RUB

RUHR-UNIVERSITÄT BOCHUM

On the Selective Opening Security of Practical Public-Key Encryption Schemes

PKC 2015, NIST, Maryland: March 30, 2015

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

hgi
Horst Görtz Institut
für IT-Sicherheit

Horst Görtz Institut für IT-Sicherheit

Selective-Opening Attacks

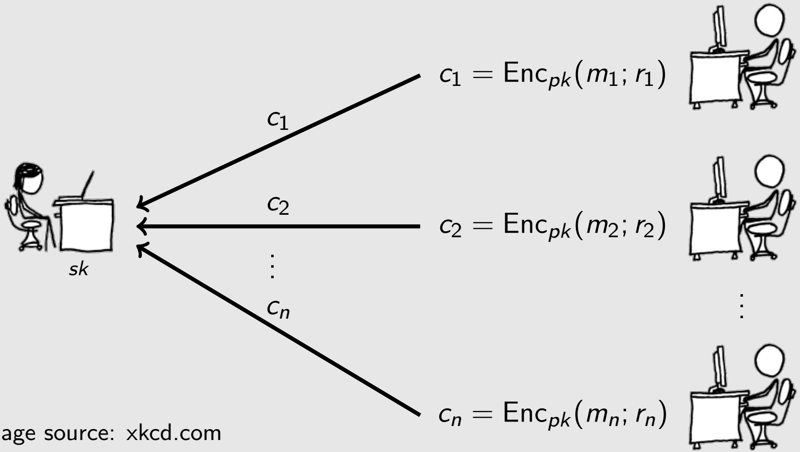


Image source: xkcd.com

Selective-Opening Attacks

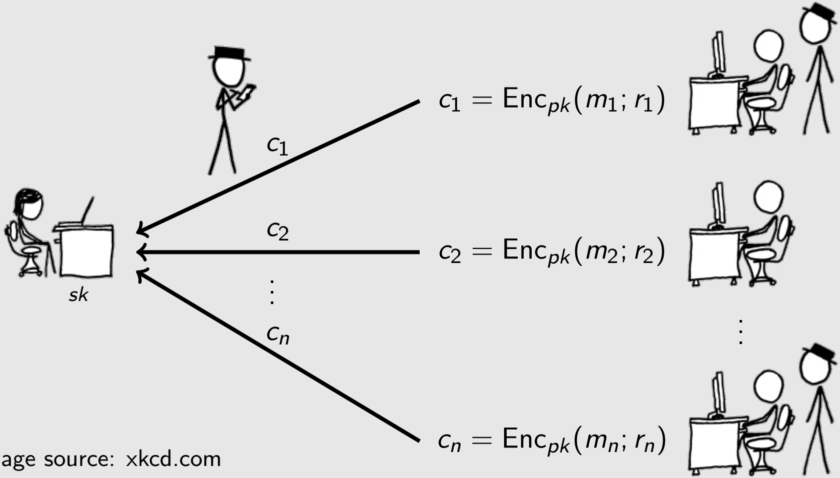
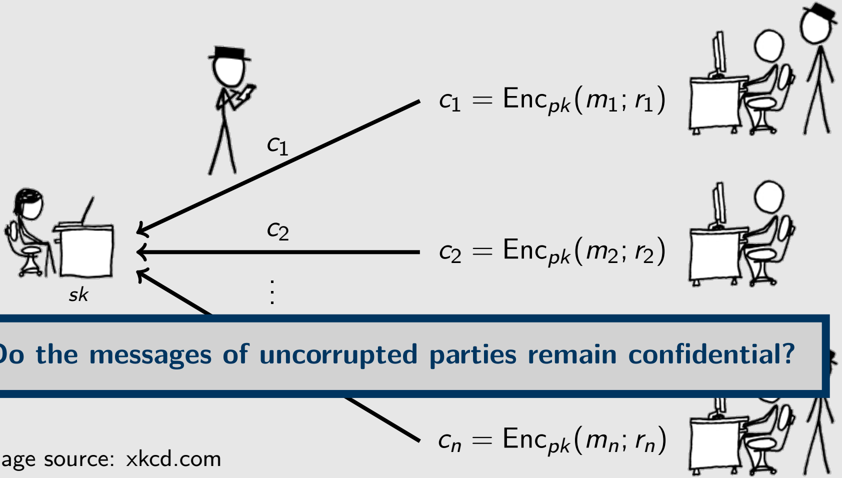


Image source: xkcd.com

Selective-Opening Attacks



Do the messages of uncorrupted parties remain confidential?

