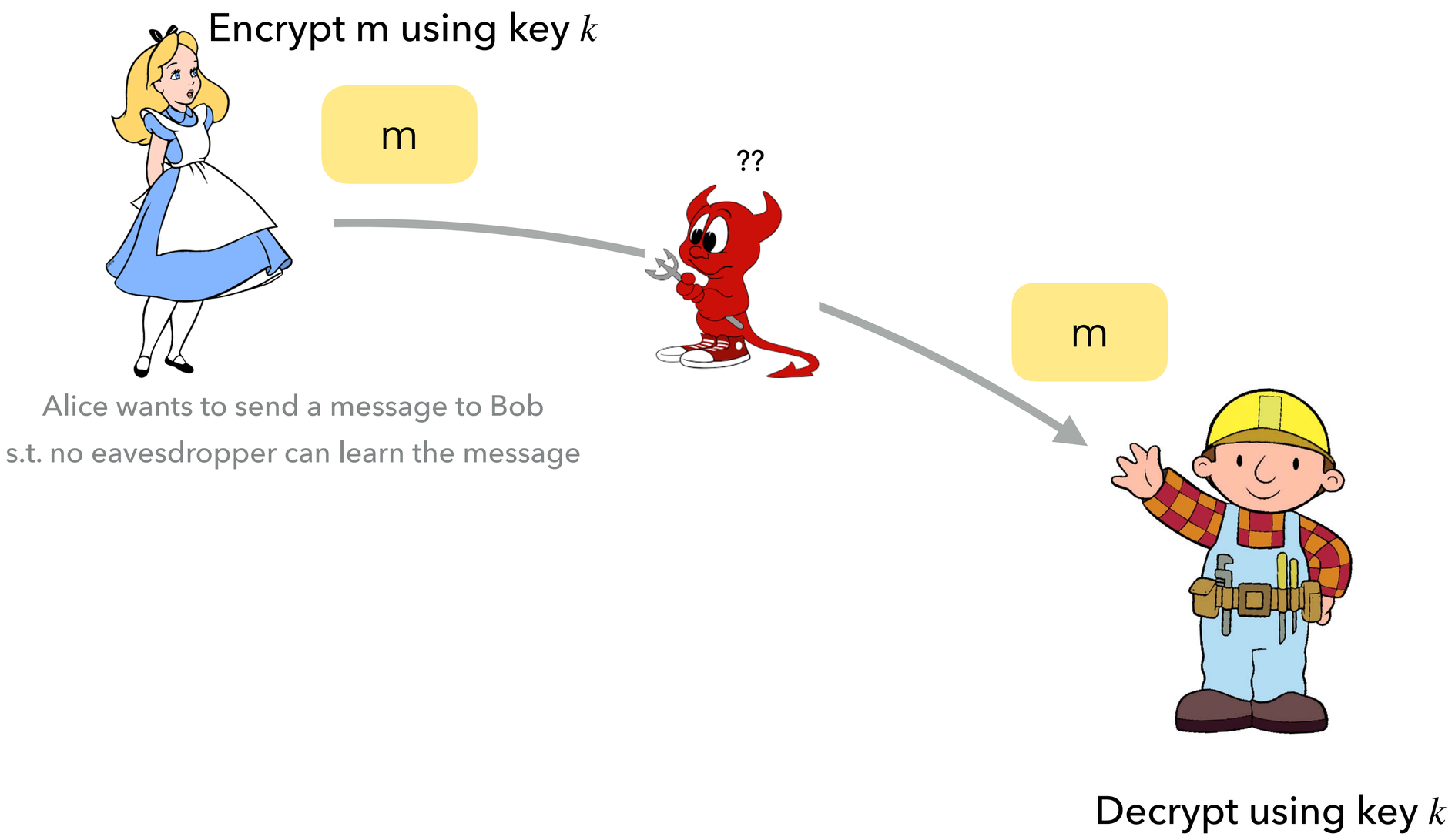
Alice wants to send a message to Bob
s.t. no eavesdropper can learn the message

SECRET KEY ENCRYPTION (SKE)



Alice wants to send a message to Bob
s.t. no eavesdropper can learn the message

SECRET KEY ENCRYPTION (SKE)

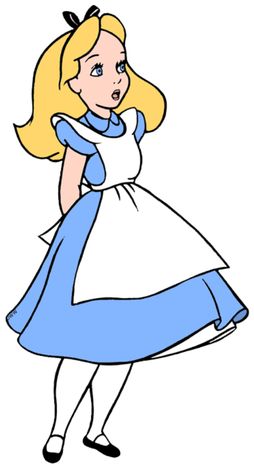


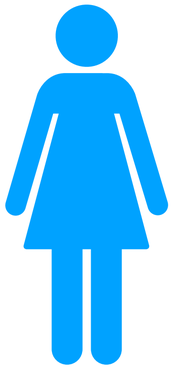
Alice wants to send a message to Bob
s.t. no eavesdropper can learn the message

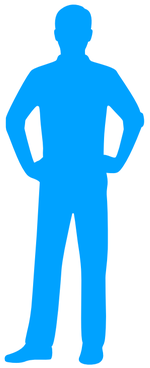
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

Parties learn only function of m



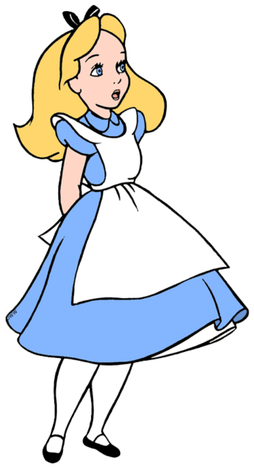




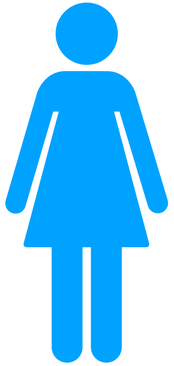
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

Parties learn only function of m



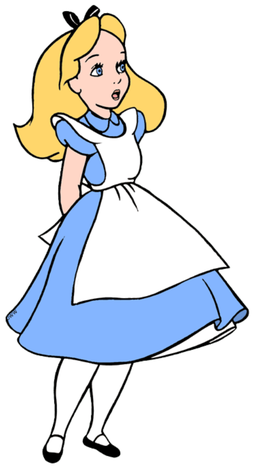
f_1



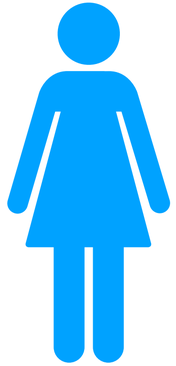
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

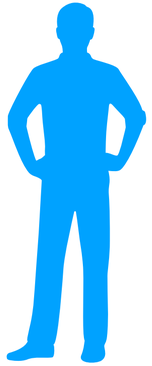
Parties learn only function of m



f_1



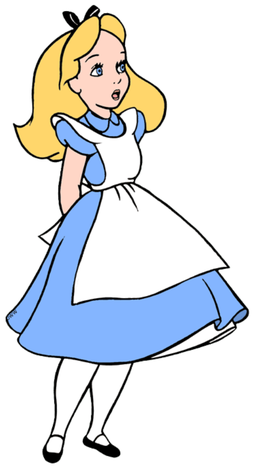
f_2



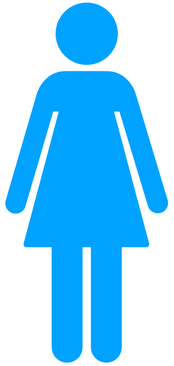
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

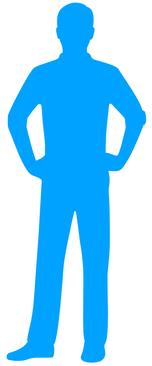
Parties learn only function of m



f_1



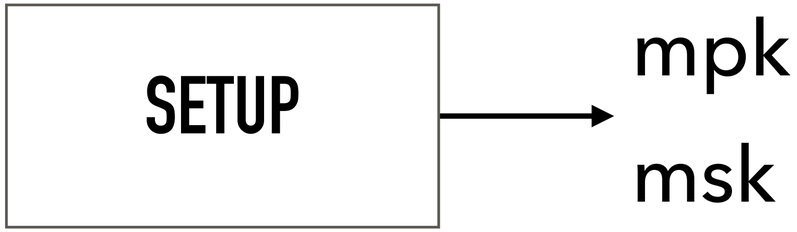
f_2



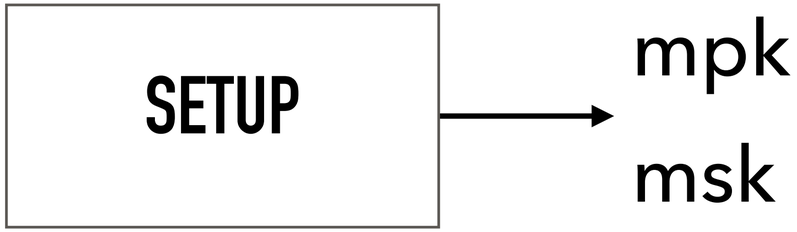
f_3

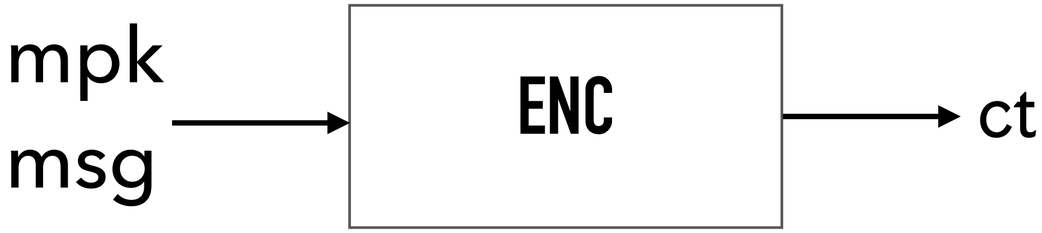
FUNCTIONAL ENCRYPTION (FE)

FUNCTIONAL ENCRYPTION (FE)

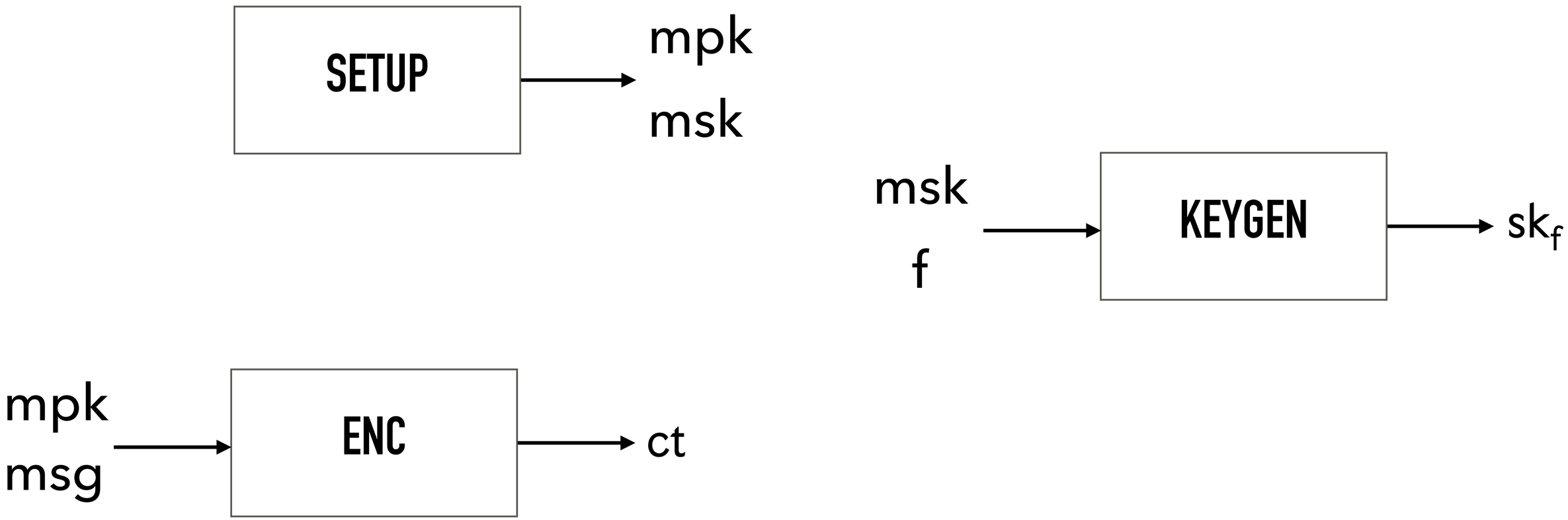


FUNCTIONAL ENCRYPTION (FE)

