RUHR-UNIVERSITÄT BOCHUM

On the Selective-Opening Security of DHIES

and other practical encryption schemes

UbiCrypt Research Retreat, Schloss Raesfeld: 29.& 30. Sep. 2014

Felix Heuer, Tibor Jager, Eike Kiltz, Sven Schäge
Horst Görtz Institute for IT Security
Ruhr University Bochum

Horst Görtz Institut für IT-Sicherheit

hgi
Horst Görtz Institut
für IT-Sicherheit

Selective-Opening Attacks

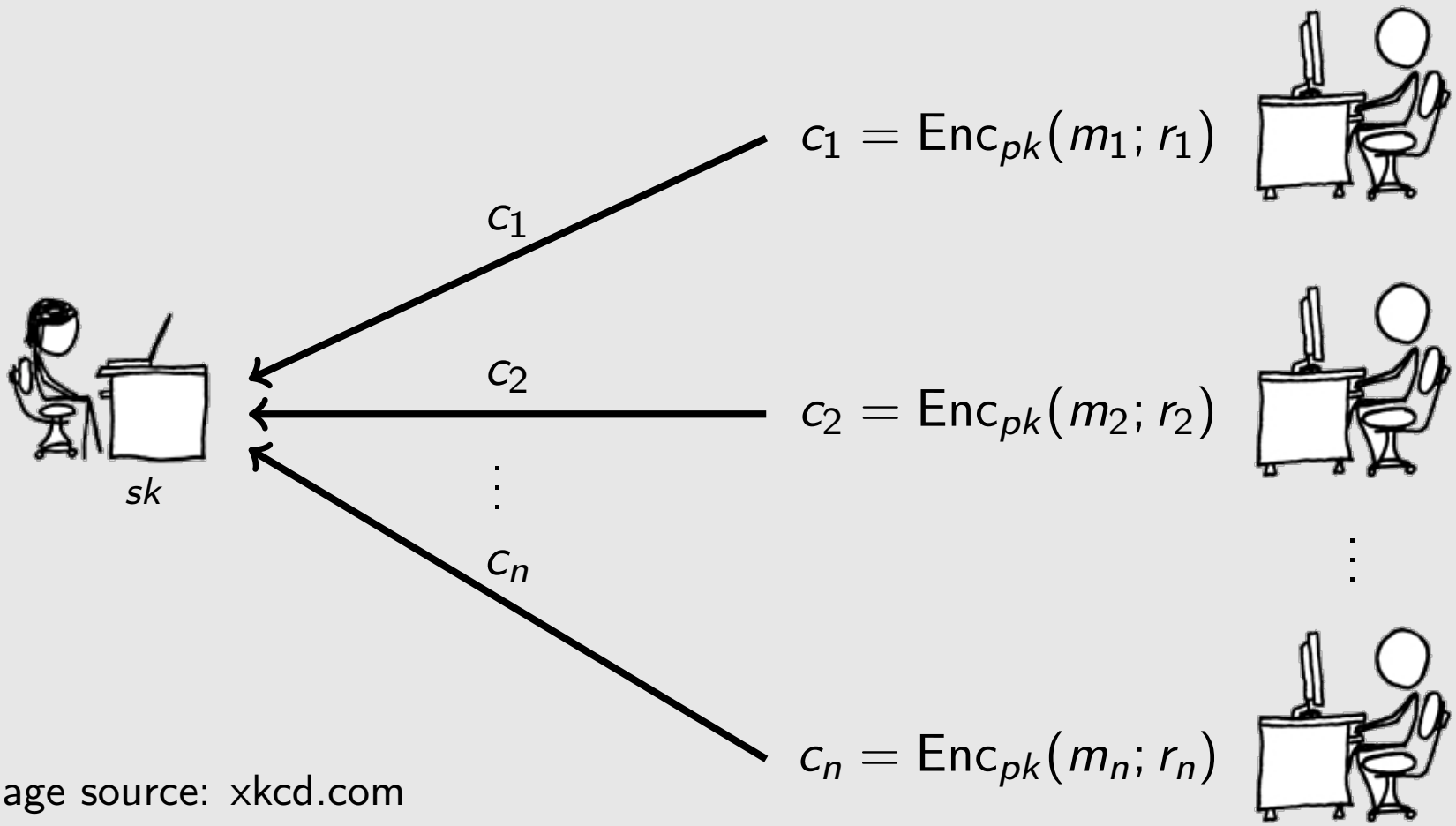


Image source: xkcd.com

Selective-Opening Attacks

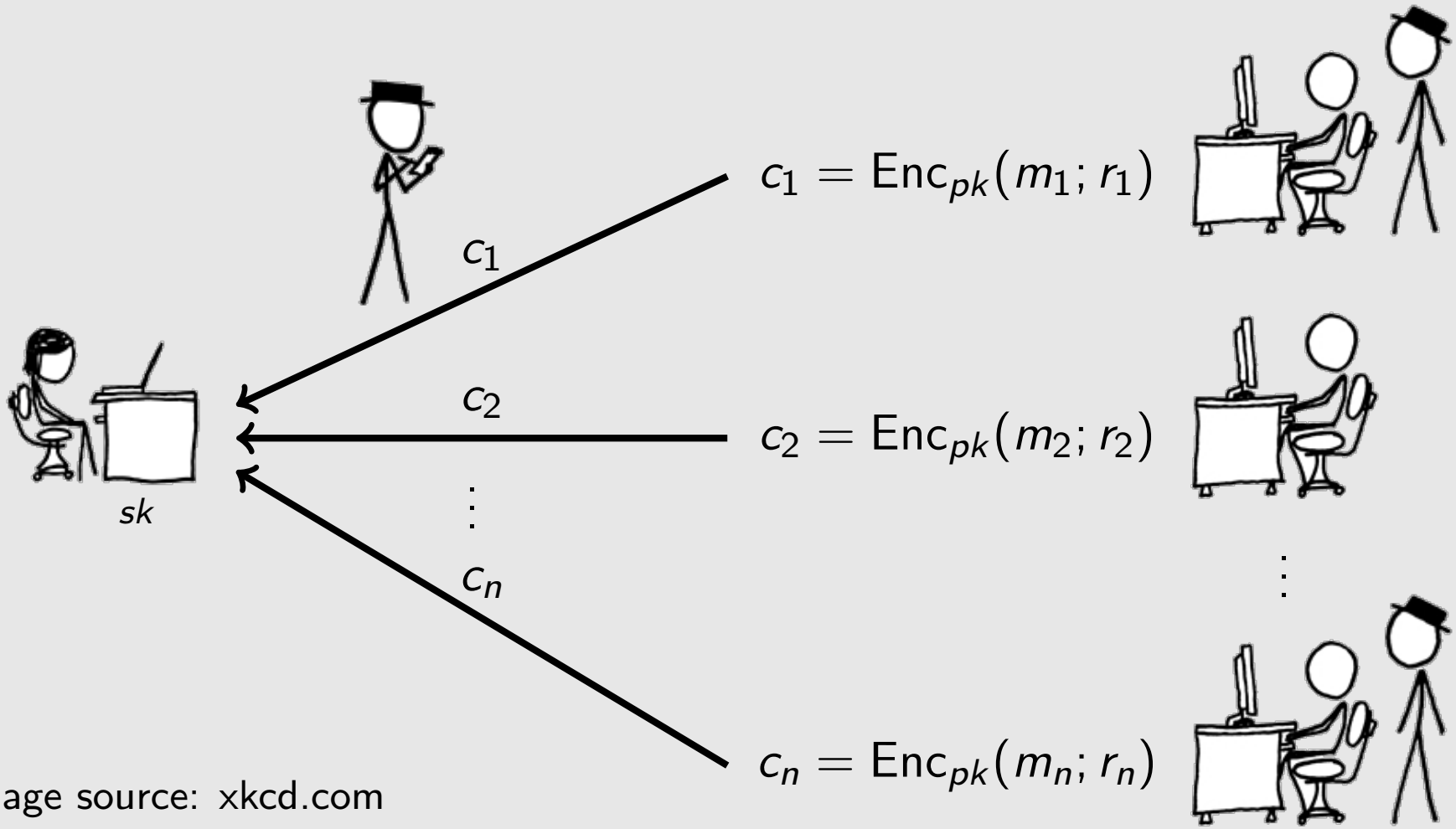
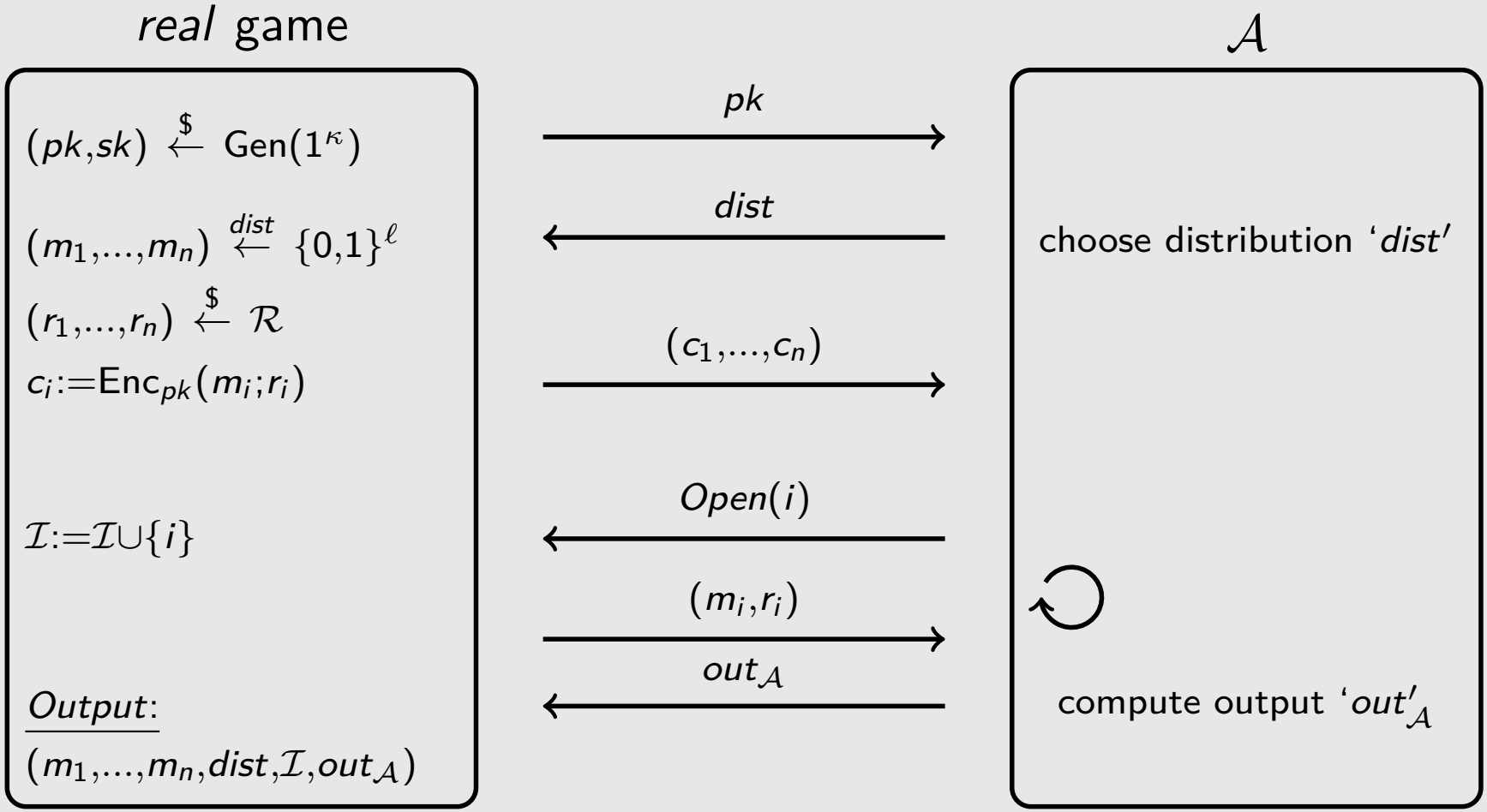
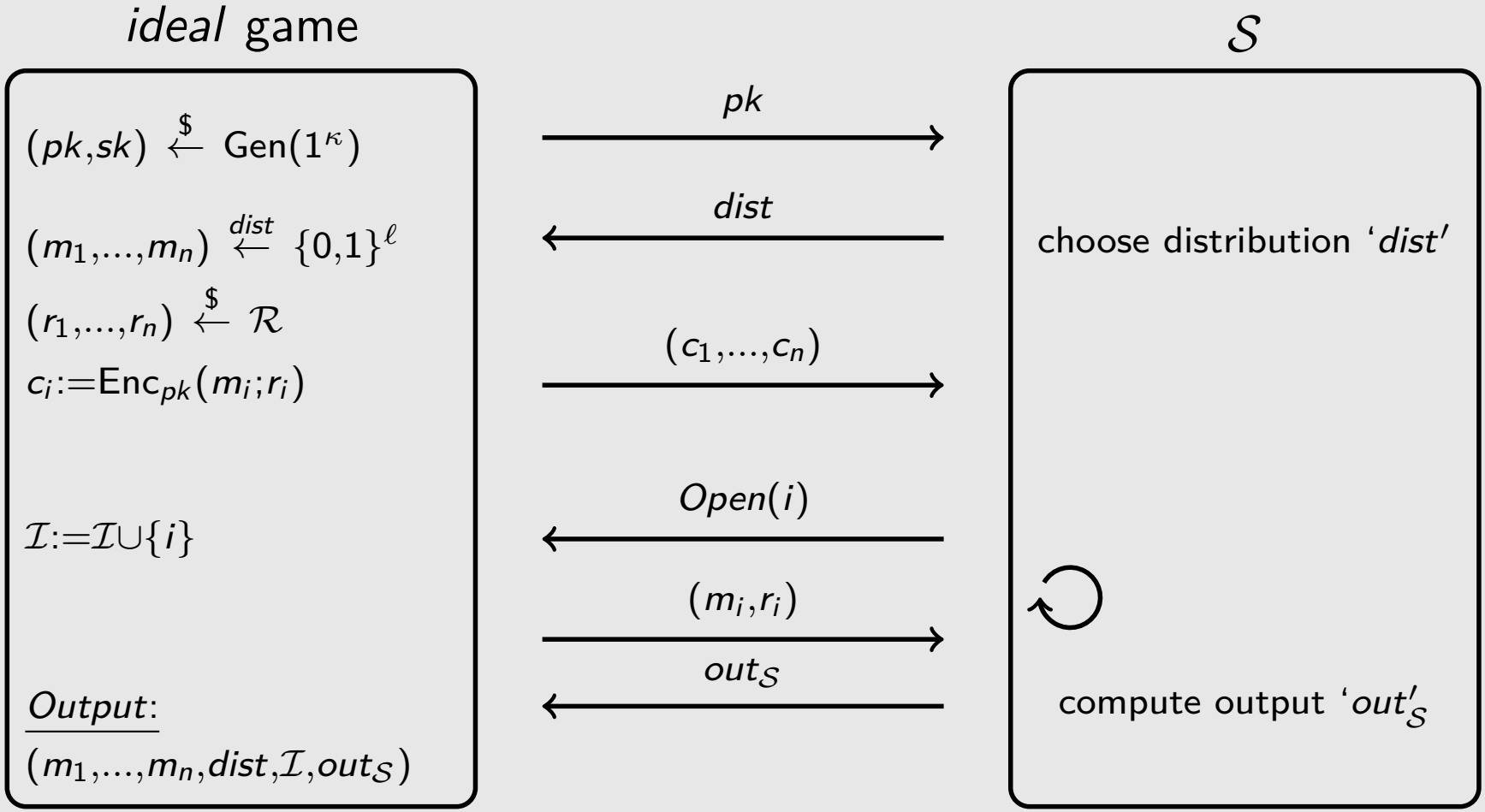