Image source: xkcd.com

SIM-SO-CCA Security Definition [FHKW10]

real game

$\mathcal{A}$

$(pk, sk) \leftarrow \mathrm{Gen}(1^\kappa)$

pk →

$(m_1, \dots, m_n) \leftarrow \mathfrak{D}$

← $\mathfrak{D}$

choose distribution $\mathfrak{D}$

$(r_1, \dots, r_n) \overset{\$}{\leftarrow} \mathit{Coins}$

$c_i := \mathrm{Enc}_{pk}(m_i; r_i)$

$(c_1, \dots, c_n)$ →

← $\mathit{Open}(i)$

$\mathcal{I} := \mathcal{I} \cup \{i\}$

(m_i, r_i) →

← $\mathit{out}_{\mathcal{A}}$

compute output "$\mathit{out}_{\mathcal{A}}$"

Output:
$(m_1, \dots, m_n, \mathfrak{D}, \mathcal{I}, \mathit{out}_{\mathcal{A}})$

Dec oracle

← $c \notin \{c_1, \dots, c_n\}$

$\mathit{Dec}_{sk}(c)$ →

SIM-SO-CCA Security Definition [FHKW10]

real game

$\mathcal{A}$

$(pk, sk) \leftarrow \mathrm{Gen}(1^\kappa)$ — pk $\longrightarrow$

$(m_1, \ldots, m_n) \leftarrow \mathfrak{D}$ — $\longleftarrow \mathfrak{D}$ — choose distribution $\mathfrak{D}$

$(r_1, \ldots, r_n) \overset{\$}{\leftarrow} \mathit{Coins}$

$c_i := \mathrm{Enc}_{pk}(m_i; r_i)$ — $(c_1, \ldots, c_n)$ $\longrightarrow$

$\mathcal{I} := \mathcal{I} \cup \{i\}$ — $\longleftarrow \mathit{Open}(i)$

(m_i, r_i) $\longrightarrow$

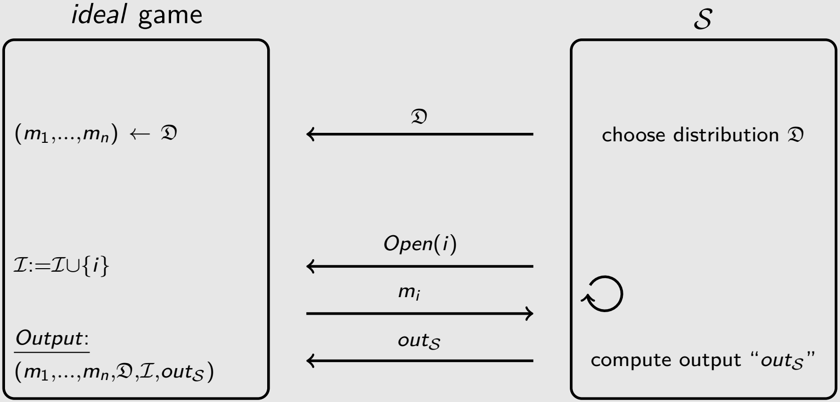
$\longleftarrow \mathit{out}_\mathcal{A}$ — compute output "$\mathit{out}_\mathcal{A}$"

<u>*Output*</u>:

$(m_1, \ldots, m_n, \mathfrak{D}, \mathcal{I}, \mathit{out}_\mathcal{A})$

Dec oracle — $\longleftarrow c \notin \{c_1, \ldots, c_n\}$ — $\mathit{Dec}_{sk}(c) \longrightarrow$

SIM-SO-CCA Security Definition [FHKW10]



SIM-SO-CCA Security Definition [FHKW10]

Definition 1 (SIM-SO-CCA security)

A public key encryption scheme is SIM-SO-CCA 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.

Selective Opening vs. Standard Security Notions

- two flavours: IND-based and SIM-based.
  SIM implies IND, not known to be equivalent.