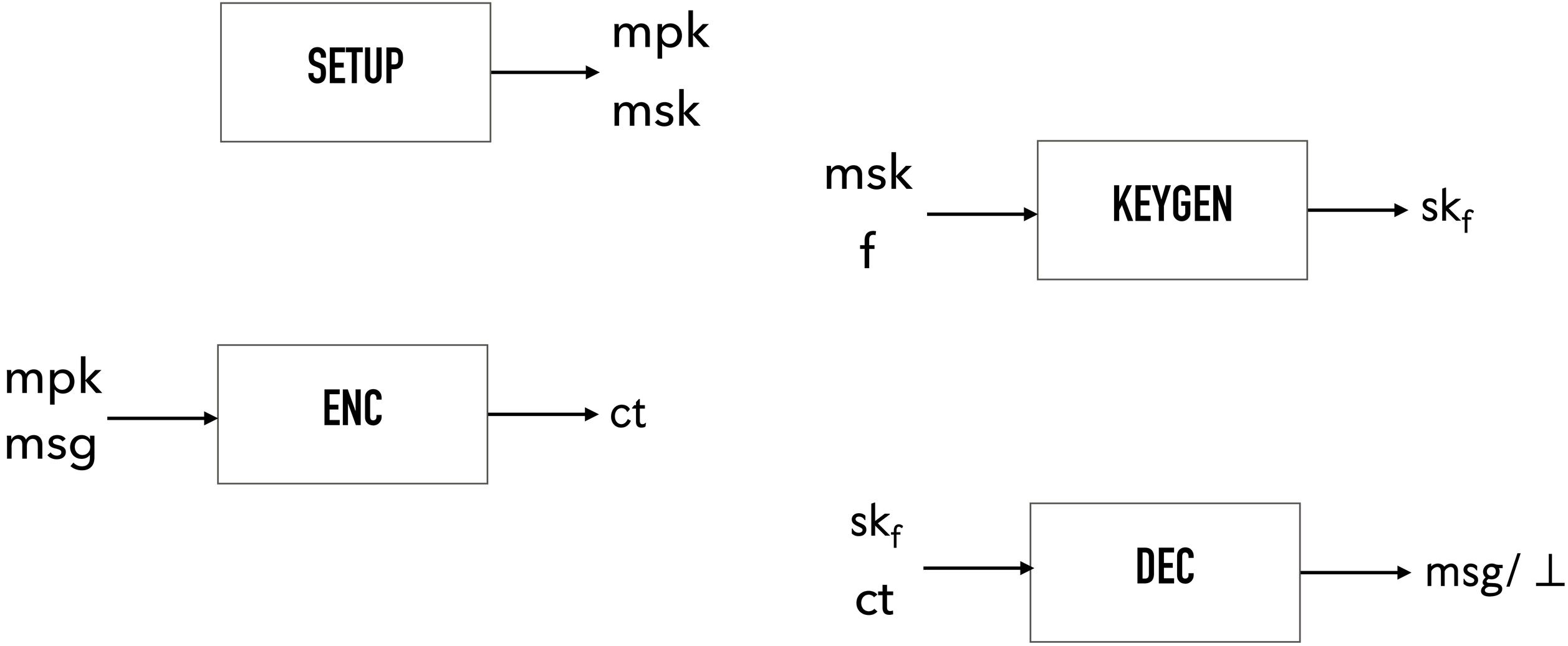




FUNCTIONAL ENCRYPTION (FE)



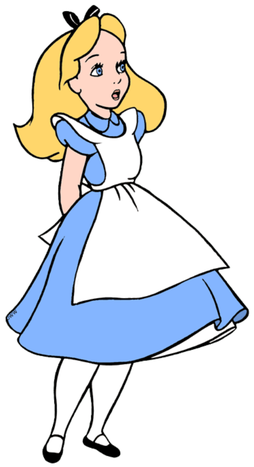
FUNCTIONAL ENCRYPTION (FE)



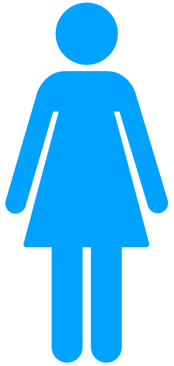
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

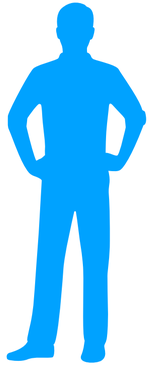
Parties learn only function of m



f_1



f_2

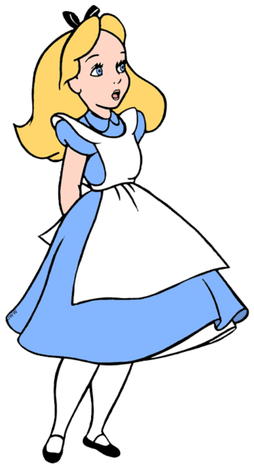


f_3

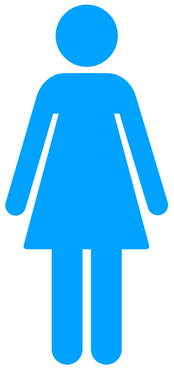
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

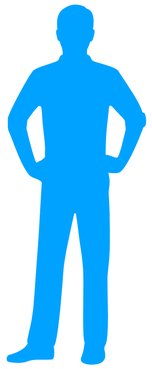
Parties learn only function of m



f_1



f_2

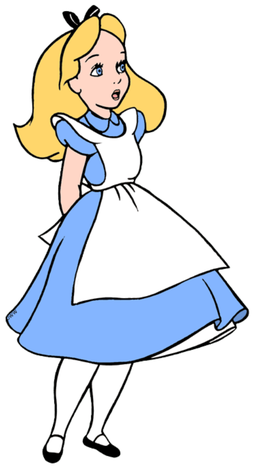


f_3

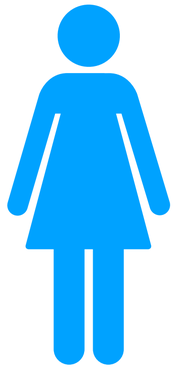
FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

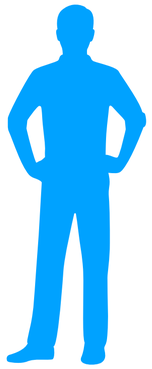
Parties learn only function of m



f_1



f_2



f_3

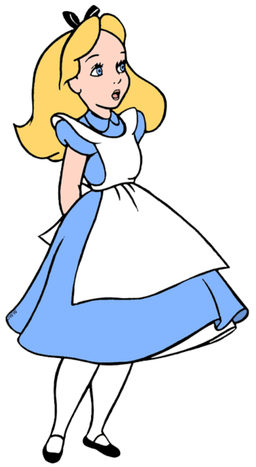
mpk

msk

FUNCTIONAL ENCRYPTION (FE)

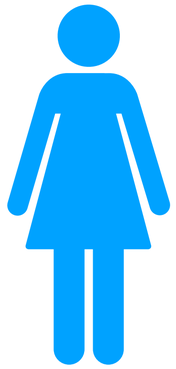
Alice wants to send m

Parties learn only function of m

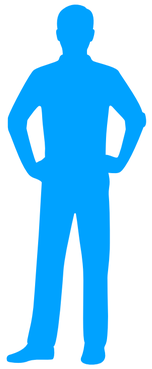


m

f_1



f_2



f_3

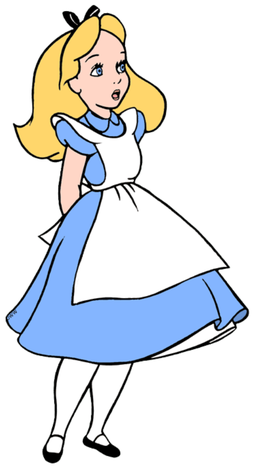
mpk

msk

FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

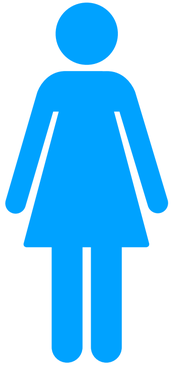
Parties learn only function of m



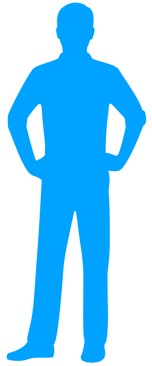
m

f_1

sk_{f_1}



f_2



f_3

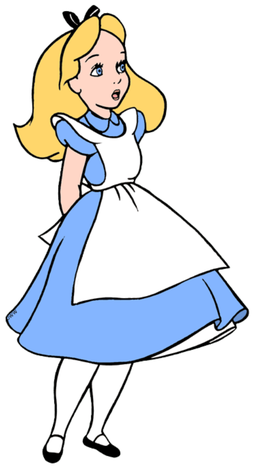
mpk

msk

FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

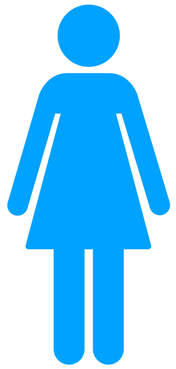
Parties learn only function of m



m

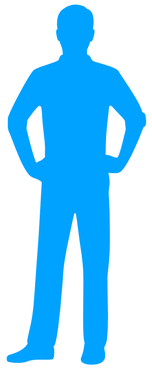
f_1

sk_{f_1}



f_2

sk_{f_2}



f_3

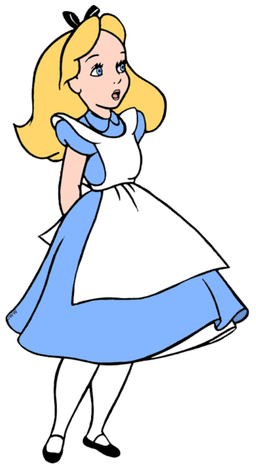
mpk

msk

FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

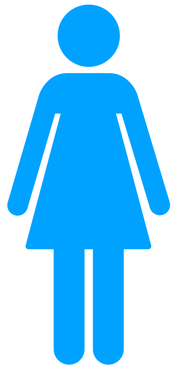
Parties learn only function of m



m

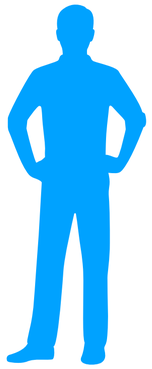
f_1

sk_{f_1}



f_2

sk_{f_2}



f_3

sk_{f_3}

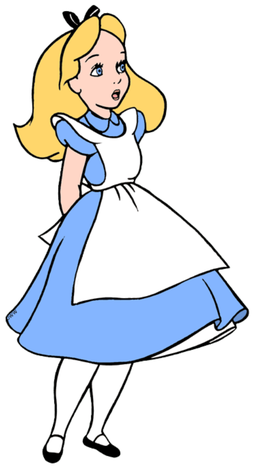
mpk

msk

FUNCTIONAL ENCRYPTION (FE)

Alice wants to send m

Parties learn only function of m



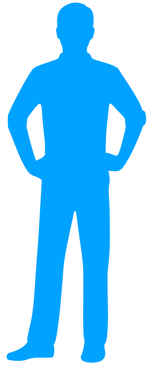
m

f_1

sk_{f_1}

f_2

sk_{f_2}



f_3

sk_{f_3}

Learns only

$f_1(m)$ and $f_2(m)$

ATTRIBUTE-BASED ENCRYPTION (ABE)