Image source: xkcd.com

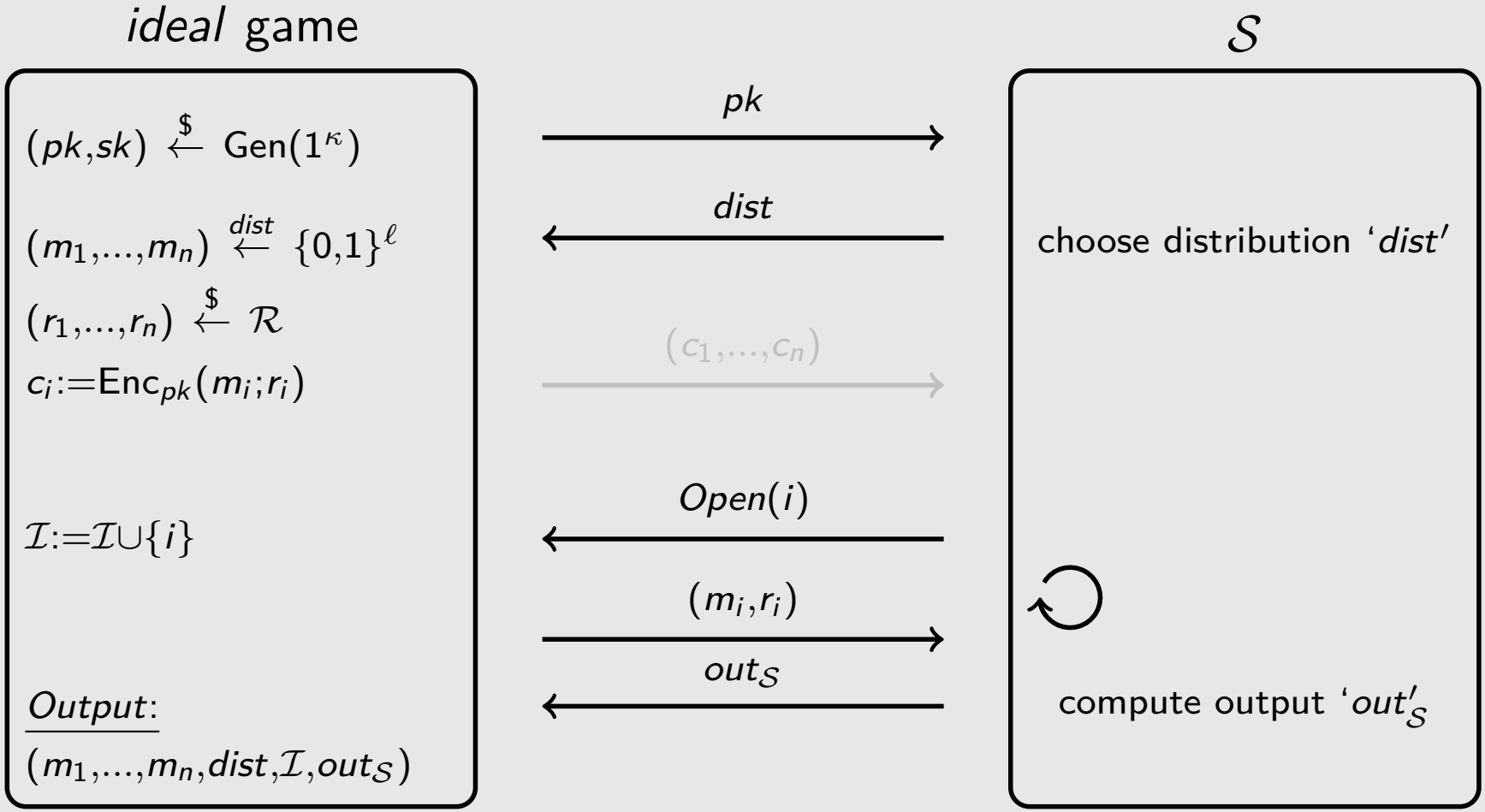
SIM-SO-CPA security definition



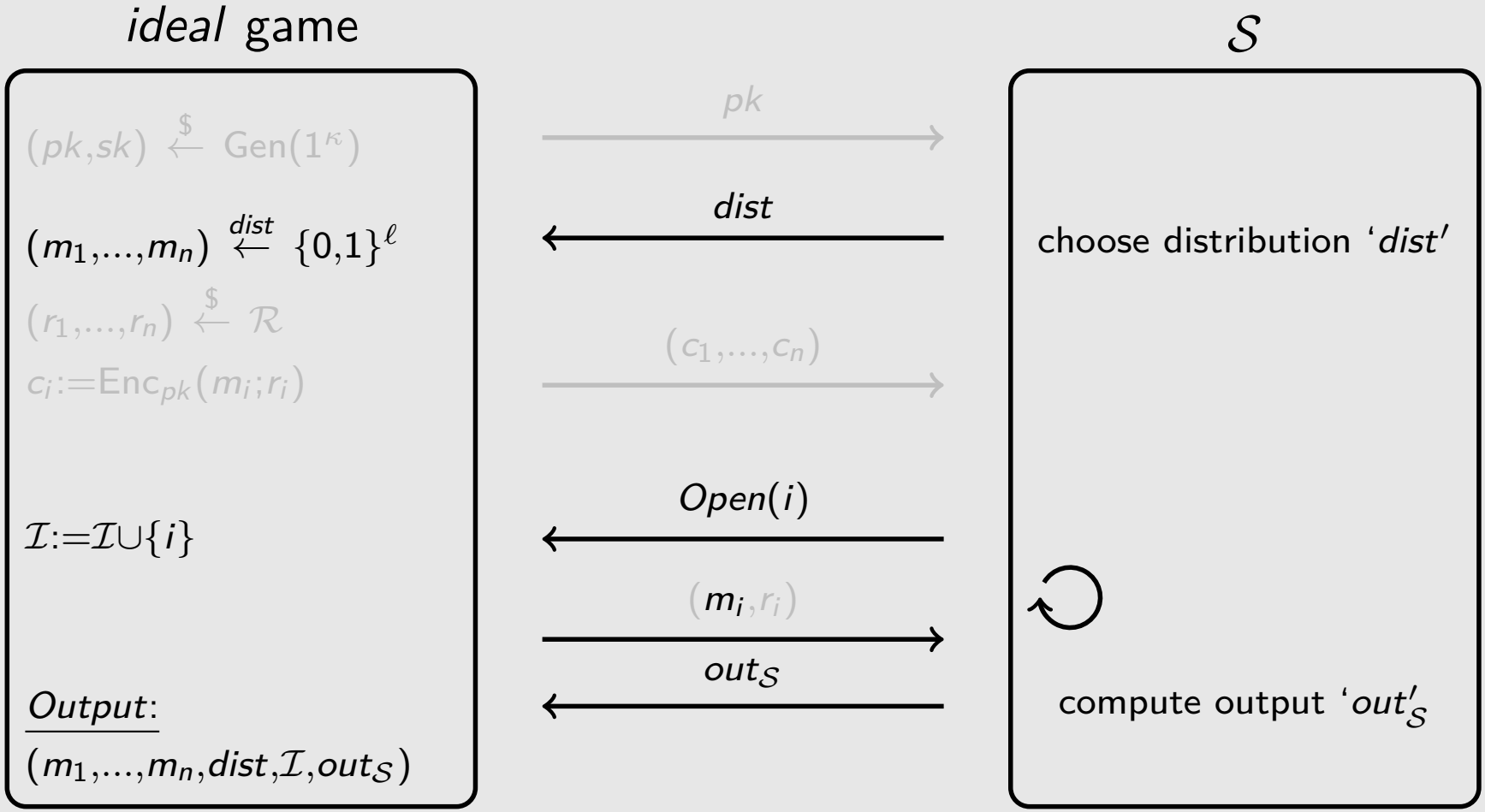
SIM-SO-CPA security definition



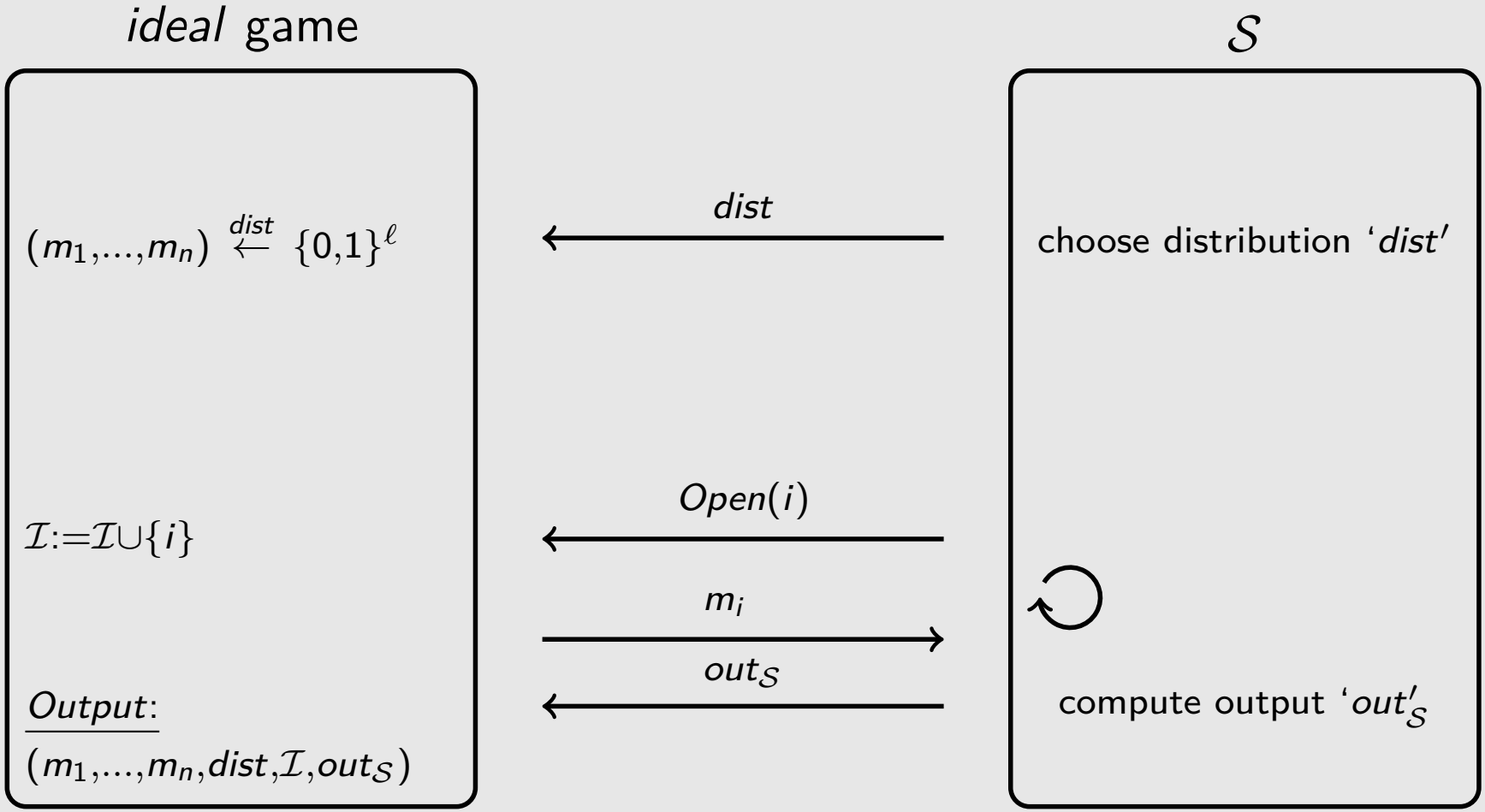
SIM-SO-CPA security definition



SIM-SO-CPA security definition



SIM-SO-CPA security definition



SIM-SO-CPA security definition

Definition 1 (SIM-SO-CPA security)

Let PKE be a public key encryption scheme. PKE is SIM-SO-CPA secure if for every PPT adversary $\mathcal{A}$ there exists a PPT simulator $\mathcal{S} := \mathcal{S}(\mathcal{A})$ such that the distributions induced by

$\mathcal{A}$ run in the *real* game and $\mathcal{S}$ run in the *ideal* game

are computationally indistinguishable.

DHIES

Let g be a generator of a group of size p, $\{0,1\}^\ell$ a message space and $H : \langle g \rangle \mapsto \{0,1\}^\ell$ a hash function.

DHIES

Let g be a generator of a group of size p, $\{0,1\}^\ell$ a message space and $H : \langle g \rangle \mapsto \{0,1\}^\ell$ a hash function.

$\underline{\text{Gen}}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$X := g^x$
$pk := (g, p, X, H)$
$sk := x$
Return pk

DHIES

Let g be a generator of a group of size p, $\{0,1\}^\ell$ a message space and $H : \langle g \rangle \mapsto \{0,1\}^\ell$ a hash function.

$\underline{\text{Gen}}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$X := g^x$
$pk := (g, p, X, H)$