Selective Opening vs. Standard Security Notions

- two flavours: IND-based and SIM-based.
  SIM implies IND, not known to be equivalent.
- IND-SO-CCA stronger security notion than IND-CCA security. [HR14]

Selective Opening vs. Standard Security Notions

- two flavours: IND-based and SIM-based.
  SIM implies IND, not known to be equivalent.
- IND-SO-CCA stronger security notion than IND-CCA security. [HR14]
- Existing SIM-SO-CCA schemes are very inefficient. [FHKW10], [Hof12], [LP15]

Selective Opening vs. Standard Security Notions

- two flavours: IND-based and SIM-based.
  SIM implies IND, not known to be equivalent.
- IND-SO-CCA stronger security notion than IND-CCA security. [HR14]
- Existing SIM-SO-CCA schemes are very inefficient. [FHKW10], [Hof12], [LP15]
- **This work:** Certain known constructions give SIM-SO-CCA security for free in the ROM.

Our Work

Part I:

PKE from any one-way PCA secure KEM

Part II:

PKE from OAEP for any partial-domain TP

Our Work

Part I:

PKE from any one-way PCA secure KEM

Part II:

PKE from OAEP for any partial-domain TP

“Hashed KEM/DEM Approach”

Let $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ be a Key Encapsulation Mechanism, $\mathsf{MAC} = (\mathsf{Tag}, \mathsf{Vrfy})$ a Message Authentication Code and H a hash function. Consider the following PKE:

"Hashed KEM/DEM Approach"

Let $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$ be a Key Encapsulation Mechanism, $\mathsf{MAC} = (\mathsf{Tag}, \mathsf{Vrfy})$ a Message Authentication Code and H a hash function. Consider the following PKE:

$\underline{\mathsf{Gen}(1^\lambda)}$
$(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$
Return pk

“Hashed KEM/DEM Approach”

Let KEM $= (\text{KGen}, \text{Encap}, \text{Decap})$ be a Key Encapsulation Mechanism, MAC $= (\text{Tag}, \text{Vrfy})$ a Message Authentication Code and H a hash function. Consider the following PKE:

$\underline{\text{Gen}(1^\lambda)}$

$(pk, sk) \leftarrow \text{KGen}(1^\lambda)$
Return pk

$\underline{\text{Enc}_{pk}(m)}$

$r \stackrel{\$}{\leftarrow} Coins$
$(k, c^{(1)}) \leftarrow \text{Encap}_{pk}(r)$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
$c^{(2)} := m \oplus k^{sym}$
$c^{(3)} := \text{Tag}_{k^{mac}}(c^{(2)})$
Return $(c^{(1)}, c^{(2)}, c^{(3)})$

“Hashed KEM/DEM Approach”

Let KEM $=$ (KGen, Encap, Decap) be a Key Encapsulation Mechanism, MAC $=$ (Tag, Vrfy) a Message Authentication Code and H a hash function. Consider the following PKE:

$\underline{\text{Gen}(1^\lambda)}$

$(pk, sk) \leftarrow \text{KGen}(1^\lambda)$
Return pk

$\underline{\text{Enc}_{pk}(m)}$

$r \xleftarrow{\$} Coins$
$(k, c^{(1)}) \leftarrow \text{Encap}_{pk}(r)$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
$c^{(2)} := m \oplus k^{sym}$
$c^{(3)} := \text{Tag}_{k^{mac}}(c^{(2)})$
Return $(c^{(1)}, c^{(2)}, c^{(3)})$

$\underline{\text{Dec}_{sk}(c^{(1)}, c^{(2)}, c^{(3)})}$

$k \leftarrow \text{Decap}_{sk}(c^{(1)})$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
if $\text{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 1$
    Return $c^{(2)} \oplus k^{sym}$
else
    Return $\perp$

"Hashed KEM/DEM Approach"

Theorem 2 (SBZ02)

The given transformation achieves IND-CCA security in the ROM if KEM *is OW-PCA secure and* MAC *is sUF-OT-CMA.*

$\underline{\mathsf{Gen}(1^\lambda)}$

$(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$

Return pk

$\underline{\mathsf{Enc}_{pk}(m)}$

$r \xleftarrow{\$} Coins$

$(k, c^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r)$

$(k^{sym}, k^{mac}) \leftarrow H(k)$

$c^{(2)} := m \oplus k^{sym}$

$c^{(3)} := \mathsf{Tag}_{k^{mac}}(c^{(2)})$

Return $(c^{(1)}, c^{(2)}, c^{(3)})$

$\underline{\mathsf{Dec}_{sk}(c^{(1)}, c^{(2)}, c^{(3)})}$

$k \leftarrow \mathsf{Decap}_{sk}(c^{(1)})$

$(k^{sym}, k^{mac}) \leftarrow H(k)$

if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 1$

  Return $c^{(2)} \oplus k^{sym}$

else

  Return $\perp$

"Hashed KEM/DEM Approach"