ATTRIBUTE-BASED ENCRYPTION (ABE)

<u>Example attribute:</u>

Student ✗

CSE Dept. ✓

Dean ✗

ATTRIBUTE-BASED ENCRYPTION (ABE)

Example attribute:

Student ✗

CSE Dept. ✓

Dean ✗

Example access policy:

((Student **AND** CSE Dept) **OR** Dean)

ATTRIBUTE-BASED ENCRYPTION (ABE)

Example attribute:

Student ✗

CSE Dept. ✓

Dean ✗

Example access policy:

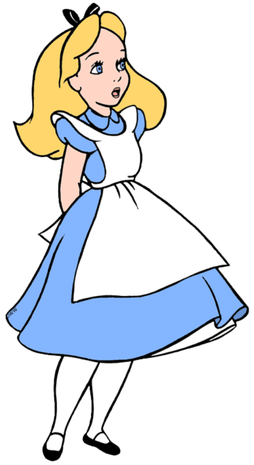
((Student **AND** CSE Dept) **OR** Dean)

Encrypt messages with 'access policy'

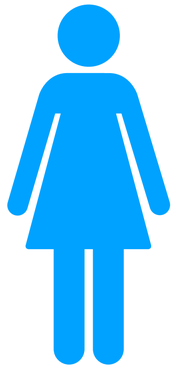
Only users having attribute satisfying access policy should learn message

ATTRIBUTE-BASED ENCRYPTION (ABE)

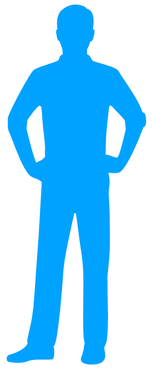
Alice wants to send m with policy f
s.t. only parties whose attributes satisfy
the policy can recover m



x_1



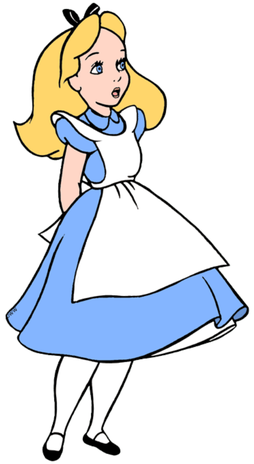
x_2



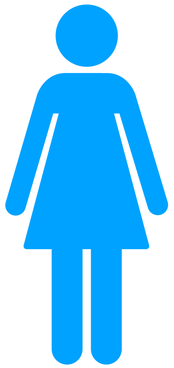
x_3

ATTRIBUTE-BASED ENCRYPTION (ABE)

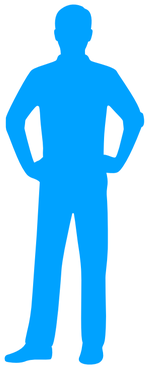
Alice wants to send m with policy f
s.t. only parties whose attributes satisfy
the policy can recover m



x_1



x_2



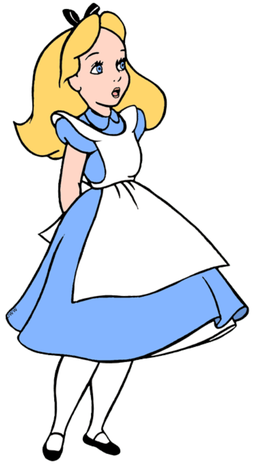
x_3

mpk

msk

ATTRIBUTE-BASED ENCRYPTION (ABE)

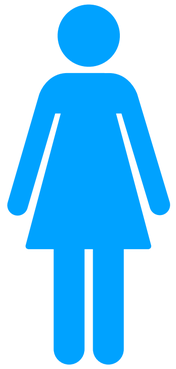
Alice wants to send m with policy f
s.t. only parties whose attributes satisfy
the policy can recover m



m; policy f

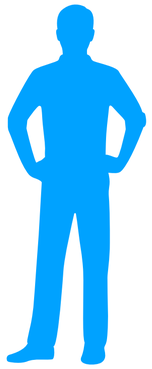
x_1

sk_{x_1}



x_2

sk_{x_2}



x_3

sk_{x_3}

mpk

msk

PROGRESS IN ABE / FE

PROGRESS IN ABE / FE

ABE

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]
- ABE for unbounded-depth circuits, succinct keys [Li, Lin, Luo 22]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]
- ABE for unbounded-depth circuits, succinct keys [Li, Lin, Luo 22]

FE

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]
- ABE for unbounded-depth circuits, succinct keys [Li, Lin, Luo 22]

FE

- FE for inner-products [Katz, Sahai, Waters 08]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]
- ABE for unbounded-depth circuits, succinct keys [Li, Lin, Luo 22]

FE

- FE for inner-products [Katz, Sahai, Waters 08]
- FE for circuits, based on obfuscation
  [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

PROGRESS IN ABE / FE

ABE

- ABE for formulas [Goyal, Pandey, Sahai, Waters 07]
- ABE for bounded-depth circuits [Gorbunov, Vaikuntanathan, Wee 12]
- ABE for bounded-depth circuits, succinct keys
  [Boneh, Gentry, Gorbunov, Halevi, Nikolaenko, Segev, Vaikuntanathan, Vinayagamurthy 13]
- ABE for unbounded-depth circuits, succinct keys [Li, Lin, Luo 22]

FE

- FE for inner-products [Katz, Sahai, Waters 08]
- FE for circuits, based on obfuscation
  [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]
- FE for circuits, based on bilinear maps + LWE
  [Jain, Lin, Sahai 20]

KEY ISSUE – KEY MANAGEMENT ISSUE

"Cryptography is a tool for turning
lots of different problems into
key management problems"

KEY ISSUE – KEY MANAGEMENT ISSUE

"Cryptography is a tool for turning lots of different problems into key management problems"

What if decryption key is compromised?

KEY ISSUE – KEY MANAGEMENT ISSUE

MICROSOFT — DATA BREACH — AI — CYBERSECURITY — NEWS

Microsoft exposed 38TB of private AI data, including passwords and secret keys

Microsoft itself warns that it is "not possible to audit the generation of SAS tokens"

ED TARGETT

September 18, 2023 . 4:10 PM — 3 min read

a tool for turning
problems into
ent problems”

What if decryption key is compromised?

KEY ISSUE – KEY MANAGEMENT ISSUE

MICROSOFT — DATA BREACH — AI — CYBERSECURITY — NEWS

Microsoft exposed 38TB of private AI data, including passwords and secret keys

Microsoft itself warns that it is "not possible to audit the generation of SAS tokens"

ED TARGETT

September 18, 2023 . 4:10 PM — 3 min read

FORBES > INNOVATION > CYBERSECURITY

EDITORS' PICK

Zoom Gets Stuffed: Here's How Hackers Got Hold Of 500,000 Passwords

Davey Winder Senior Contributor

Co-founder, Straight Talking Cyber

Follow

Apr 28, 2020, 06:46am EDT

What if decryption key is compromised?

KEY ISSUE – KEY MANAGEMENT ISSUE

KEY ISSUE – KEY MANAGEMENT ISSUE

- BIG KEY CRYPTOGRAPHY: make key so large that it is difficult for adversary to get the whole key

  [Dziembowzki 06; Di Crescenzo, Lipton 06; Bellare, Kane, Rogaway 16]

KEY ISSUE – KEY MANAGEMENT ISSUE

- BIG KEY CRYPTOGRAPHY: make key so large that it is difficult for adversary to get the whole key

  [Dziembowzki 06; Di Crescenzo, Lipton 06; Bellare, Kane, Rogaway 16]

- FORWARD SECRECY VIA KEY UPDATES: key in epoch t cannot be used to decrypt ciphertexts in earlier epochs

  [Canetti, Halevi, Katz 03; Kitagawa, Kojima, Attrapadung, Imai 15]

KEY ISSUE – KEY MANAGEMENT ISSUE

- BIG KEY CRYPTOGRAPHY: make key so large that it is difficult for adversary to get the whole key

  [Dziembowzki 06; Di Crescenzo, Lipton 06; Bellare, Kane, Rogaway 16]

- FORWARD SECRECY VIA KEY UPDATES: key in epoch t cannot be used to decrypt ciphertexts in earlier epochs

  [Canetti, Halevi, Katz 03; Kitagawa, Kojima, Attrapadung, Imai 15]

- INCOMPRESSIBLE ENCRYPTION: this talk

INCOMPRESSIBLE ENCRYPTION

INCOMPRESSIBLE ENCRYPTION

Adi Shamir

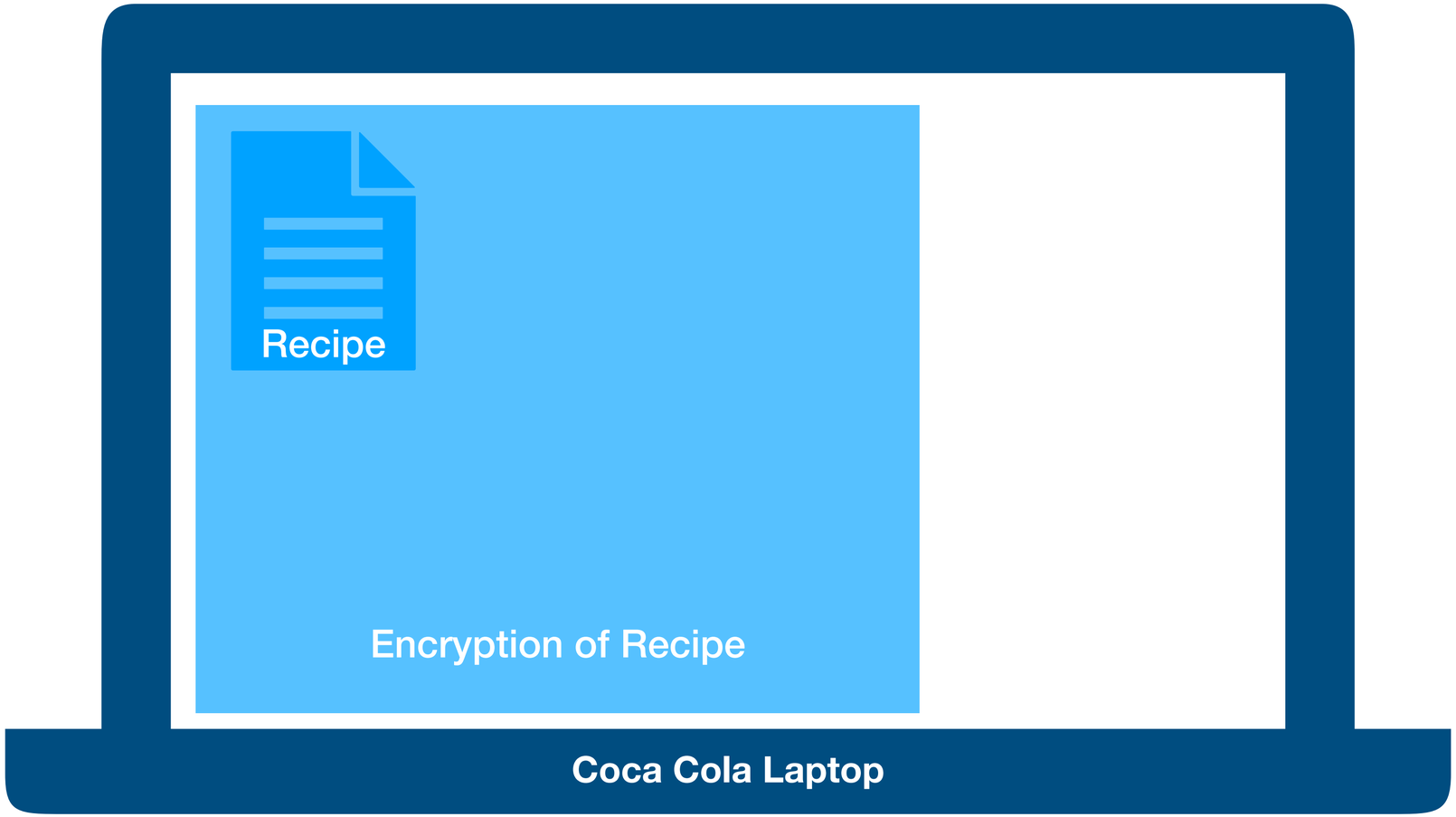
I want that the secret of the Coco-Cola company to be kept not in a tiny file of one kilobyte, which can be exfiltrated easily by an APT (Advanced Persistent Threat). I want that file to be a terabyte, which cannot be [easily] exfiltrated.