$sk := x$
Return pk

$\underline{\text{Enc}_{pk}(m)}$
$r \xleftarrow{\$} \mathbb{Z}_p$
$c_1 := g^r$
$c_2 := H(X^r) \oplus m$
Return (c_1, c_2)

DHIES

Let g be a generator of a group of size p, $\{0,1\}^\ell$ a message space and $H : \langle g \rangle \mapsto \{0,1\}^\ell$ a hash function.

$\underline{\text{Gen}}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$X := g^x$
$pk := (g, p, X, H)$
$sk := x$
Return pk

$\underline{\text{Enc}_{pk}(m)}$
$r \xleftarrow{\$} \mathbb{Z}_p$
$c_1 := g^r$
$c_2 := H(X^r) \oplus m$
Return (c_1, c_2)

$\underline{\text{Dec}_{sk}(c_1, c_2)}$
Return $H({c_1}^x) \oplus c_2$

DHIES

Let g be a generator of a group of size p, $\{0,1\}^\ell$ a message space and $H : \langle g \rangle \mapsto \{0,1\}^\ell$ a hash function.

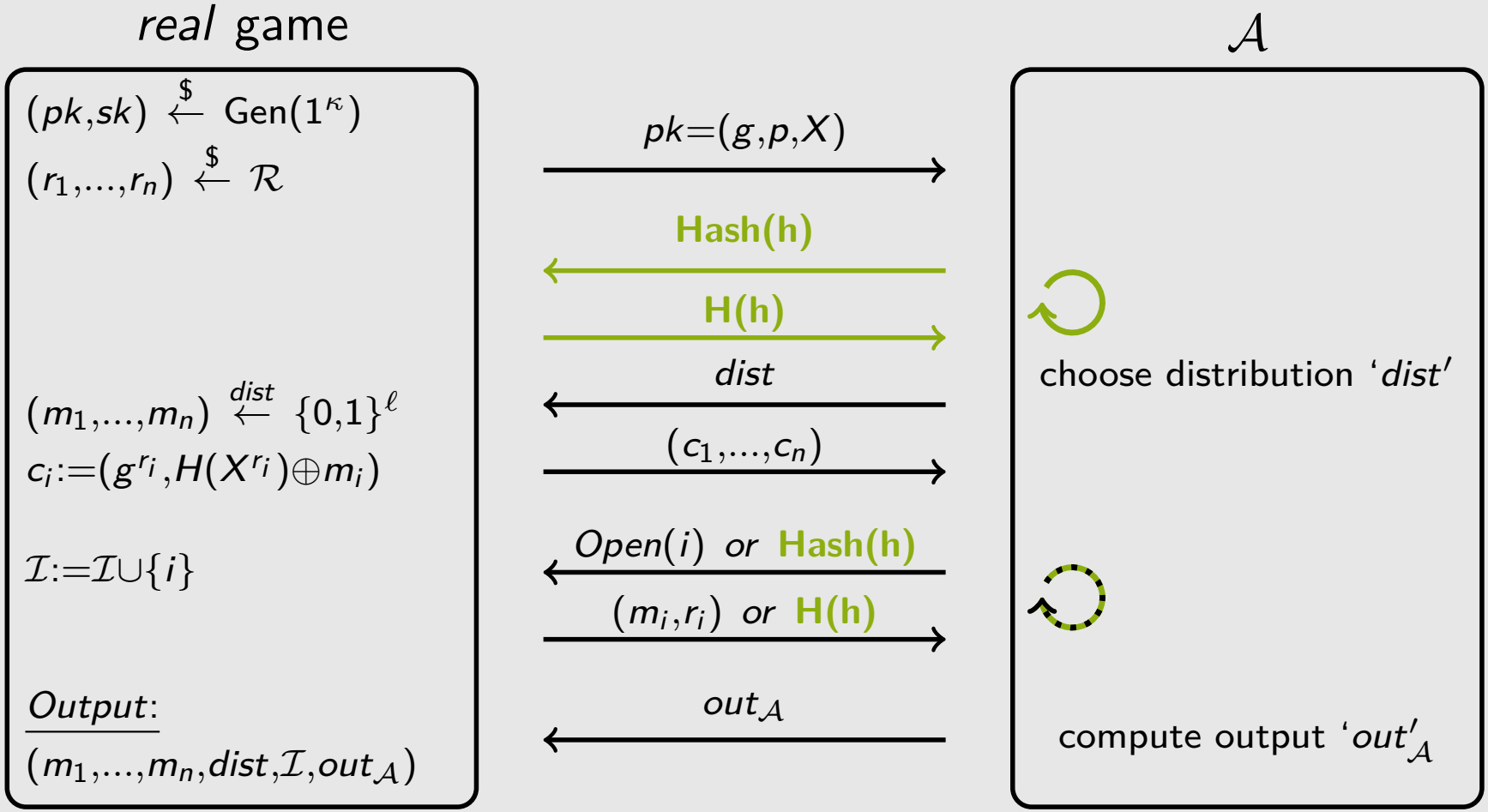
$\underline{\text{Gen}}$
$x \xleftarrow{\$} \mathbb{Z}_p$
$X := g^x$
$pk := (g, p, X, H)$
$sk := x$
Return pk

$\underline{\text{Enc}_{pk}(m)}$
$r \xleftarrow{\$} \mathbb{Z}_p$
$c_1 := g^r$
$c_2 := H(X^r) \oplus m$
Return (c_1, c_2)

$\underline{\text{Dec}_{sk}(c_1, c_2)}$
Return $H({c_1}^x) \oplus c_2$

Notice, that we have to provide $\mathcal{A}$ oracle access to H.

SIM-SO-CPA security game for DHIES



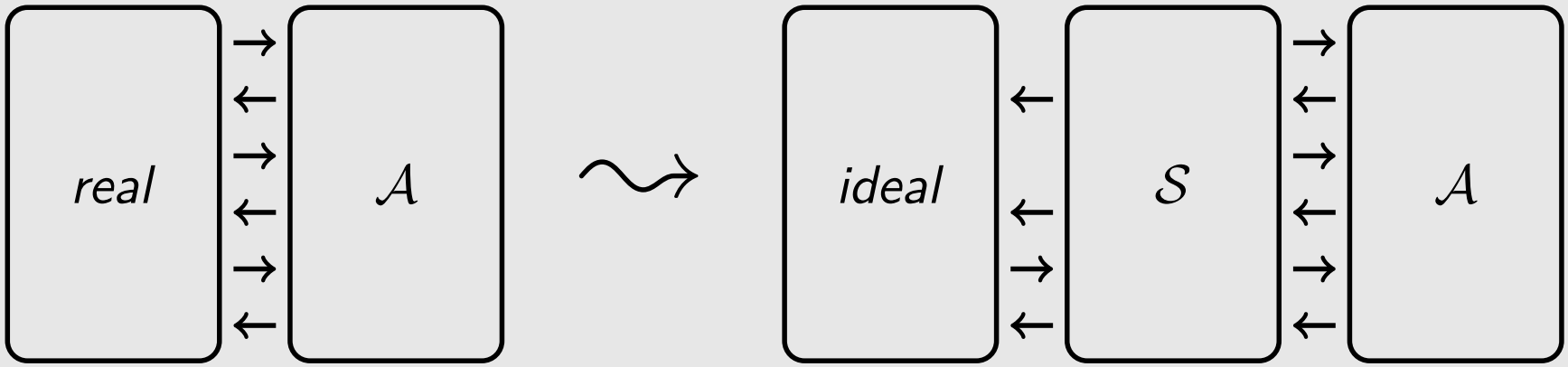
Notice, that we sample r_i in advance.

Theorem 2

The DHIES encryption scheme is SIM-SO-CPA secure in the random oracle model, if the CDH assumption holds.