Theorem 2 (SBZ02)

The given transformation achieves IND-CCA security in the ROM if KEM *is OW-PCA secure and* MAC *is sUF-OT-CMA.*

$\underline{\mathsf{Gen}(1^\lambda)}$
$(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$
Return pk

$\underline{\mathsf{Enc}_{pk}(m)}$
$r \xleftarrow{\$} Coins$
$(k, c^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r)$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
$c^{(2)} := m \oplus k^{sym}$
$c^{(3)} := \mathsf{Tag}_{k^{mac}}(c^{(2)})$
Return $(c^{(1)}, c^{(2)}, c^{(3)})$

$\underline{\mathsf{Dec}_{sk}(c^{(1)}, c^{(2)}, c^{(3)})}$
$k \leftarrow \mathsf{Decap}_{sk}(c^{(1)})$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 1$
    Return $c^{(2)} \oplus k^{sym}$
else
    Return $\perp$

– Use RSA KEM under RSA assumption

“Hashed KEM/DEM Approach”

Theorem 2 (SBZ02)

The given transformation achieves IND-CCA security in the ROM if KEM *is OW-PCA secure and* MAC *is sUF-OT-CMA.*

$\underline{\mathsf{Gen}(1^\lambda)}$
$(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$
Return pk

$\underline{\mathsf{Enc}_{pk}(m)}$
$r \xleftarrow{\$} Coins$
$(k, c^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r)$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
$c^{(2)} := m \oplus k^{sym}$
$c^{(3)} := \mathsf{Tag}_{k^{mac}}(c^{(2)})$
Return $(c^{(1)}, c^{(2)}, c^{(3)})$

$\underline{\mathsf{Dec}_{sk}(c^{(1)}, c^{(2)}, c^{(3)})}$
$k \leftarrow \mathsf{Decap}_{sk}(c^{(1)})$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 1$
  Return $c^{(2)} \oplus k^{sym}$
else
  Return $\perp$

- Use RSA KEM under RSA assumption
- Use DH KEM under strong DH assumption

Instantiation of DHIES

"Hashed KEM/DEM Approach"

Theorem 3 (This work)

The same transformation gives rise to a SIM-SO-CCA secure PKE in the ROM without additional assumptions.