(RSA 2013 conference)

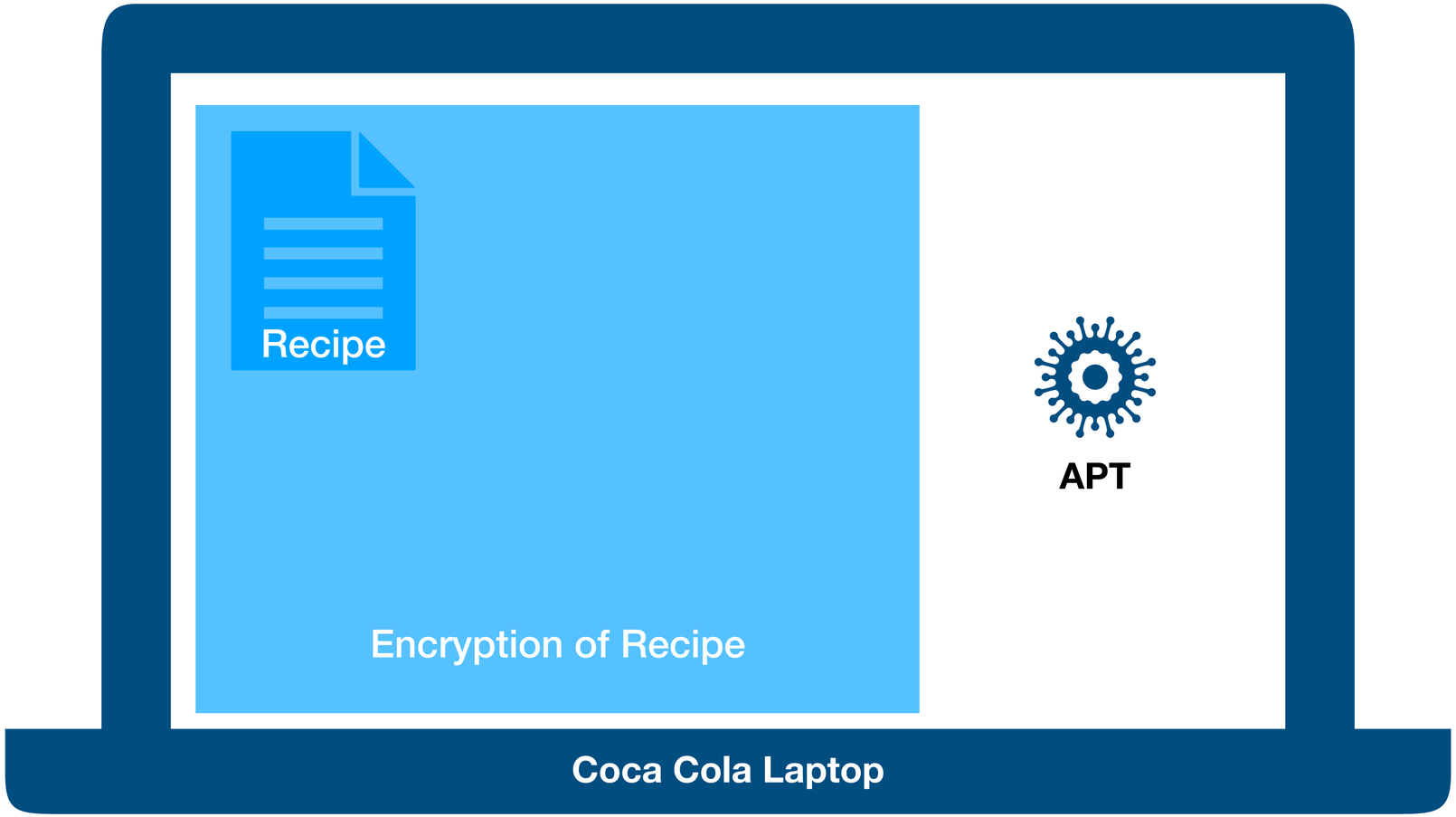
INCOMPRESSIBLE ENCRYPTION

INCOMPRESSIBLE ENCRYPTION

INCOMPRESSIBLE ENCRYPTION

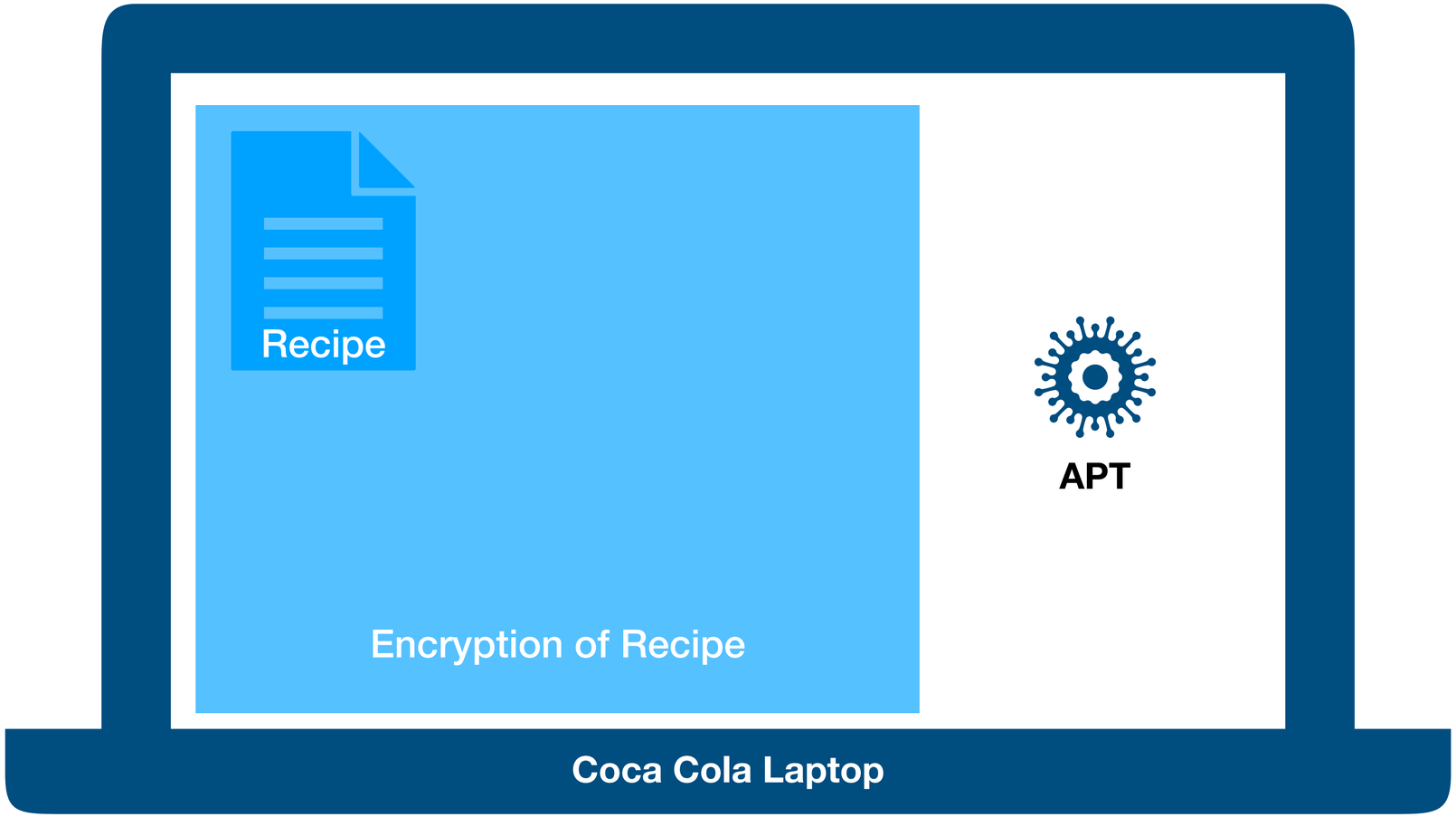


INCOMPRESSIBLE ENCRYPTION

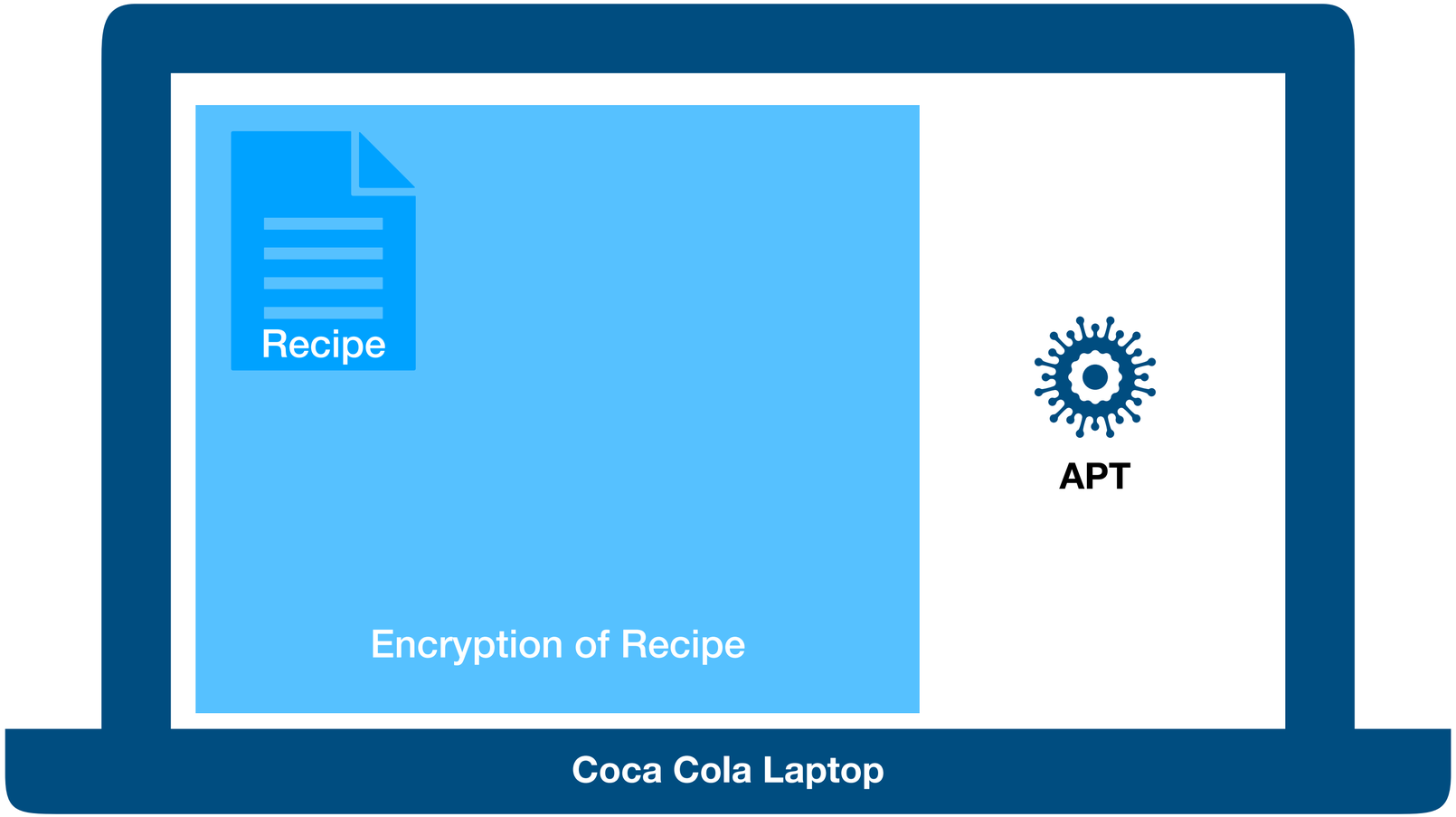


INCOMPRESSIBLE ENCRYPTION

APT can transmit a few MBs to adversary.