Theorem 2

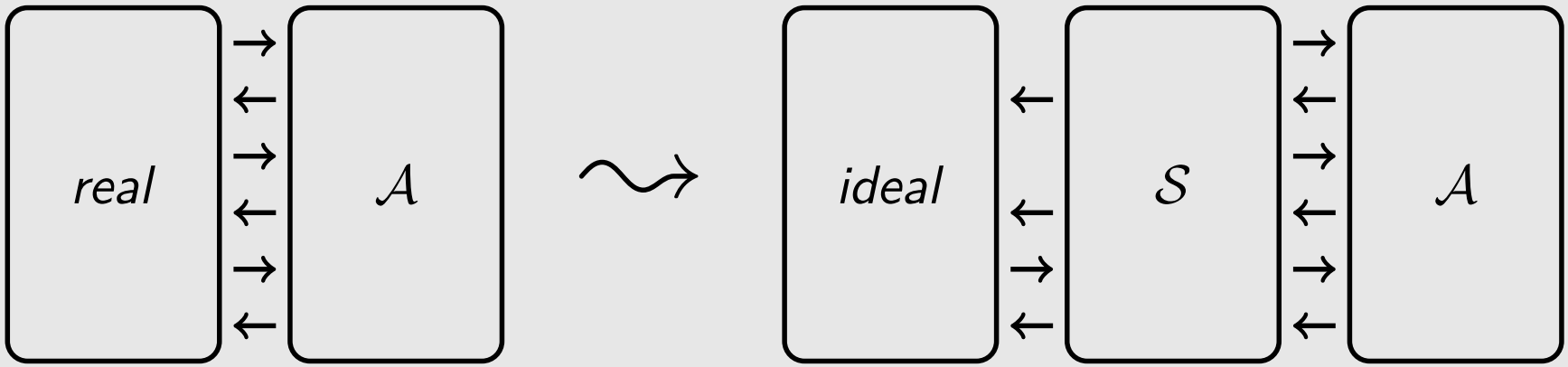
The DHIES encryption scheme is SIM-SO-CPA secure in the random oracle model, if the CDH assumption holds.



Usual idea: Proceed in a sequence of games until a simulator can take over and run $\mathcal{A}$ on its own.

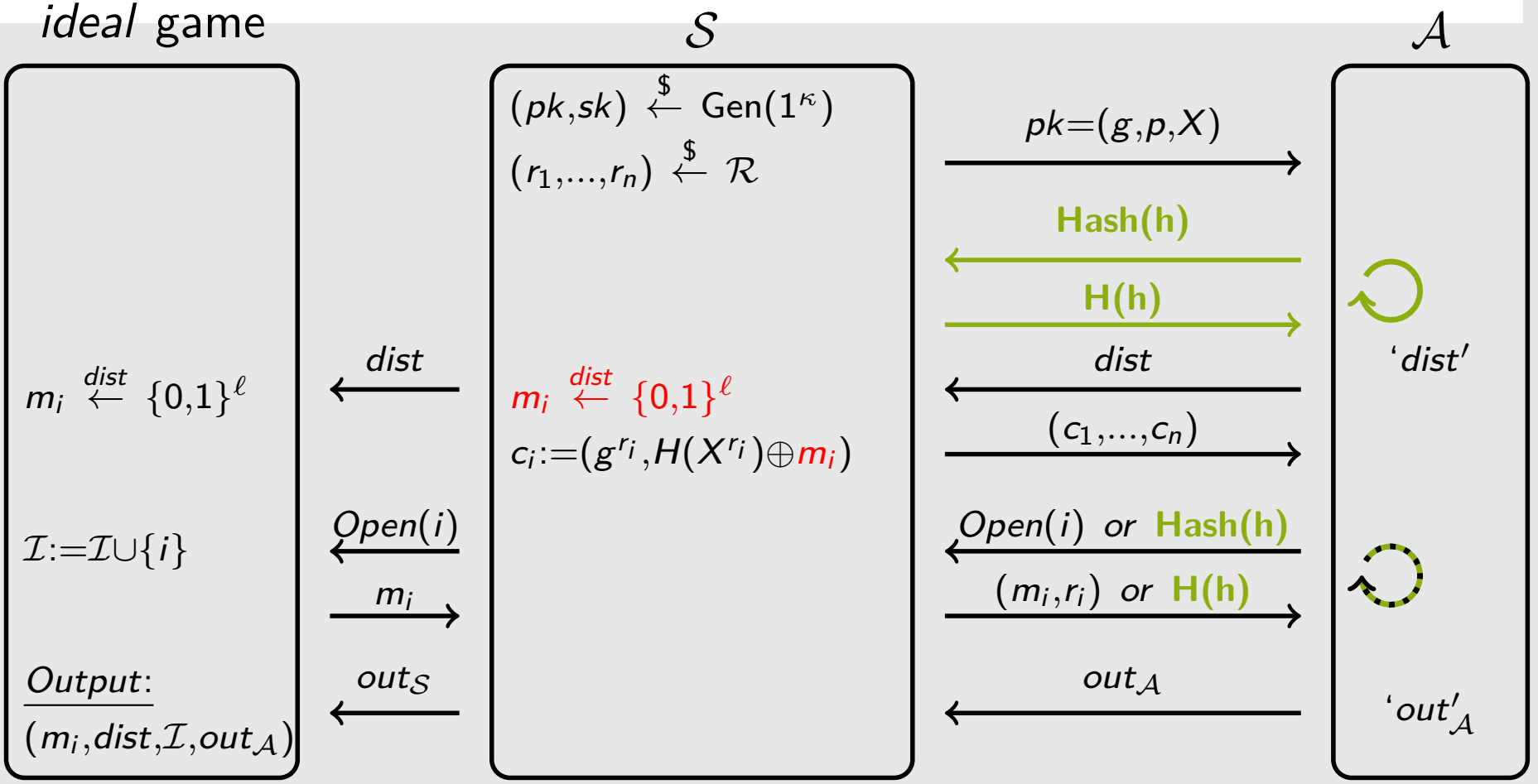
Theorem 2

The DHIES encryption scheme is SIM-SO-CPA secure in the random oracle model, if the CDH assumption holds.

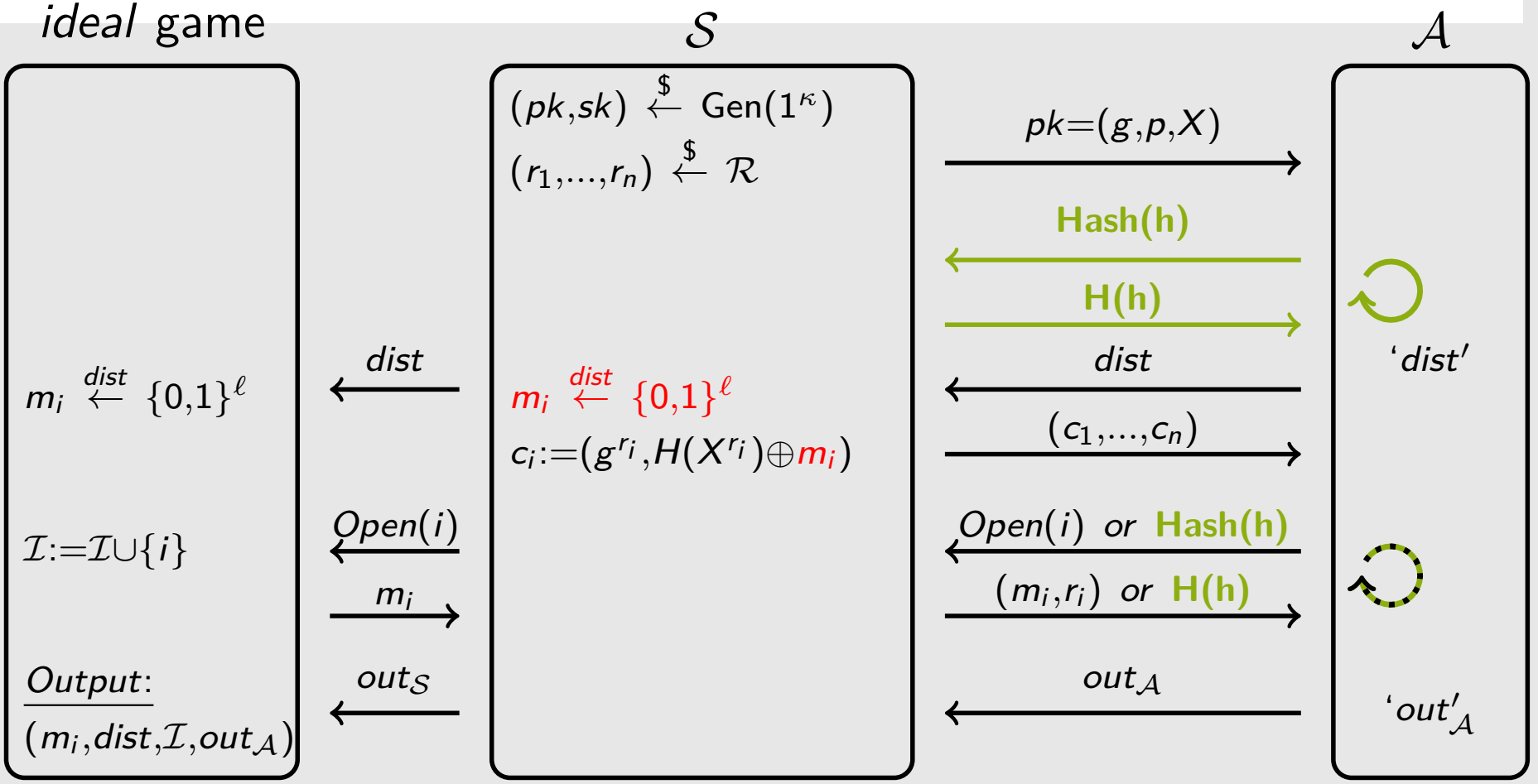


Our approach: We try to construct a simulator right away to see where we run into pitfalls.

Proof.


ideal game
$\mathcal{S}$
$\mathcal{A}$
$(pk,sk) \xleftarrow{\$} \mathrm{Gen}(1^\kappa)$
$(r_1,\ldots,r_n) \xleftarrow{\$} \mathcal{R}$
$pk=(g,p,X)$
Hash(h)
H(h)
'dist'
dist
$m_i \xleftarrow{dist} \{0,1\}^\ell$
$m_i \xleftarrow{dist} \{0,1\}^\ell$
$c_i:=(g^{r_i},H(X^{r_i})\oplus m_i)$
$(c_1,\ldots,c_n)$
$\mathcal{I}:=\mathcal{I}\cup\{i\}$
Open(i)
Open(i) or Hash(h)
m_i
(m_i,r_i) or H(h)
Output:
$(m_i,dist,\mathcal{I},out_\mathcal{A})$
$out_\mathcal{S}$
$out_\mathcal{A}$
'out'$_\mathcal{A}$

Proof.



Step 1) Make $H(X^{r_i}) \oplus m_i$ uniformly random.

Proof.

Step 1) Make $H(X^{r_i}) \oplus m_i$ uniformly random.

Proof.

Step 1) Make $H(X^{r_i}) \oplus m_i$ uniformly random.

Abort condition (earlyAbort)

We abort $\mathcal{A}$ if $\mathcal{A}$ should query some $H(X^{r_i})$ before sending dist.

$$\Pr[earlyAbort] \leq negl.$$

Proof.

Step 1) Make $H(X^{r_i}) \oplus m_i$ uniformly random.

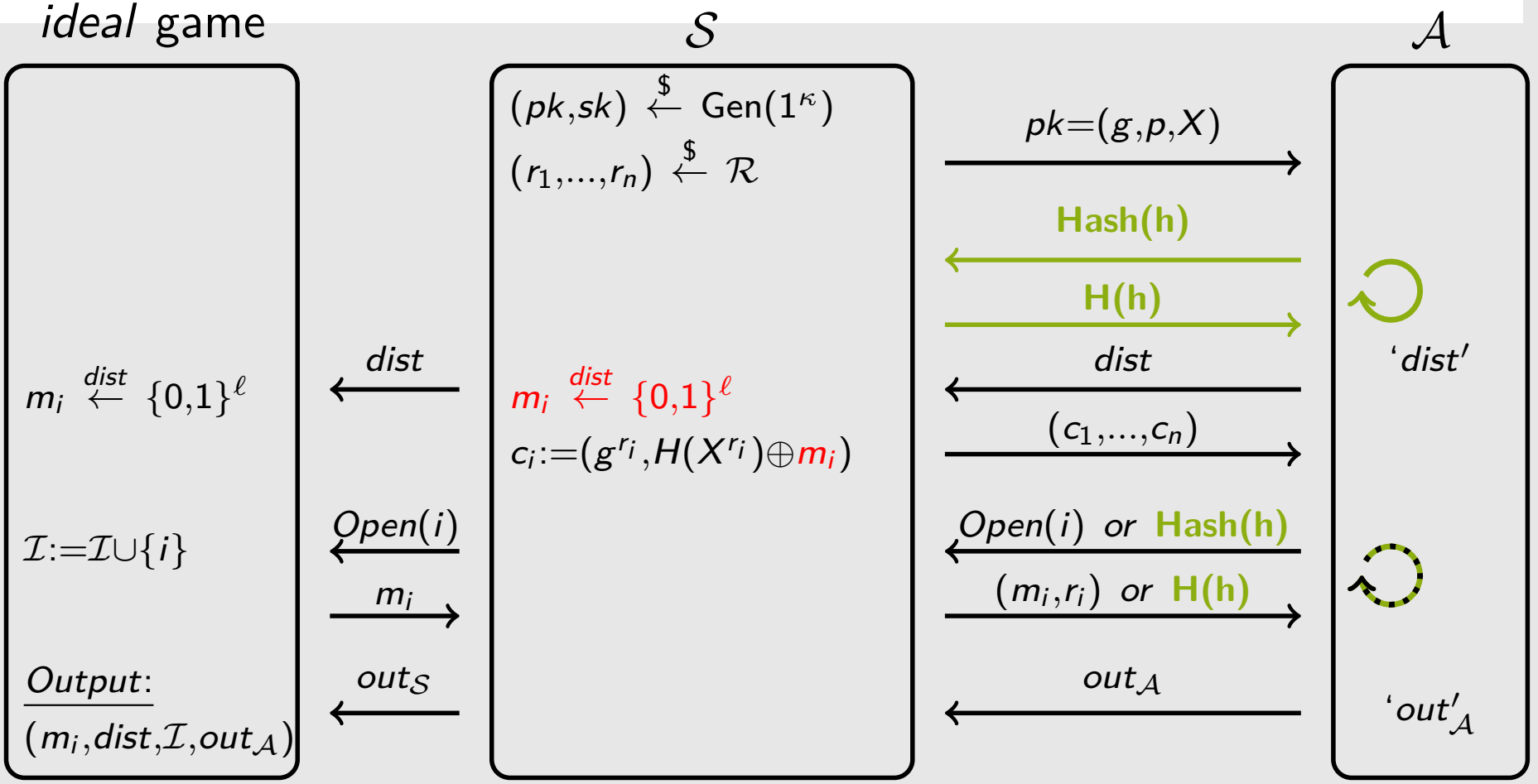
Abort condition (earlyAbort)