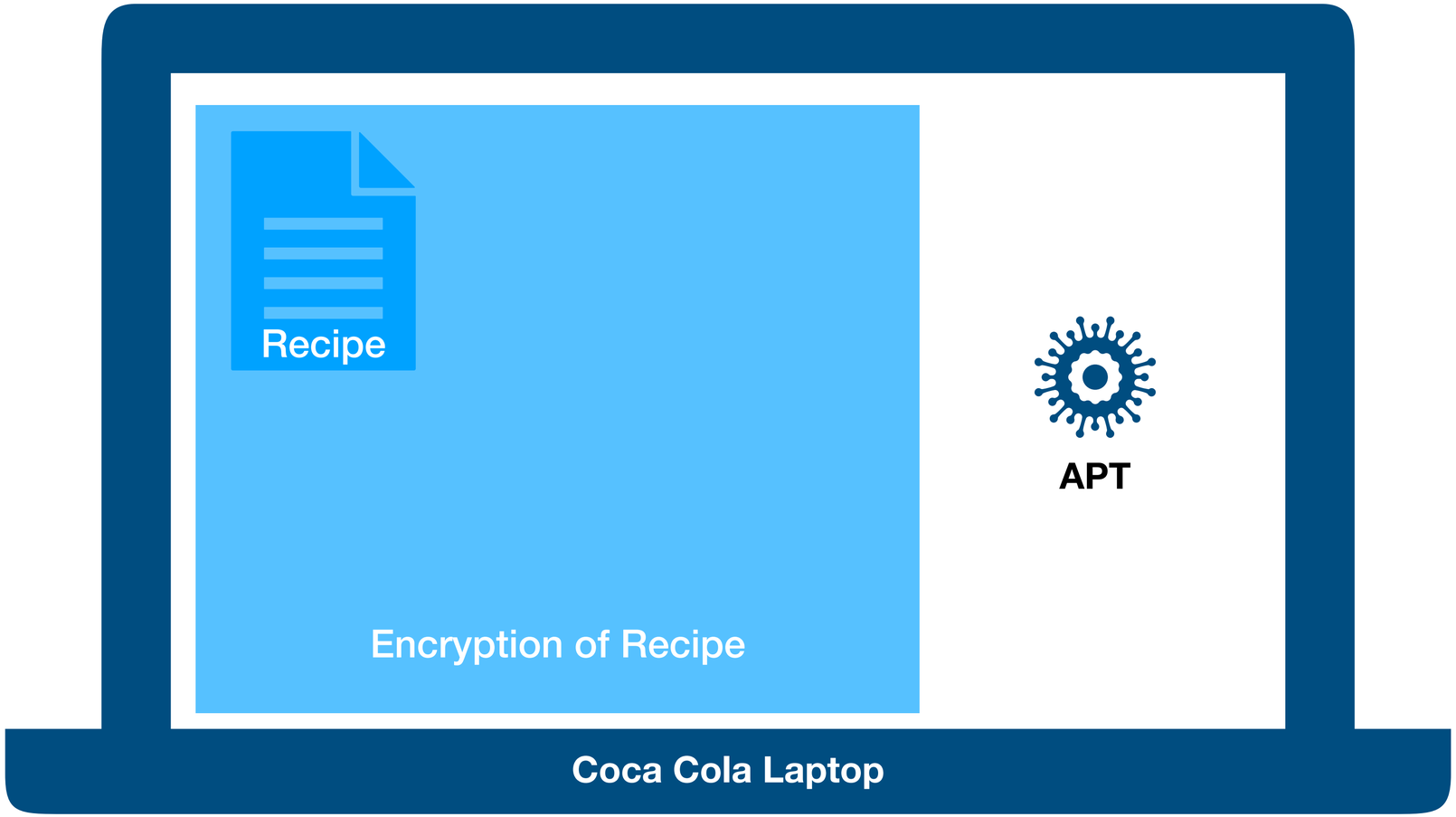
INCOMPRESSIBLE ENCRYPTION



APT can transmit a few MBs to adversary.

- Can either try to learn recipe from the ciphertext, and send the recipe

INCOMPRESSIBLE ENCRYPTION



APT can transmit a few MBs to adversary.

- Can either try to learn recipe from the ciphertext, and send the recipe
- Can send a short summary of the ciphertext. **Later, adversary learns key**, and uses this summary to learn recipe.

INCOMPRESSIBLE ENCRYPTION : A BRIEF HISTORY

INCOMPRESSIBLE ENCRYPTION : A BRIEF HISTORY

- ALL-OR-NOTHING ENCRYPTION [Rivest 97]
  Weak form of incompressible enc. for secret key enc.

INCOMPRESSIBLE ENCRYPTION : A BRIEF HISTORY

- ALL-OR-NOTHING ENCRYPTION [Rivest 97]
  Weak form of incompressible enc. for secret key enc.

- FORWARD SECURE STORAGE [Dziembowski 06]
  Incompressible Secret Key Encryption

INCOMPRESSIBLE ENCRYPTION : A BRIEF HISTORY

- ALL-OR-NOTHING ENCRYPTION [Rivest 97]
  Weak form of incompressible enc. for secret key enc.

- FORWARD SECURE STORAGE [Dziembowski 06]
  Incompressible Secret Key Encryption

- INCOMPRESSIBLE CRYPTOGRAPHY [Guan, Wichs, Zhandry 22]
  Incompressible Public Key Encryption

- RATE-1 INCOMPRESSIBLE ENCRYPTION FROM STANDARD ASSUMPTIONS
  [Branco, Dottling, Dujmovic 22]
  Efficient incompressible PKE schemes from LWE/DDH

INCOMPRESSIBLE ENCRYPTION : OUR CONTRIBUTIONS

INCOMPRESSIBLE ENCRYPTION : OUR CONTRIBUTIONS

- DEFINE INCOMPRESSIBILITY FOR IBE/ABE/FE
  Multiple definitions possible - does adversary learn a distinguishing key, or the entire master secret key?

INCOMPRESSIBLE ENCRYPTION : OUR CONTRIBUTIONS

- DEFINE INCOMPRESSIBILITY FOR IBE/ABE/FE
  Multiple definitions possible - does adversary learn a distinguishing key, or the entire master secret key?

- CONSTRUCTIONS BASED ON MINIMAL ASSUMPTIONS
  Incompressible SKE + IBE/ABE/FE –> Incompressible IBE/ABE/FE

INCOMPRESSIBLE ENCRYPTION : OUR CONTRIBUTIONS

- Define Incompressibility for IBE/ABE/FE
  Multiple definitions possible - does adversary learn a distinguishing key, or the entire master secret key?

- Constructions based on Minimal Assumptions
  Incompressible SKE + IBE/ABE/FE –> Incompressible IBE/ABE/FE

- Optimal* Rate Constructions from Standard Assumptions

***Optimality lies in the eyes of the beholder.**

PLAN FOR THE REMAINING TALK

PLAN FOR THE REMAINING TALK

- Security Definitions, and connections to other crypto primitives

PLAN FOR THE REMAINING TALK

- Security Definitions, and connections to other crypto primitives
- Incompressible SKE

PLAN FOR THE REMAINING TALK

- Security Definitions, and connections to other crypto primitives
- Incompressible SKE
- Our Incompressible PKE scheme

PLAN FOR THE REMAINING TALK

- Security Definitions, and connections to other crypto primitives
- Incompressible SKE
- Our Incompressible PKE scheme
- Conclusion and Open Questions

Security Definitions

INCOMPRESSIBLE ENCRYPTION

INCOMPRESSIBLE ENCRYPTION

Challenger

Adversary

INCOMPRESSIBLE ENCRYPTION

Challenger

Adversary

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

pk →

INCOMPRESSIBLE ENCRYPTION

Challenger		Adversary
$(sk, pk) \leftarrow Setup()$		
	$\xrightarrow{\quad pk \quad}$	
	$\xleftarrow{\quad m_0, m_1 \quad}$	
$b \leftarrow \{0,1\}$		
$c \leftarrow Enc(pk, m_b)$		
	$\xrightarrow{\quad c \quad}$	

INCOMPRESSIBLE ENCRYPTION

Challenger		Adversary
$(sk, pk) \leftarrow Setup()$		
	$\xrightarrow{pk}$	
	$\xleftarrow{m_0, m_1}$	
$b \leftarrow \{0,1\}$		
$c \leftarrow Enc(pk, m_b)$	$\xrightarrow{c}$	
	$\xleftarrow[\lvert state \rvert \leq S]{state}$	
		$state$

INCOMPRESSIBLE ENCRYPTION

Challenger		Adversary
$(sk, pk) \leftarrow Setup()$		
	$\xrightarrow{pk}$	
	$\xleftarrow{m_0, m_1}$	
$b \leftarrow \{0,1\}$		
$c \leftarrow Enc(pk, m_b)$	$\xrightarrow{c}$	
	$\xleftarrow[\lvert state \rvert \leq S]{state}$	
		$state$
	$\xrightarrow{sk}$	
	$\xleftarrow{b'}$	

wins if $b = b'$

INCOMPRESSIBLE ENCRYPTION

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext
- Eventually, gets the whole secret key. Must use the compressed ciphertext and key to recover message

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext
- Eventually, gets the whole secret key. Must use the compressed ciphertext and key to recover message

Leakage Resilient Enc

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext
- Eventually, gets the whole secret key. Must use the compressed ciphertext and key to recover message

Leakage Resilient Enc

- Adversary initially has access to whole secret key

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext
- Eventually, gets the whole secret key. Must use the compressed ciphertext and key to recover message

Leakage Resilient Enc

- Adversary initially has access to whole secret key
- Must compress secret key

INCOMPRESSIBLE ENCRYPTION

Incompressible Enc

- Adversary initially has access to whole ciphertext
- Must compress ciphertext
- Eventually, gets the whole secret key. Must use the compressed ciphertext and key to recover message

Leakage Resilient Enc

- Adversary initially has access to whole secret key
- Must compress secret key
- Eventually, gets the whole ciphertext. Must use the compressed key and ciphertext to recover message

LEAKAGE RESILIENT SECURITY

Challenger

Adversary

LEAKAGE RESILIENT SECURITY

Challenger

Adversary

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

pk →

LEAKAGE RESILIENT SECURITY

Challenger

Adversary

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

pk →

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

← f

$f(sk)$ →

LEAKAGE RESILIENT SECURITY

Challenger		Adversary
$(sk, pk) \leftarrow Setup()$	$\xrightarrow{pk}$	$\lvert f(sk) \rvert < S < \lvert sk \rvert$
	$\xleftarrow{f}$	
$b \leftarrow \{0,1\}$	$\xrightarrow{f(sk)}$	
$c \leftarrow Enc(pk, m_b)$	$\xleftarrow{m_0, m_1}$	
	$\xrightarrow{c}$	

LEAKAGE RESILIENT SECURITY

Challenger		Adversary
$(sk, pk) \leftarrow Setup()$	$\xrightarrow{pk}$	$\lvert f(sk) \rvert < S < \lvert sk \rvert$
	$\xleftarrow{f}$	
$b \leftarrow \{0,1\}$	$\xrightarrow{f(sk)}$	
$c \leftarrow Enc(pk, m_b)$	$\xleftarrow{m_0, m_1}$	
	$\xrightarrow{c}$	
	$\xleftarrow{b' \in \{0,1\}}$	

Adversary wins if $b = b'$

Incompressible SKE

ONE TIME PAD IS COMPRESSIBLE

ONE TIME PAD IS COMPRESSIBLE

- $sk \in \{0,1\}^n$
  Enc(sk, m): $ct = m \oplus sk$.