We abort $\mathcal{A}$ if $\mathcal{A}$ should query some $H(X^{r_i})$ before sending dist.

$$\Pr[earlyAbort] \leq negl.$$

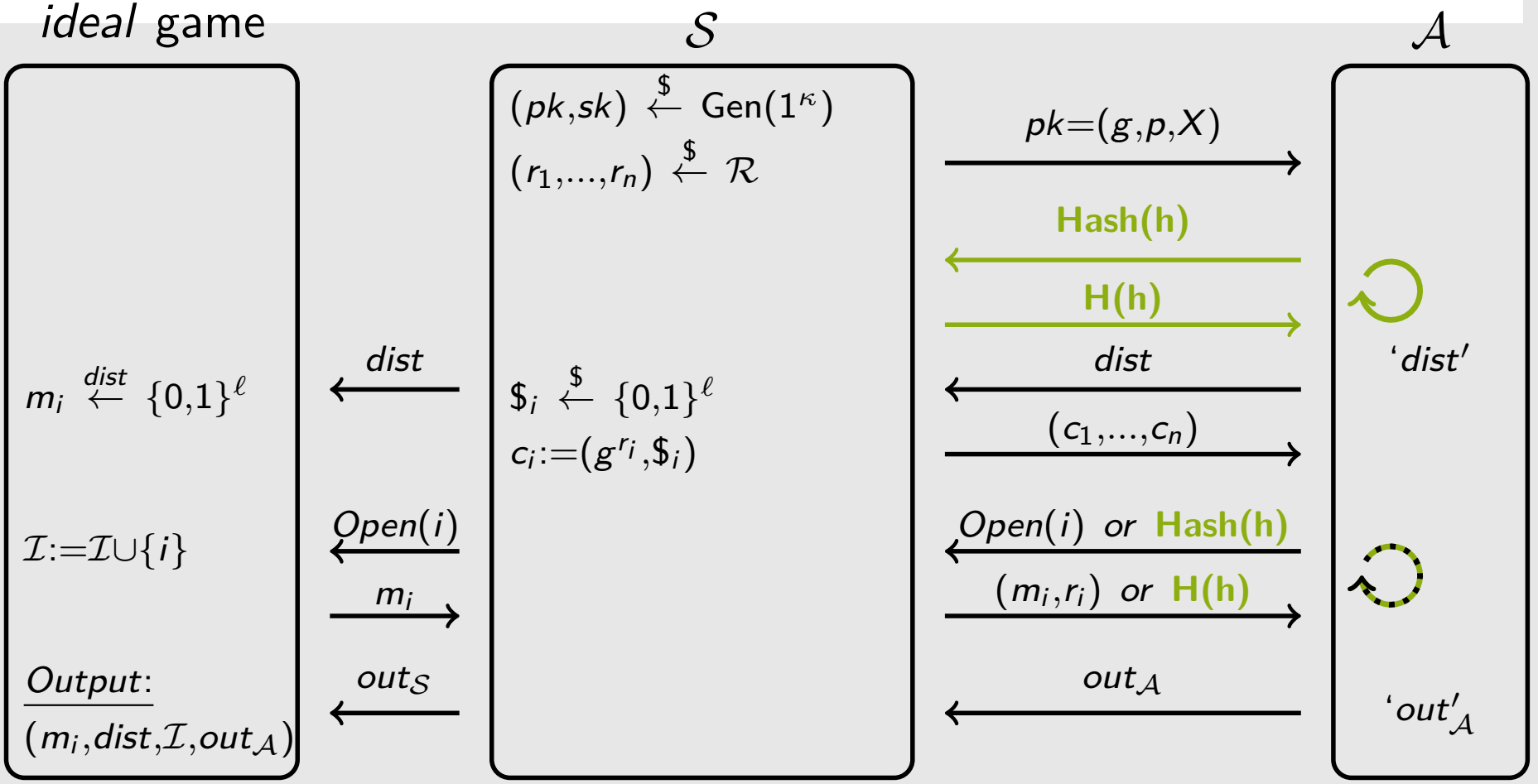
Statistical argument suffices.

Proof.



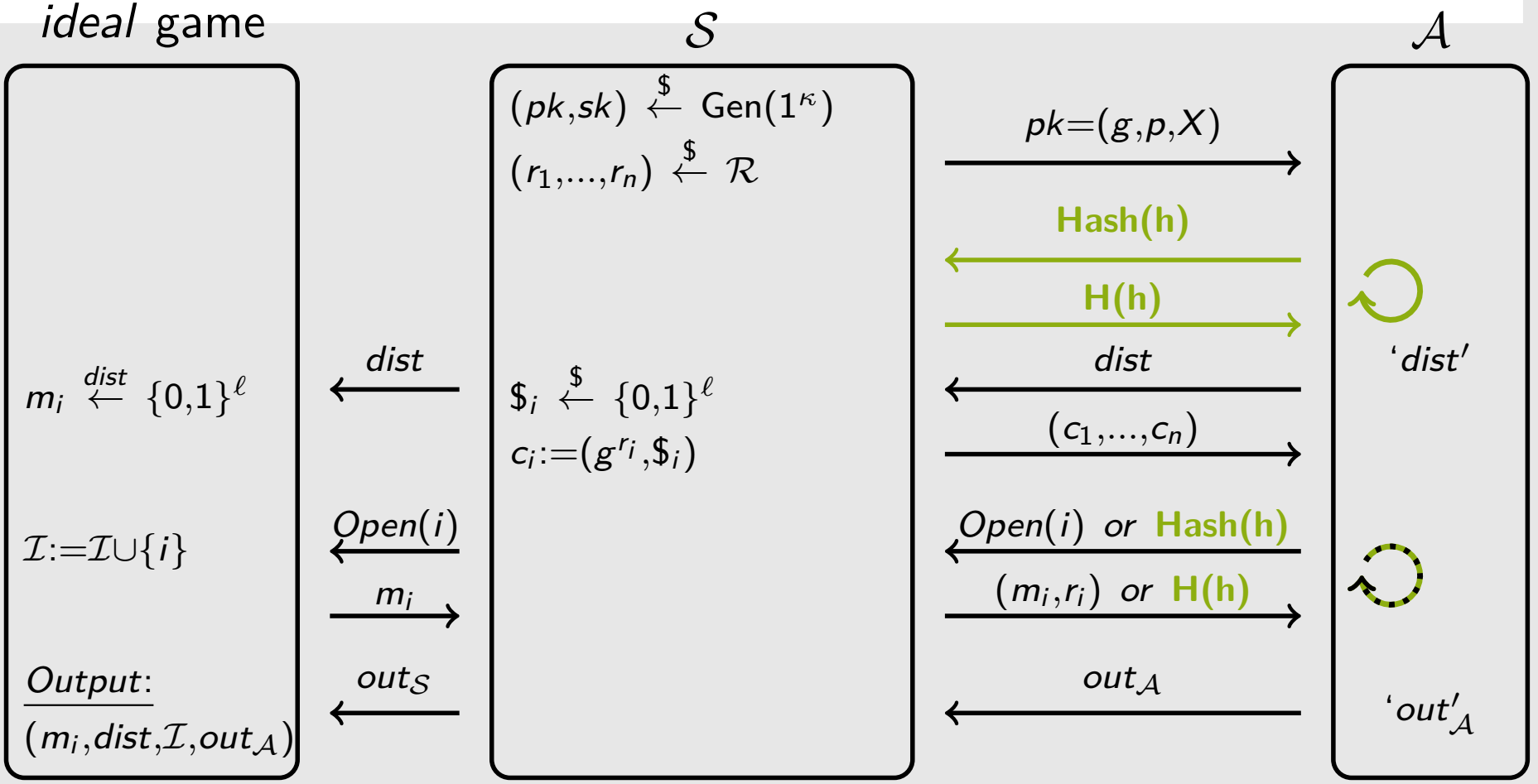
Step 2) Change encryption

Proof.



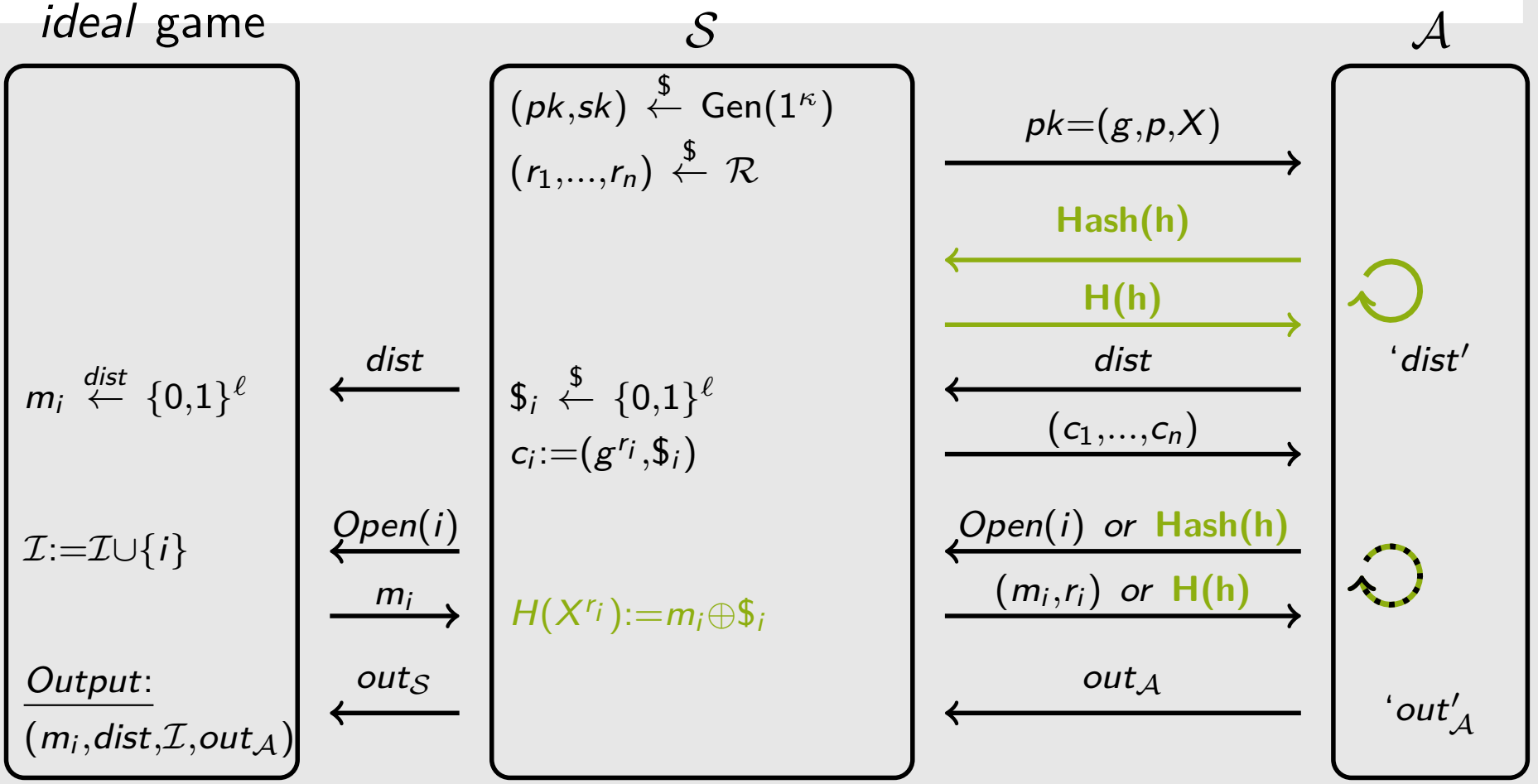
Step 2) Change encryption

Proof.