ONE TIME PAD IS COMPRESSIBLE

- $sk \in \{0,1\}^n$
  Enc(sk, m): $ct = m \oplus sk$.

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

ONE TIME PAD IS COMPRESSIBLE

- $sk \in \{0,1\}^n$
  Enc(sk, m): $ct = m \oplus sk$.

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

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

FIXING ONE TIME PAD [DZIEMBOWSKI 06]

FIXING ONE TIME PAD [DZIEMBOWSKI 06]

- Idea: use a ‘strong randomness extractor’

FIXING ONE TIME PAD [DZIEMBOWSKI 06]

- Idea: use a ‘strong randomness extractor’
- To encrypt a message m, compute $sk' = Ext(R; sk)$ which will used in OTP. Here, R is a huge random string.

FIXING ONE TIME PAD [DZIEMBOWSKI 06]

- Idea: use a ‘strong randomness extractor’
- To encrypt a message m, compute $sk' = Ext(R; sk)$ which will used in OTP. Here, R is a huge random string.
- Output $c = (R, m \oplus sk')$.

Incompressible Security

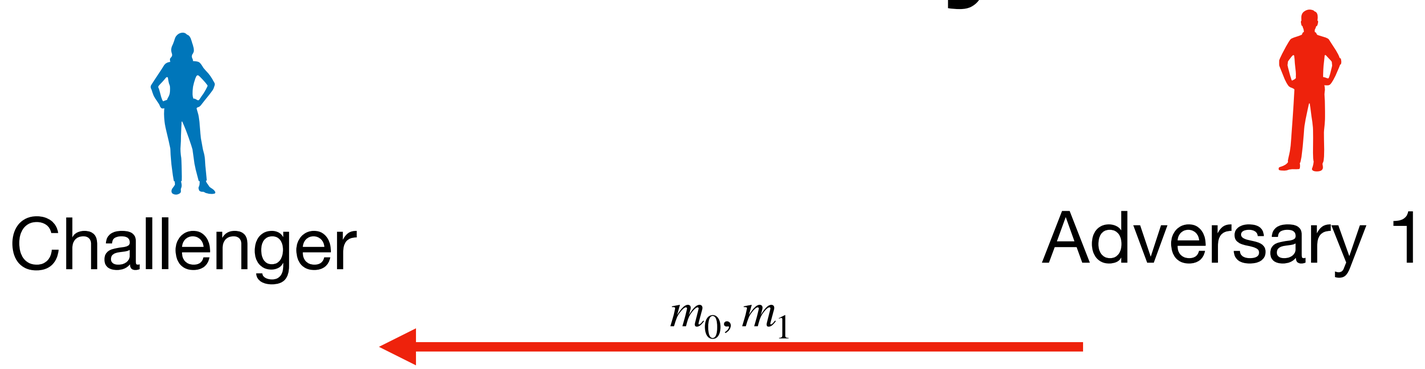
Incompressible Security

Incompressible Security

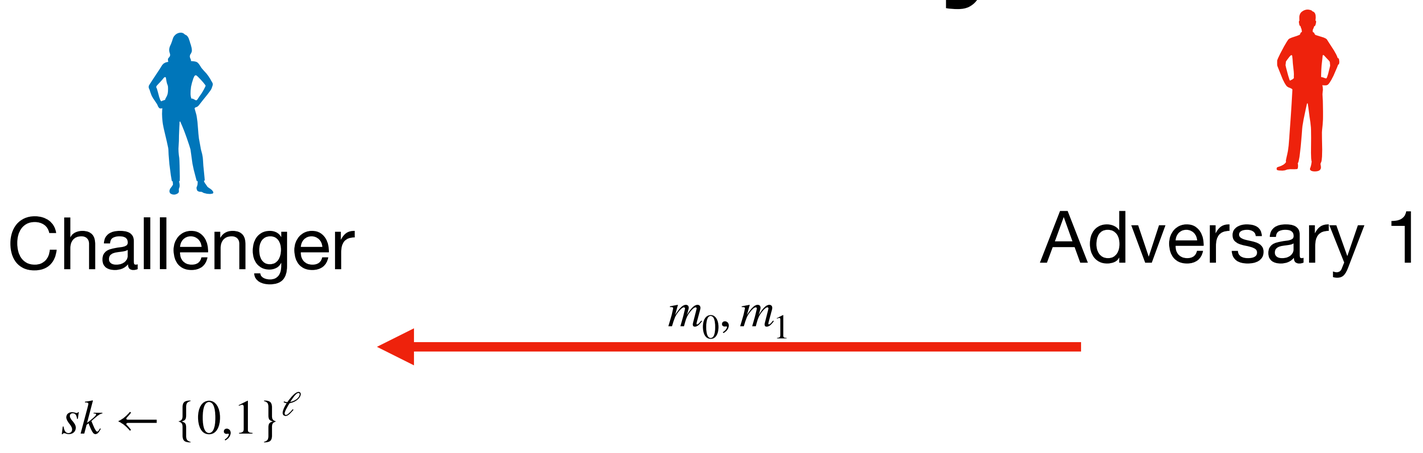
Challenger

Adversary 1

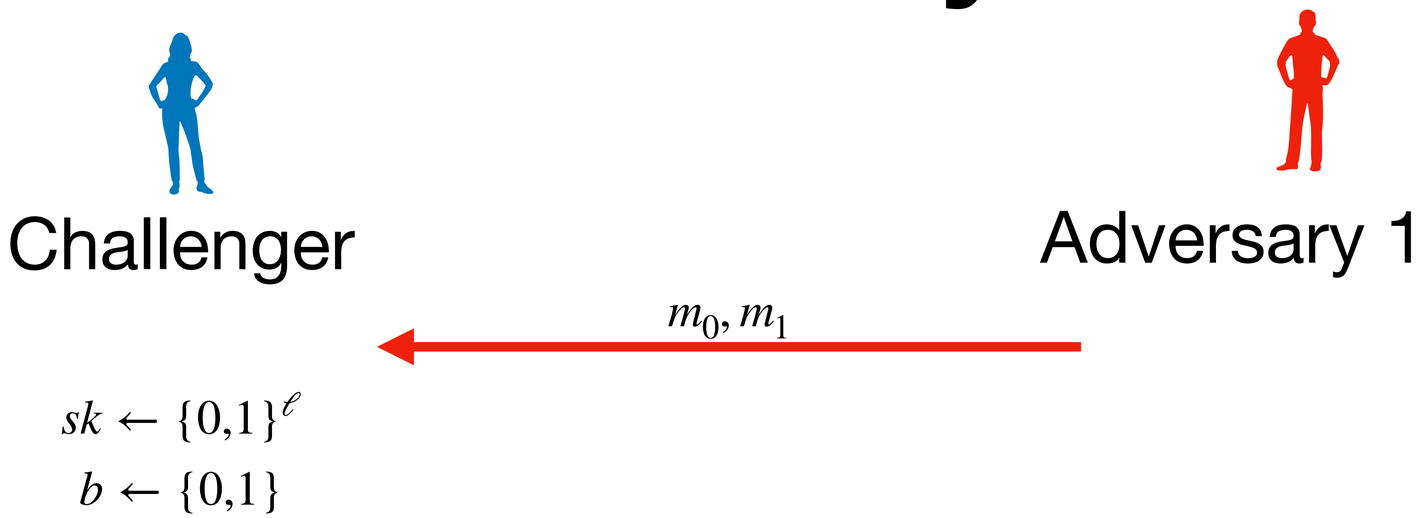
Incompressible Security



Incompressible Security



Incompressible Security



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

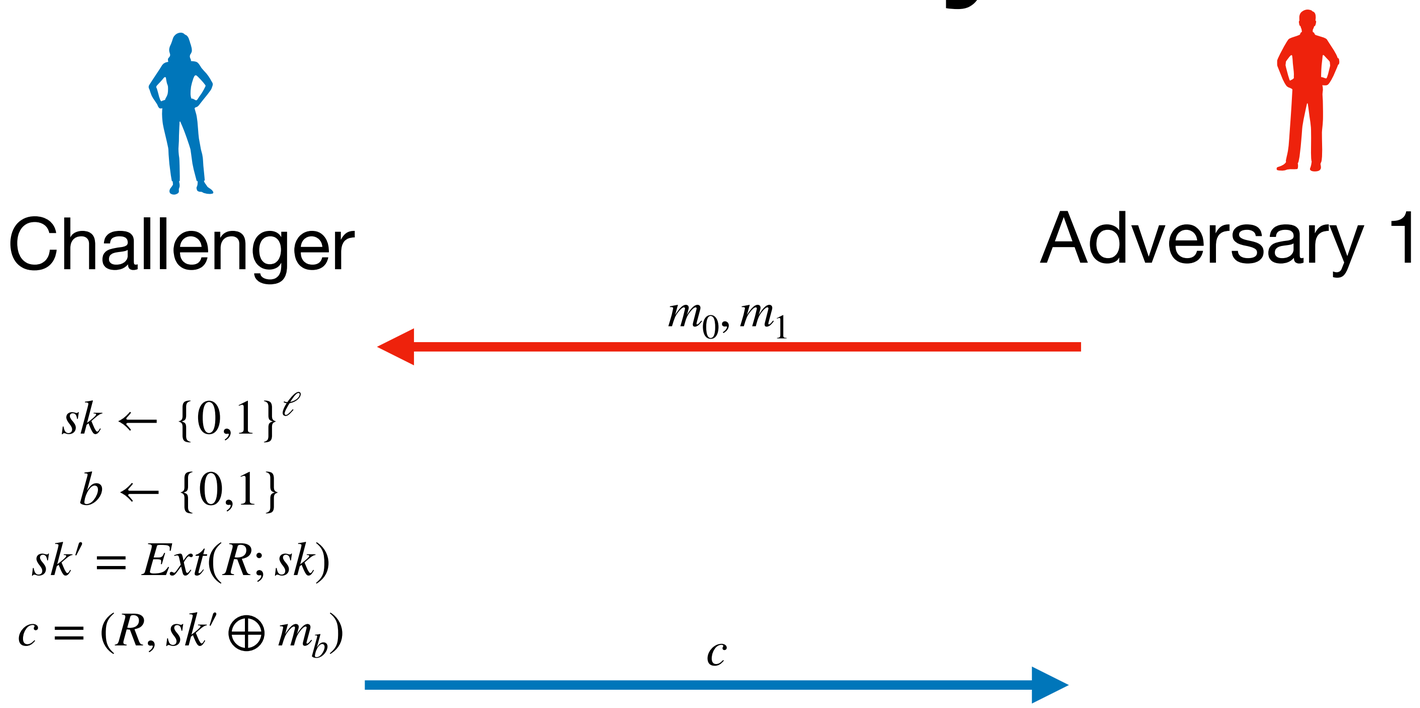
$$sk \leftarrow \{0,1\}^{\ell}$$

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

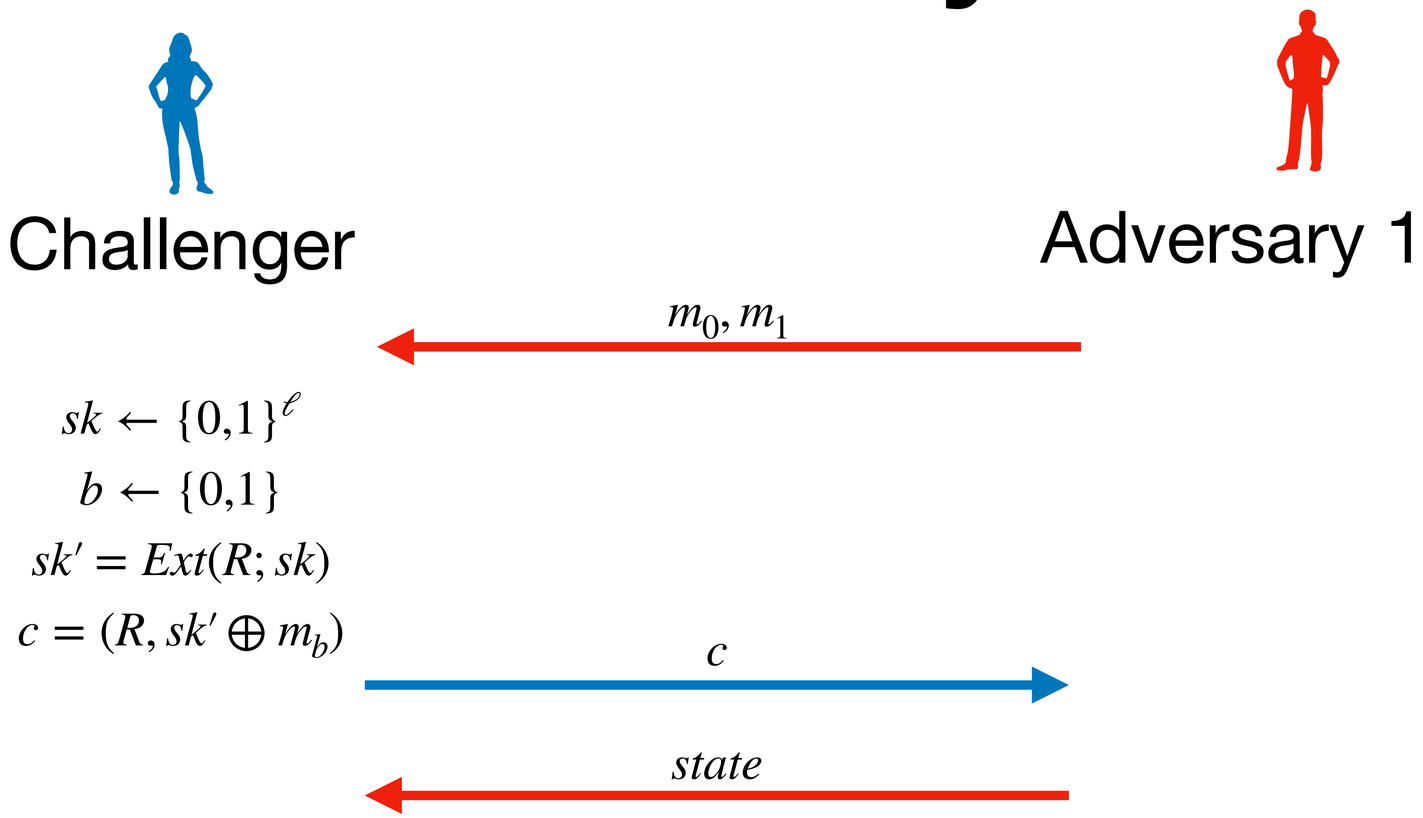
$$sk' = Ext(R; sk)$$

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

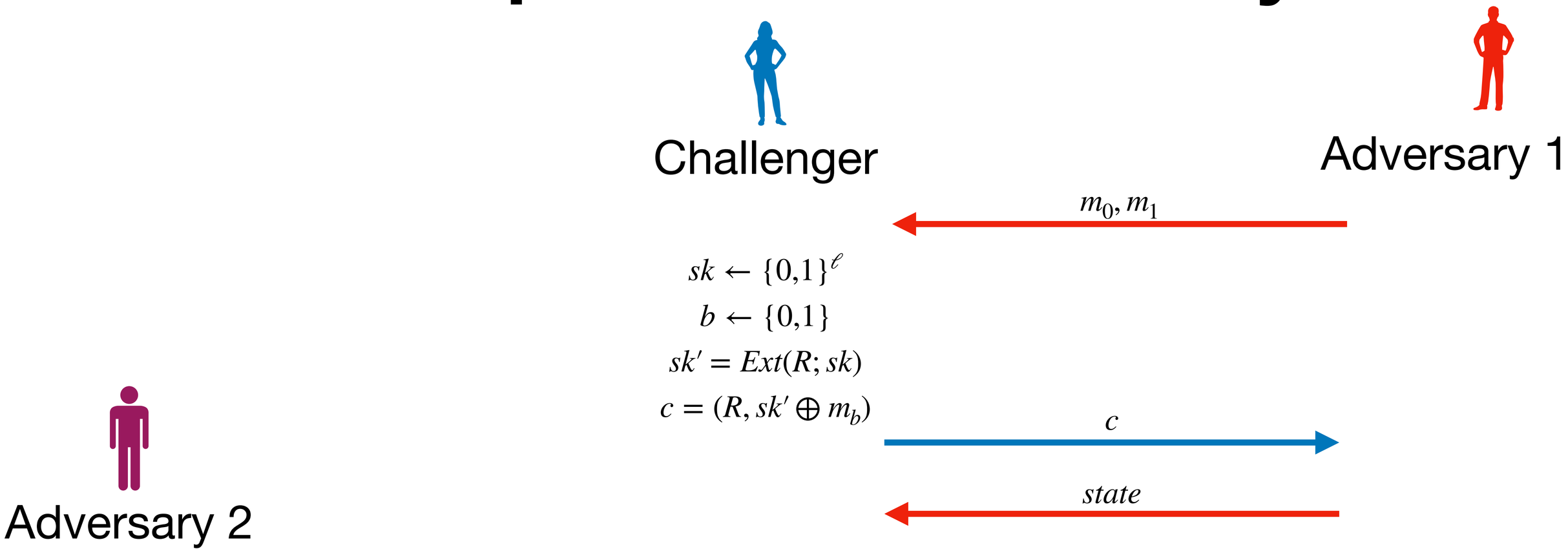
Incompressible Security



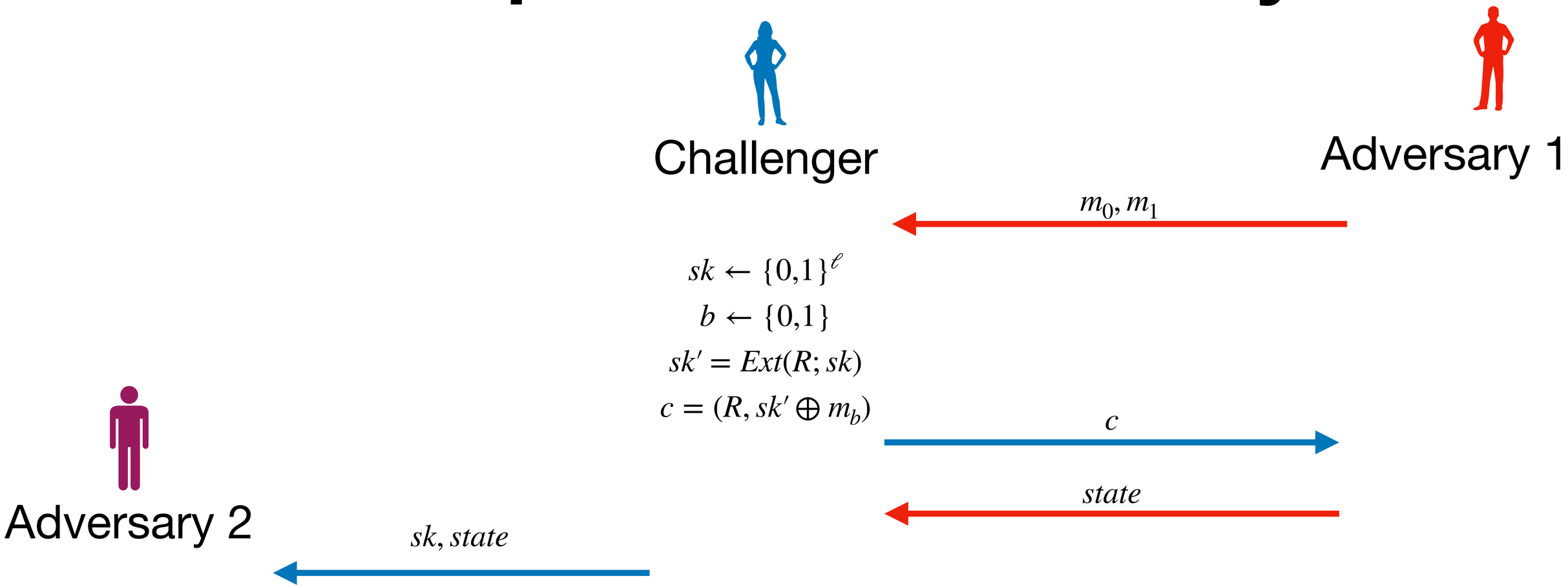
Incompressible Security



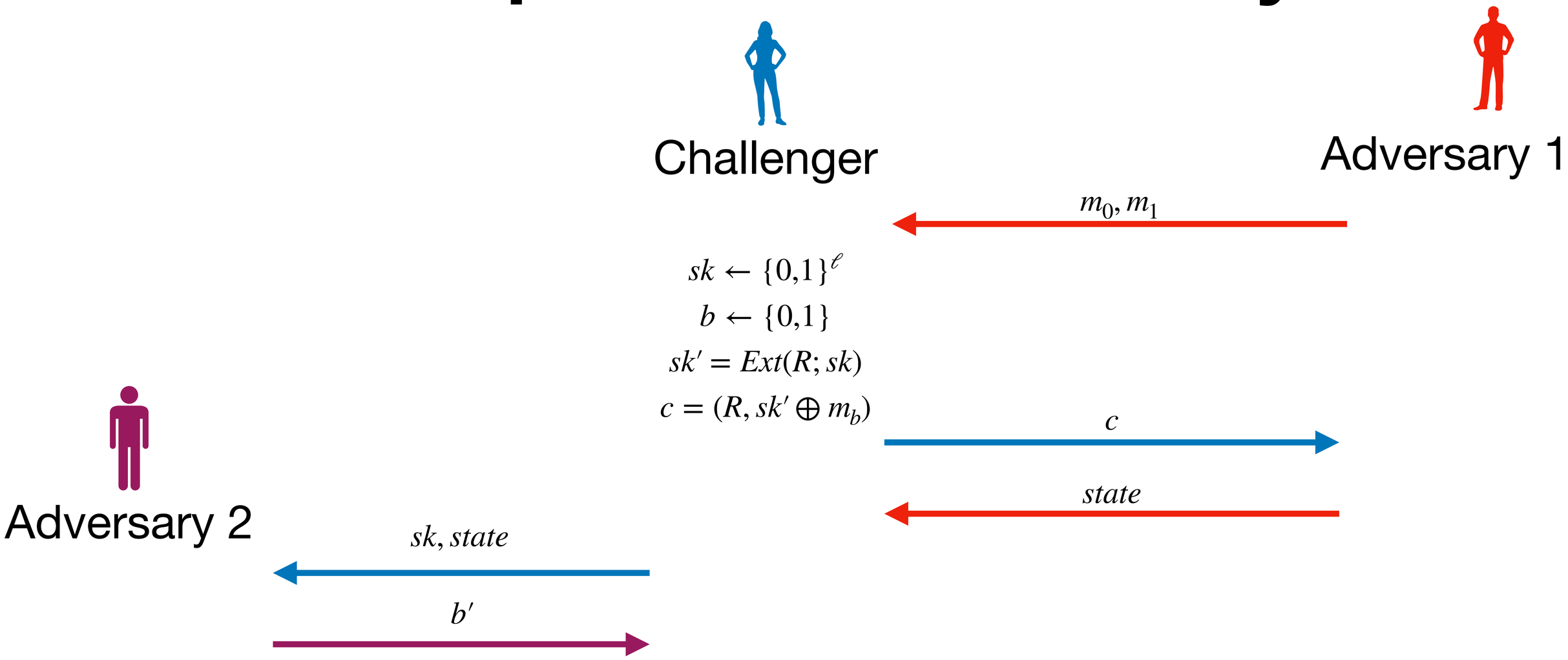
Incompressible Security



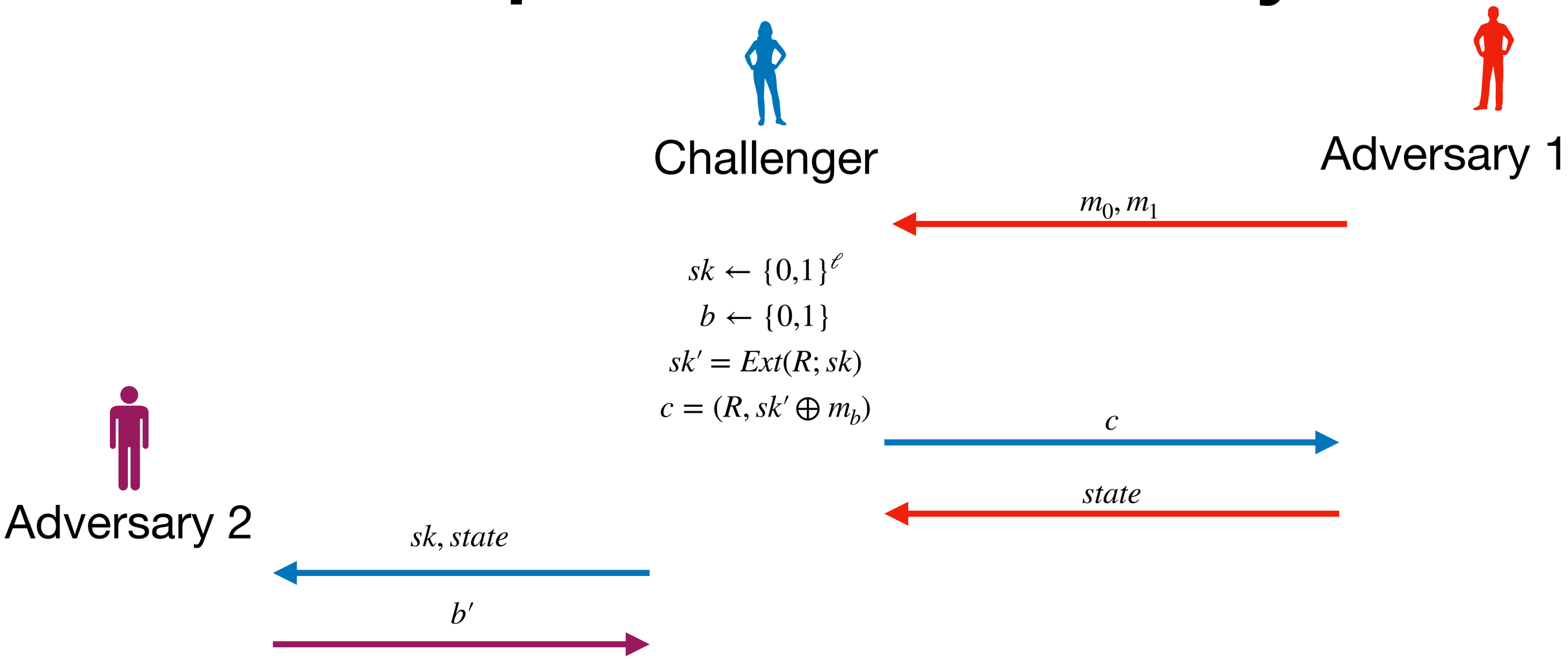
Incompressible Security



Incompressible Security



Incompressible Security



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'

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

$state$

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

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 $state$

$Ext(R; sk)$ 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

$state$

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

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

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

Adversary 2

$sk, state$

b'

Incompressible Security

Challenger

Adversary 1

m_0, m_1

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

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

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

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

c

$state$

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

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

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

Adversary 2

$sk, state$

b'

Our Incompressible PKE Scheme

OUR INCOMPRESSIBLE PKE SCHEME

OUR INCOMPRESSIBLE PKE SCHEME

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

OUR INCOMPRESSIBLE PKE SCHEME

- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template

OUR INCOMPRESSIBLE PKE SCHEME

- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template
- Key Idea : Deferred encryption.

OUR INCOMPRESSIBLE PKE SCHEME

- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template
- Key Idea : Deferred encryption.
    - During encryption, garble incompressible SKE encryption circuit with message hardwired. This outputs a garbled circuit, together with encrypted labels.

OUR INCOMPRESSIBLE PKE SCHEME

- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template
- Key Idea : Deferred encryption.
    - During encryption, garble incompressible SKE encryption circuit with message hardwired. This outputs a garbled circuit, together with encrypted labels.
    - During decryption, first recover labels.

OUR INCOMPRESSIBLE PKE SCHEME

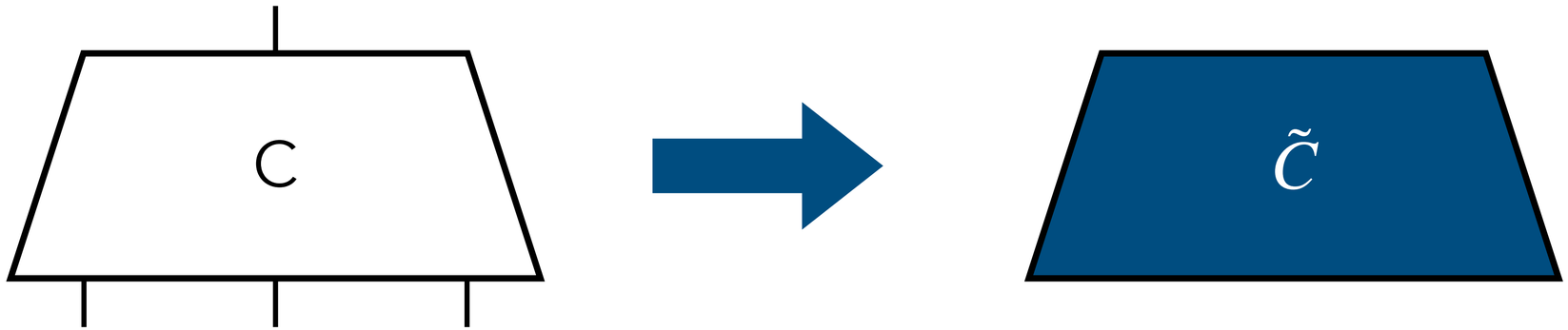
- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template
- Key Idea : Deferred encryption.
    - During encryption, garble incompressible SKE encryption circuit with message hardwired. This outputs a garbled circuit, together with encrypted labels.