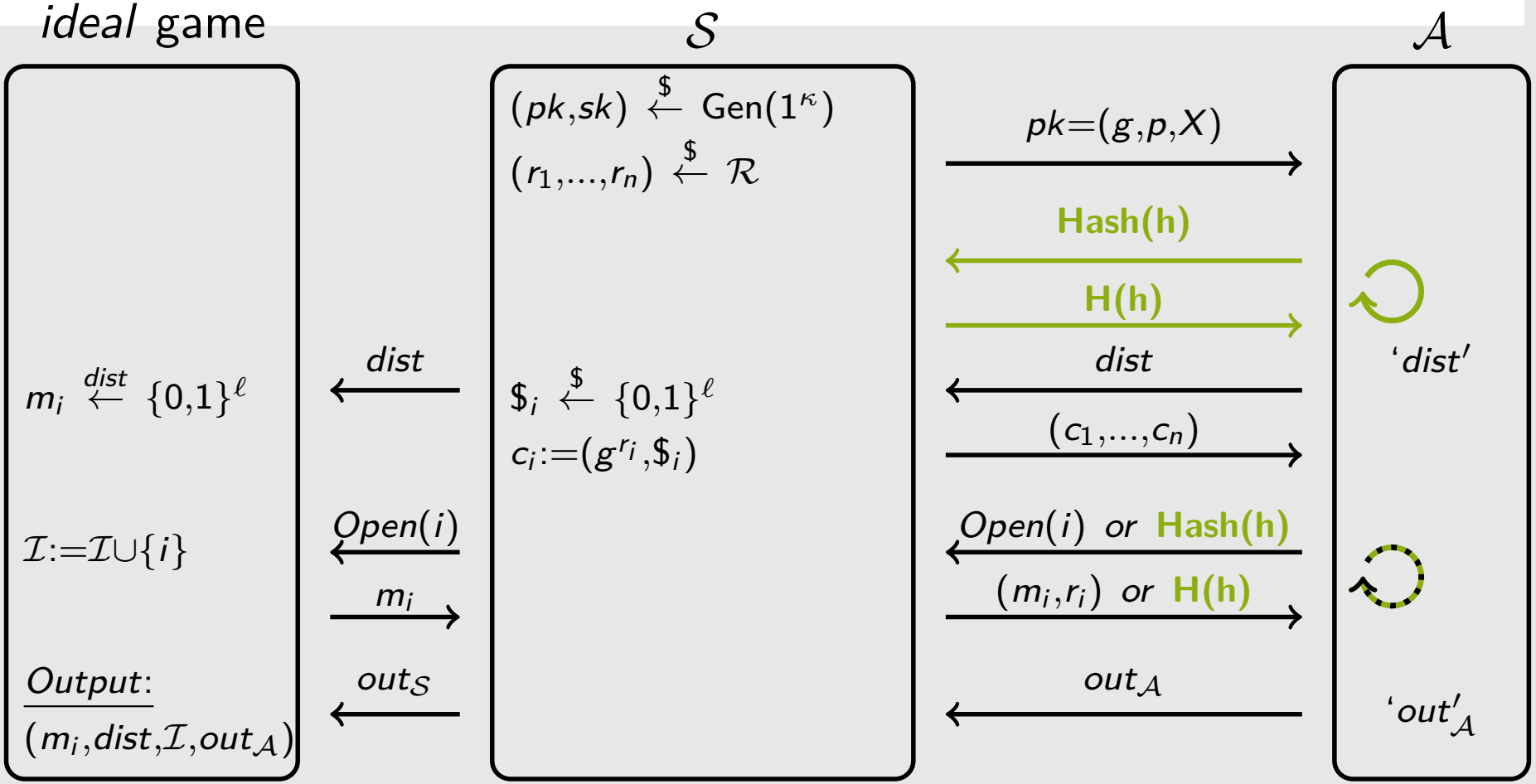
Step 3) How to process **Hash(X^{r_i})** queries?

Proof.



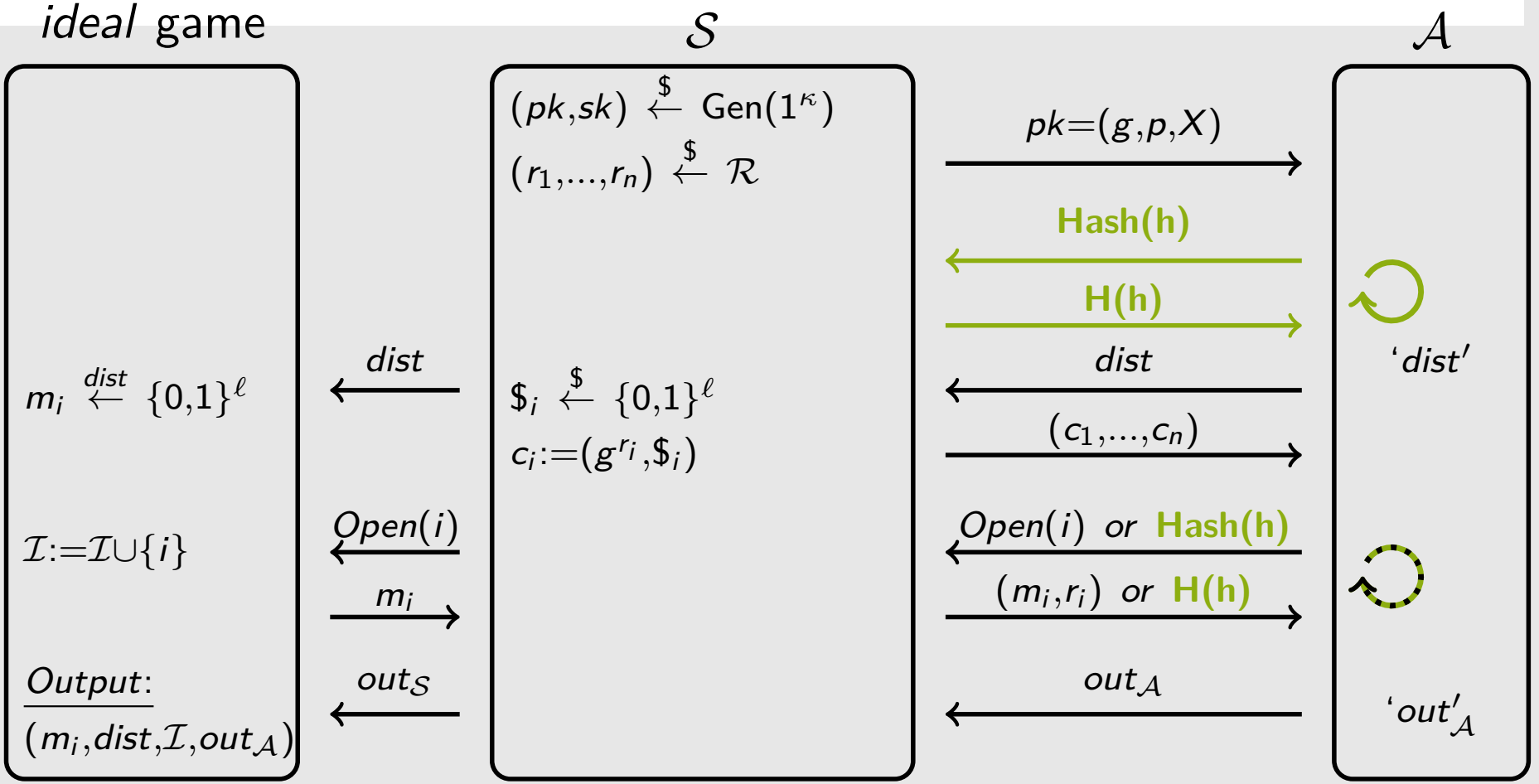
Case 1) $\mathcal{A}$ called $Open(i)$ before querying **Hash(X^{r_i})**.

Proof.



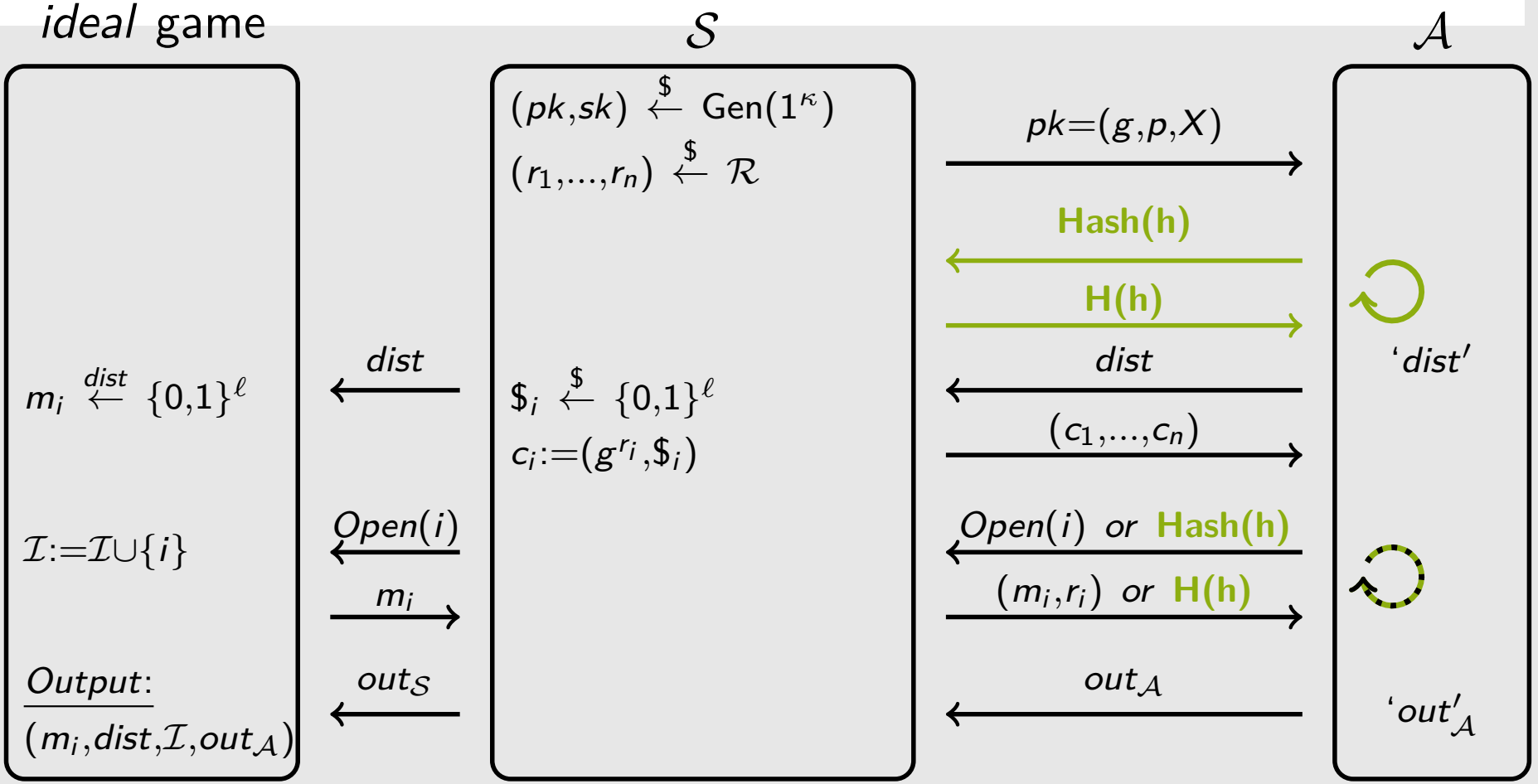
Case 2) $\mathcal{A}$ did not called $Open(i)$ before querying **Hash(X^{r_i})**.

Proof.



Case 2) $\mathcal{A}$ did not called $Open(i)$ before querying **Hash(X^{r_i})**.
$\mathcal{S}$ can neither call $Open(i)$,

Proof.



Case 2) $\mathcal{A}$ did not called $Open(i)$ before querying **Hash(X^{r_i})**.
$\mathcal{S}$ can neither call $Open(i)$, nor answer $\mathcal{A}$'s query.

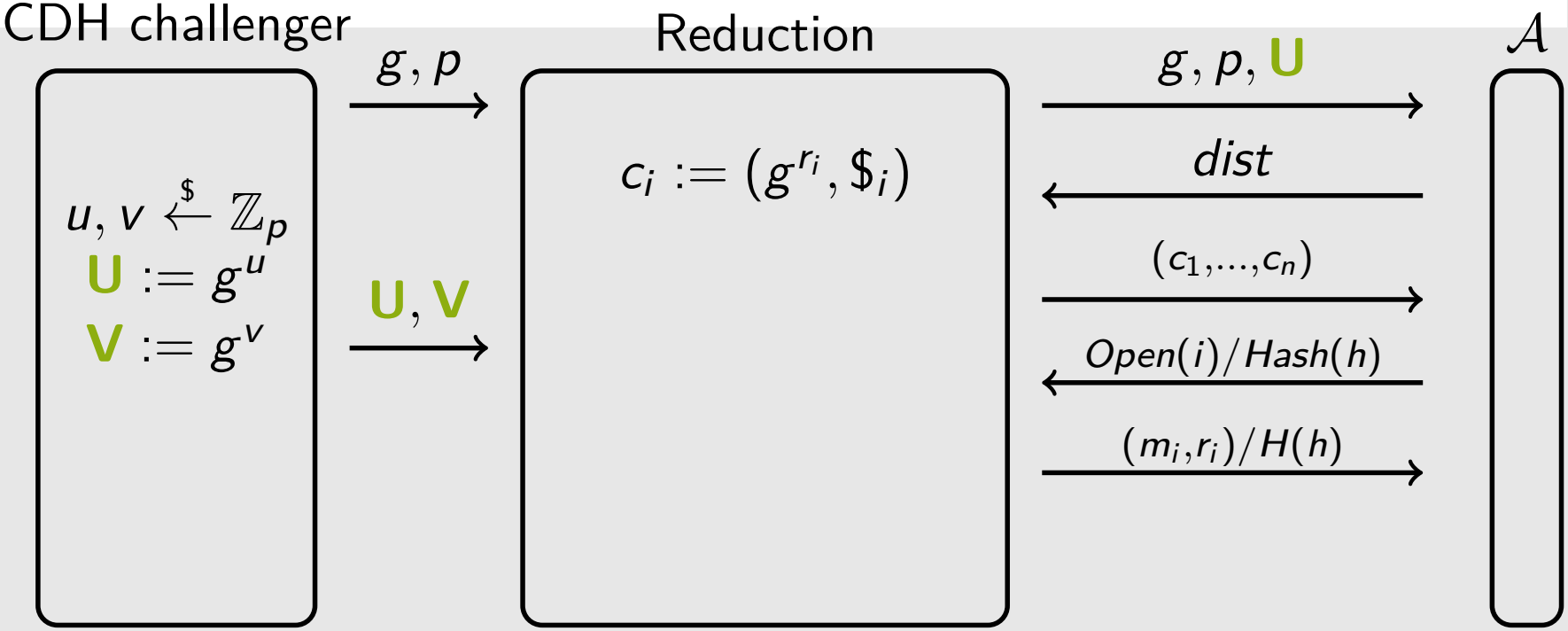
Proof.

Abort condition (AbortH)

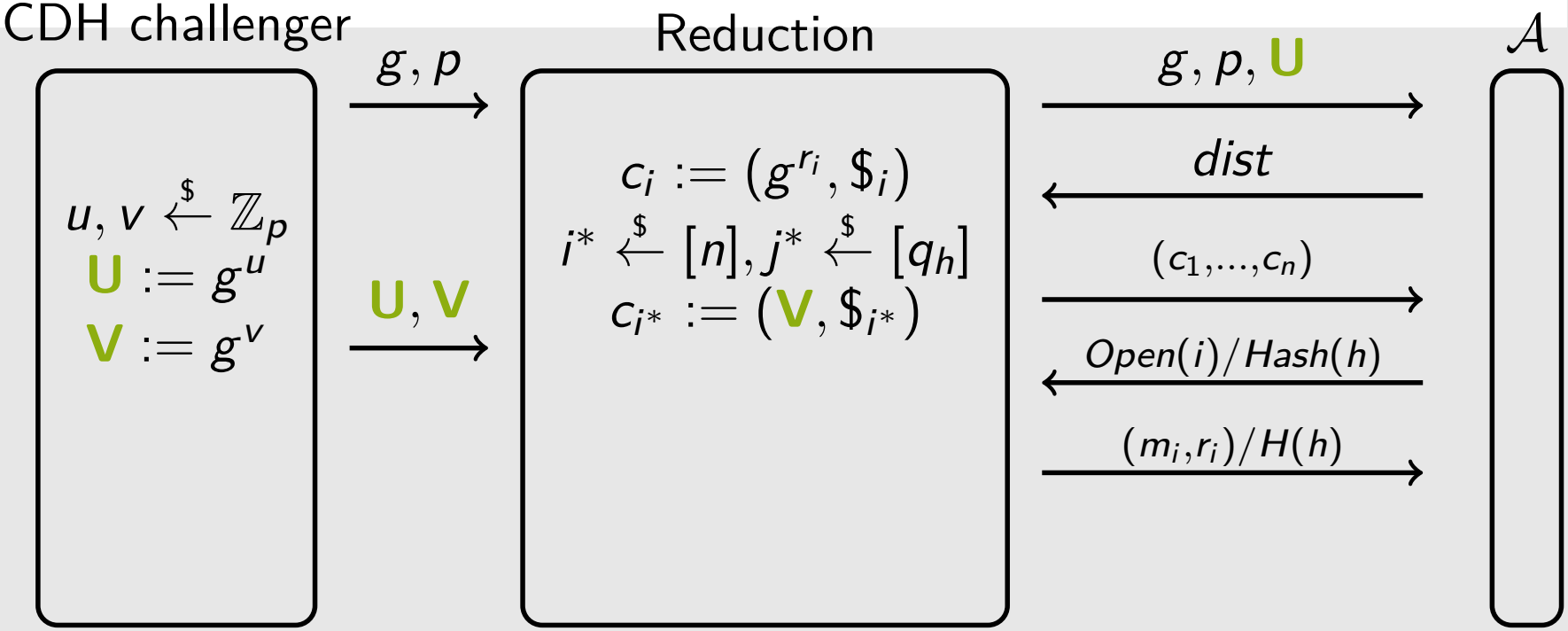
We abort $\mathcal{A}$ if $\mathcal{A}$ calls $H(X^{r_i})$ and did not call $Open(i)$ before.

$$\Pr[AbortH] \leq negl.$$

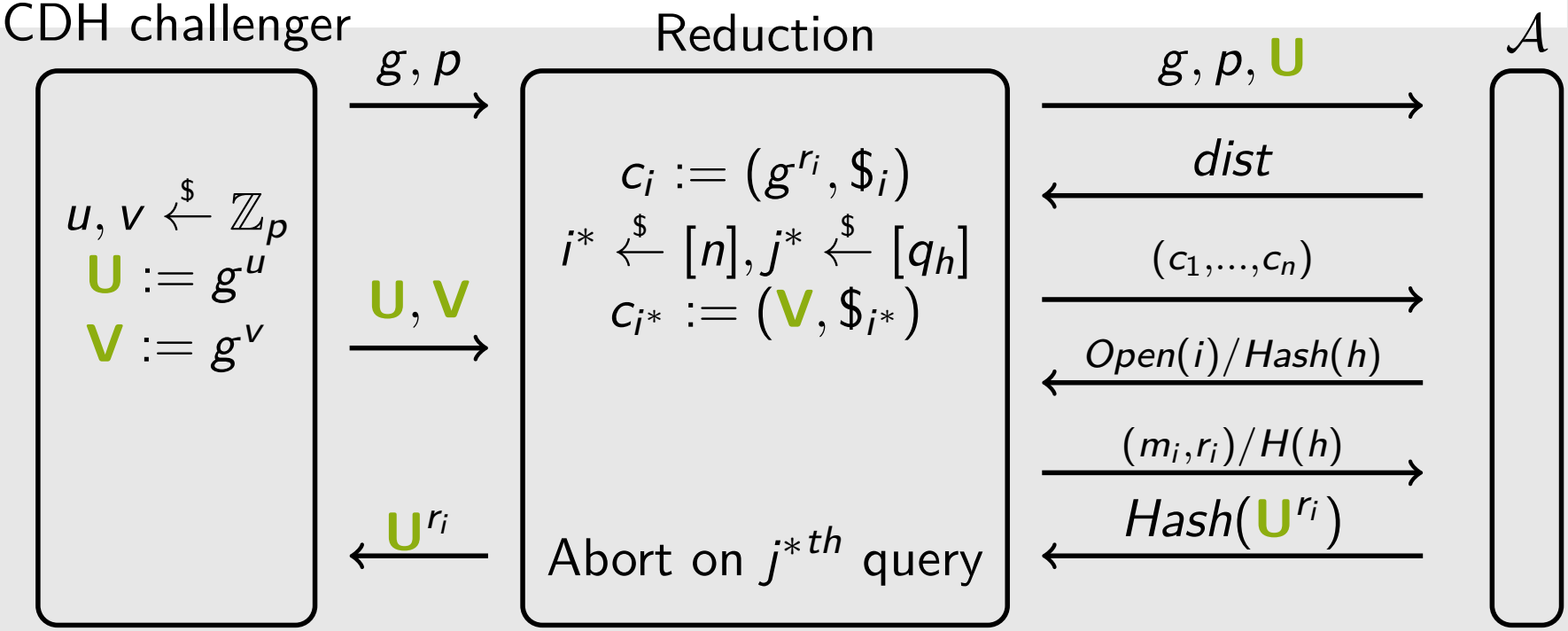
Proof.


CDH challenger
Reduction
$\mathcal{A}$
g, p
$g, p, \mathbf{U}$
$u, v \xleftarrow{\$} \mathbb{Z}_p$
$\mathbf{U} := g^u$
$\mathbf{V} := g^v$
$\mathbf{U}, \mathbf{V}$
$c_i := (g^{r_i}, \$_i)$
$dist$
$(c_1,\ldots,c_n)$
$Open(i)/Hash(h)$
$(m_i,r_i)/H(h)$

Proof.