    - During decryption, first recover labels.
    - Then evaluate garbled circuit. This produces an incomp. SKE ciphertext.

OUR INCOMPRESSIBLE PKE SCHEME

- Primitives required - PKE, incompressible SKE and garbling scheme.
- Incompressible IBE/ABE follow similar template
- Key Idea : Deferred encryption.
    - During encryption, garble incompressible SKE encryption circuit with message hardwired. This outputs a garbled circuit, together with encrypted labels.
    - During decryption, first recover labels.
    - Then evaluate garbled circuit. This produces an incomp. SKE ciphertext.
    - Decrypt the incompressible SKE ciphertext.

OUR INCOMPRESSIBLE PKE SCHEME

Circuit Garbling

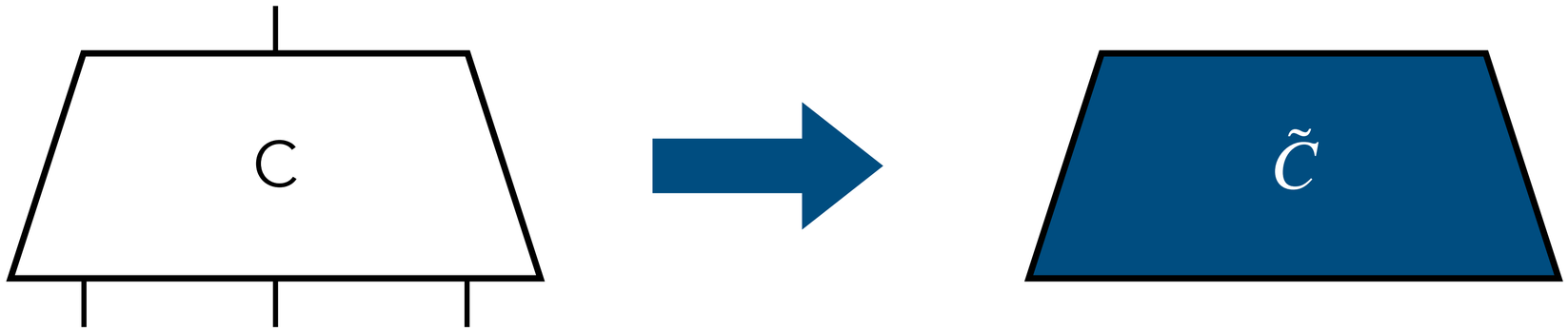


$lab_{1,0}$ $lab_{2,0}$ $lab_{3,0}$

$lab_{1,1}$ $lab_{2,1}$ $lab_{3,1}$

OUR INCOMPRESSIBLE PKE SCHEME

Circuit Garbling



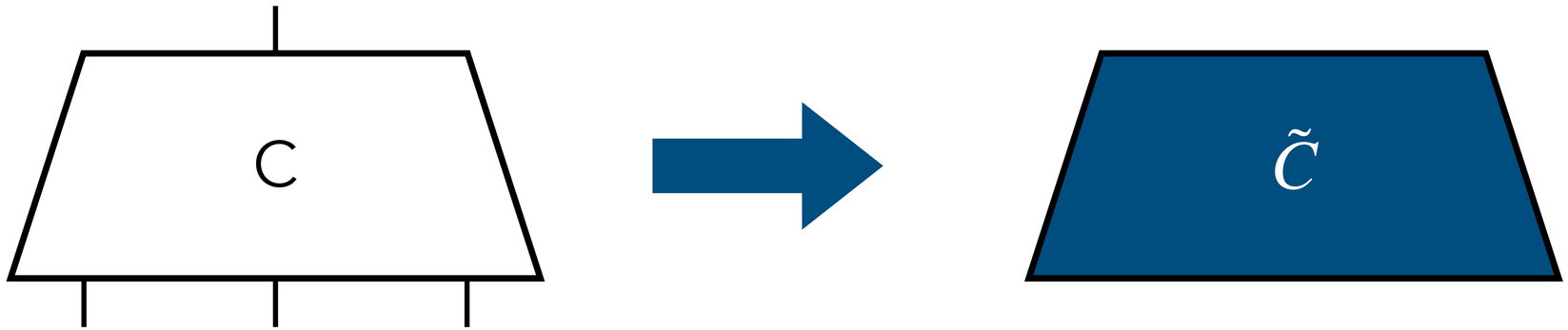
$lab_{1,0}$ $lab_{2,0}$ $lab_{3,0}$

$lab_{1,1}$ $lab_{2,1}$ $lab_{3,1}$

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

OUR INCOMPRESSIBLE PKE SCHEME

Circuit Garbling



$lab_{1,0}$ $lab_{2,0}$ $lab_{3,0}$

$lab_{1,1}$ $lab_{2,1}$ $lab_{3,1}$

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

Security - $\tilde{C}$ and $\{lab_{i,x_i}\}$ reveal $C(x)$, but nothing else.

OUR INCOMPRESSIBLE PKE SCHEME

OUR INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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 INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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 INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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

OUR INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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

OUR INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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

OUR INCOMPRESSIBLE PKE SCHEME

- $Setup()$:
  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

Thank You!