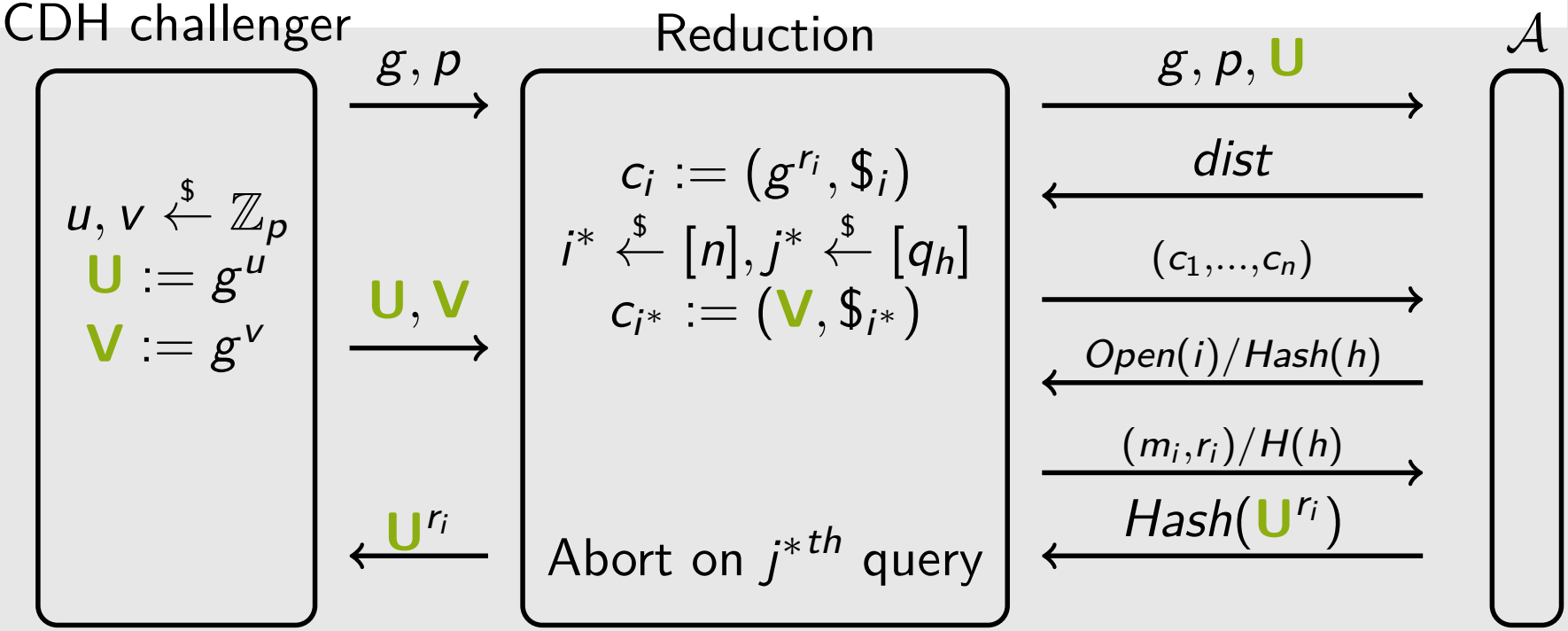

CDH challenger
Reduction
$\mathcal{A}$
g, p
$u, v \xleftarrow{\$} \mathbb{Z}_p$
$\mathbf{U} := g^u$
$\mathbf{V} := g^v$
$\mathbf{U}, \mathbf{V}$
$c_i := (g^{r_i}, \$_i)$
$i^* \xleftarrow{\$} [n], j^* \xleftarrow{\$} [q_h]$
$c_{i^*} := (\mathbf{V}, \$_{i^*})$
$g, p, \mathbf{U}$
$dist$
$(c_1,\ldots,c_n)$
$Open(i)/Hash(h)$
$(m_i,r_i)/H(h)$

Proof.



We have to hide our own challenge in the right ciphertext: $1/n$

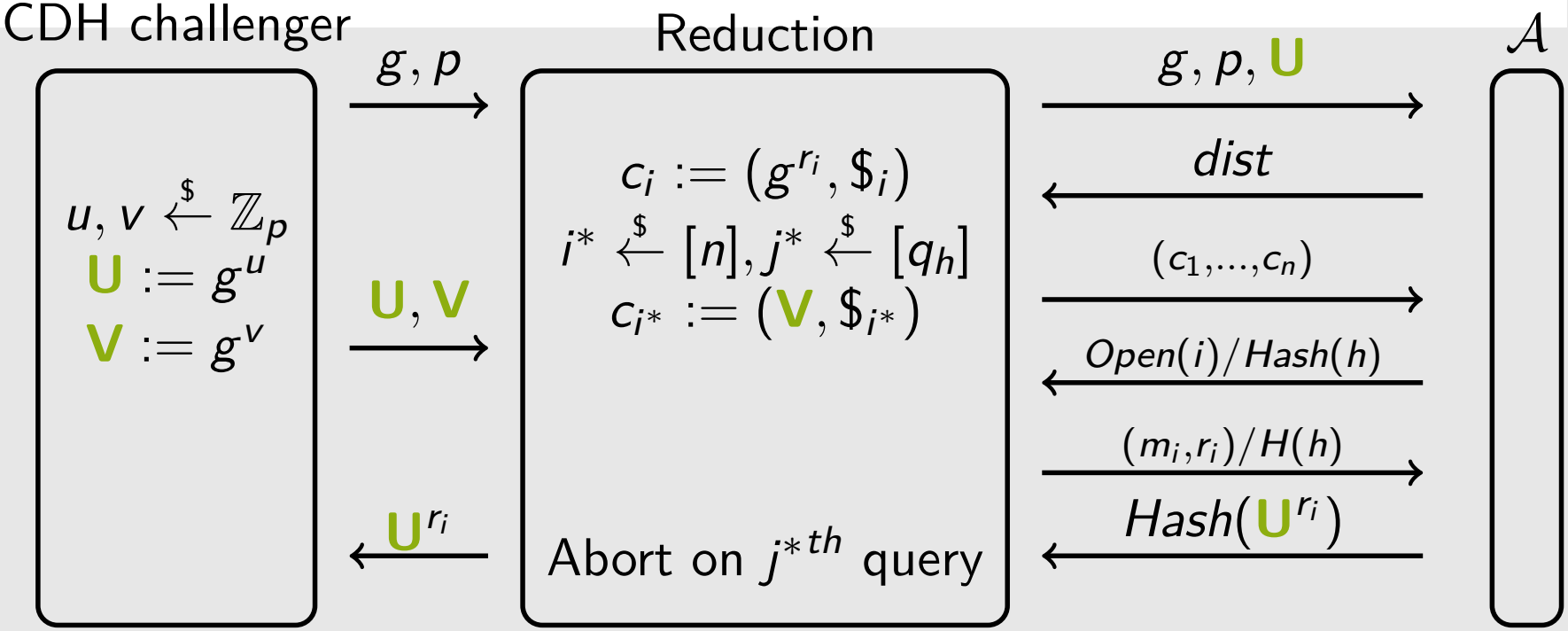
Proof.



We have to hide our own challenge in the right ciphertext: $1/n$

Have to abort on the right query: $1/q_h$

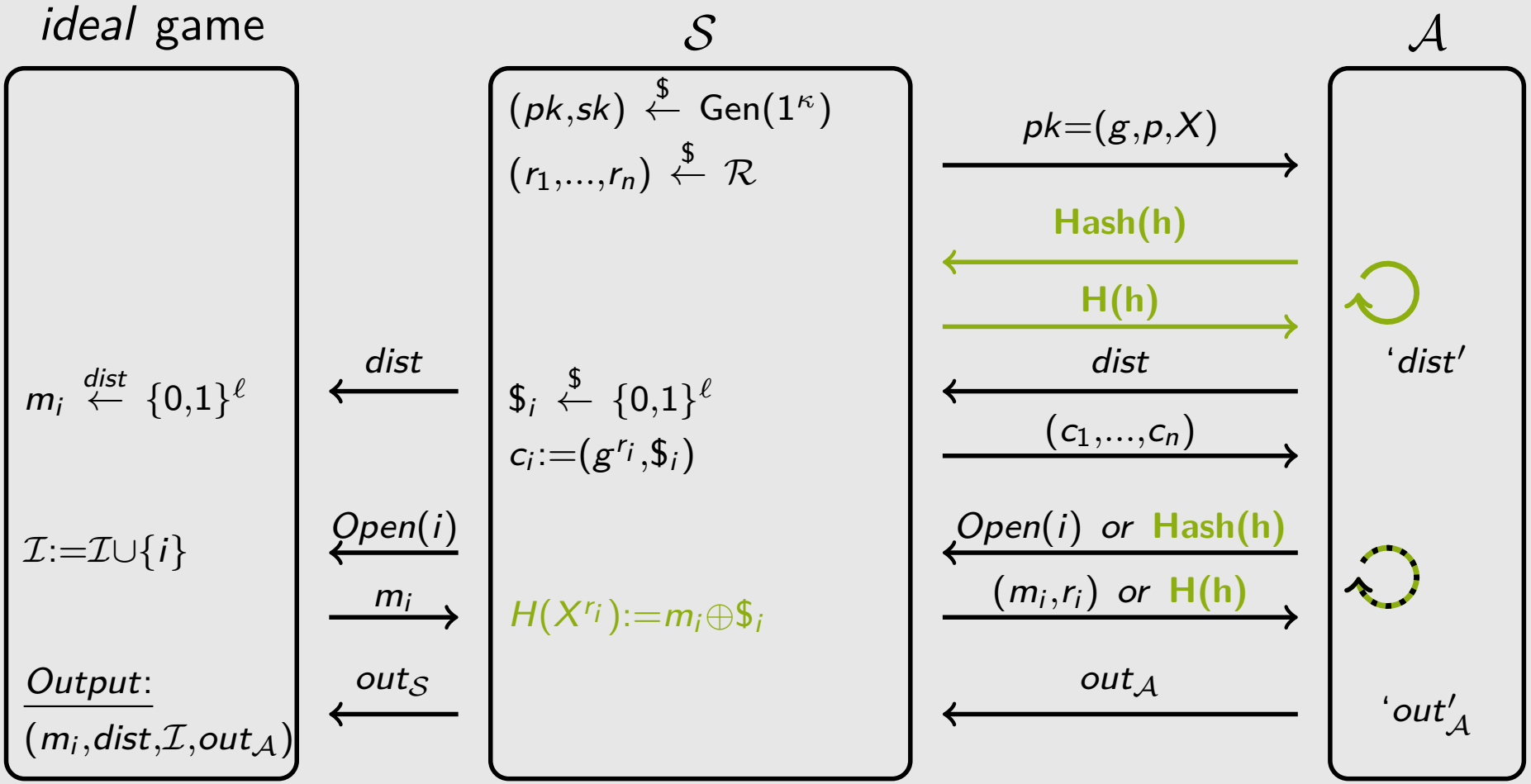
Proof.



We have to hide our own challenge in the right ciphertext: $1/n$
Have to abort on the right query: $1/q_h$

$$\Pr[AbortH] \leq n \cdot q_h \cdot \mathbf{Adv}_g^{\mathrm{CDH}}(\mathcal{B})$$

Proof.


ideal game
$\mathcal{S}$
$\mathcal{A}$
$(pk,sk) \xleftarrow{\$} \mathrm{Gen}(1^\kappa)$
$(r_1,\ldots,r_n) \xleftarrow{\$} \mathcal{R}$
$pk=(g,p,X)$
Hash(h)
H(h)
$m_i \xleftarrow{dist} \{0,1\}^\ell$
dist
$\$_i \xleftarrow{\$} \{0,1\}^\ell$
$c_i:=(g^{r_i},\$_i)$
dist
$(c_1,\ldots,c_n)$
'dist'
$\mathcal{I}:=\mathcal{I}\cup\{i\}$
Open(i)
Open(i) or Hash(h)
m_i
$H(X^{r_i}):=m_i\oplus\$_i$
(m_i,r_i) or H(h)
$\underline{Output:}$
$(m_i,dist,\mathcal{I},out_\mathcal{A})$
$out_\mathcal{S}$
$out_\mathcal{A}$
'$out'_\mathcal{A}$

Results

- DHIES is SIM-SO-CPA secure in the ROM.

Results

- DHIES is SIM-SO-CPA secure in the ROM.
- Actually, DHIES is SIM-SO-CCA secure in the ROM.

Results

- DHIES is SIM-SO-CPA secure in the ROM.
- Actually, DHIES is SIM-SO-CCA secure in the ROM.
- Actually, there is a well known transformation

$$\text{OW-CPA KEM} + \text{sUF-CMA MAC} \rightsquigarrow \text{IND-CCA PKE}$$

we can proof to achieve SIM-SO-CCA security in the ROM without additional assumptions.

Results

- DHIES is SIM-SO-CPA secure in the ROM.
- Actually, DHIES is SIM-SO-CCA secure in the ROM.
- Actually, there is a well known transformation

$$\text{OW-CPA KEM} + \text{sUF-CMA MAC} \rightsquigarrow \text{IND-CCA PKE}$$

  we can proof to achieve SIM-SO-CCA security in the ROM without additional assumptions.
- Actually, we (Jager, Schäge) can proof the widely used RSA OAEP to be SIM-SO-CCA secure in the ROM as well.

Results

- DHIES is SIM-SO-CPA secure in the ROM.
- Actually, DHIES is SIM-SO-CCA secure in the ROM.
- Actually, there is a well known transformation

$$\text{OW-CPA KEM} + \text{sUF-CMA MAC} \rightsquigarrow \text{IND-CCA PKE}$$

  we can proof to achieve SIM-SO-CCA security in the ROM without additional assumptions.
- Actually, we (Jager, Schäge) can proof the widely used RSA OAEP to be SIM-SO-CCA secure in the ROM as well.

SIM-SO-CCA security for free in the ROM

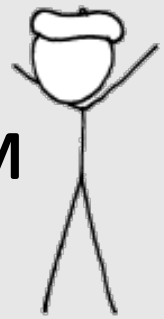


Image source: xkcd.com

RUHR-UNIVERSITÄT BOCHUM

Many thanks for your attention!

QUESTIONS?

Horst Görtz Institut für IT-Sicherheit