$\underline{\mathsf{Gen}(1^\lambda)}$

$(pk, sk) \leftarrow \mathsf{KGen}(1^\lambda)$

Return pk

$\underline{\mathsf{Enc}_{pk}(m)}$

$r \xleftarrow{\$} Coins$

$(k, c^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r)$

$(k^{sym}, k^{mac}) \leftarrow H(k)$

$c^{(2)} := m \oplus k^{sym}$

$c^{(3)} := \mathsf{Tag}_{k^{mac}}(c^{(2)})$

Return $(c^{(1)}, c^{(2)}, c^{(3)})$

$\underline{\mathsf{Dec}_{sk}(c^{(1)}, c^{(2)}, c^{(3)})}$

$k \leftarrow \mathsf{Decap}_{sk}(c^{(1)})$

$(k^{sym}, k^{mac}) \leftarrow H(k)$

if $\mathsf{Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 1$

    Return $c^{(2)} \oplus k^{sym}$

else

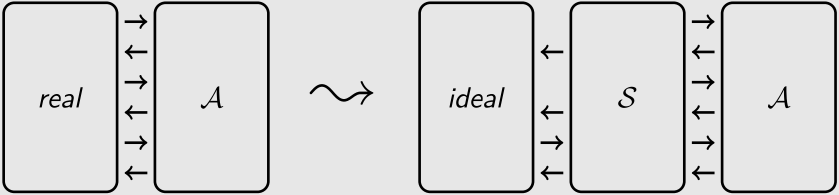
    Return $\perp$

– Use RSA KEM under RSA assumption

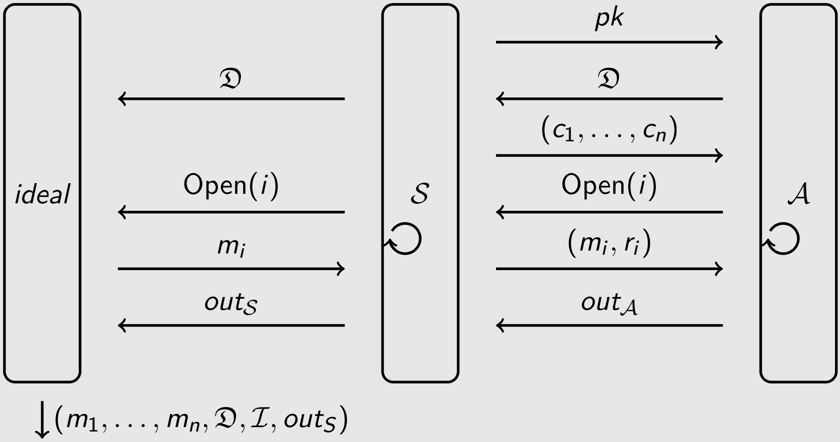
– Use DH KEM under strong DH assumption

Instantiation of DHIES

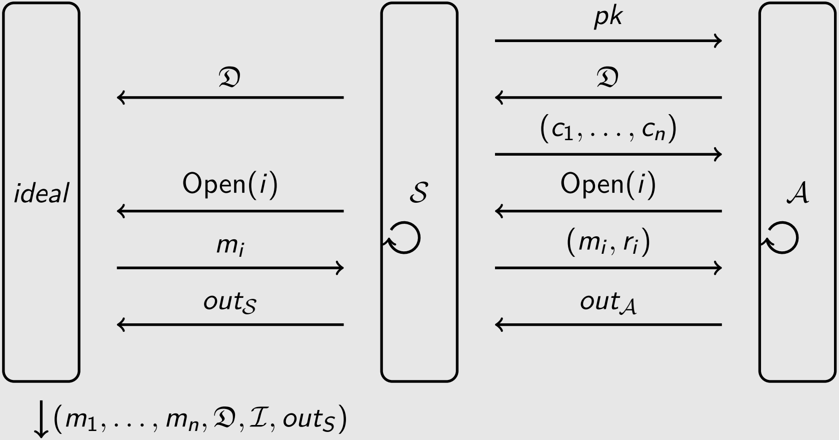
How to Prove SIM-SO-CCA Security



How to Construct a Simulator



How to Construct a Simulator



$\mathcal{A}$ is allowed to make additional **Hash** or **Dec** queries at any time!

Possible Tripping Hazards for a Simulator

- $\mathcal{S}$ must not make more opening queries than $\mathcal{A}$ to learn m_i.

Possible Tripping Hazards for a Simulator

- $\mathcal{S}$ must not make more opening queries than $\mathcal{A}$ to learn m_i.
- $\mathcal{S}$ has to create non-committing "dummy"-encryptions that allow for later opening to any message.

Possible Tripping Hazards for a Simulator

- $\mathcal{S}$ must not make more opening queries than $\mathcal{A}$ to learn m_i.
- $\mathcal{S}$ has to create non-committing "dummy"-encryptions that allow for later opening to any message.
- Answering Hash or Decryption queries is easy if $\mathcal{A}$ called Open(i) earlier.

Possible Tripping Hazards for a Simulator

- $\mathcal{S}$ must not make more opening queries than $\mathcal{A}$ to learn m_i.
- $\mathcal{S}$ has to create non-committing "dummy"-encryptions that allow for later opening to any message.
- Answering Hash or Decryption queries is easy if $\mathcal{A}$ called Open(i) earlier.
- However, $\mathcal{S}$ has to answer Hash and Decryption queries without committing to m_i if $\mathcal{A}$ did not call Open(i) (yet).

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

$\underline{\mathsf{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := m_i \oplus k_i^{sym}$

$c_i^{(3)} := \mathsf{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

G_0

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathsf{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

$\underline{\mathsf{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := m_i \oplus k_i^{sym}$

$c_i^{(3)} := \mathsf{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

G_0 G_1

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := m_i \oplus k_i^{sym}$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1$$

statistical argument

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 \qquad G_2$$

statistical argument

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 \; = \; G_2$$

statistical argument

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.
G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.
G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$
$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$
$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$
$c_i^{(2)} := uniform$
$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$
$\ldots^{(2)}, c_i^{(3)})$

How to answer Hash or Decryption queries?
Assume $\mathcal{A}$ makes a valid $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ query:

G_0

st
ar

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

$\ldots{}^{(2)}, c_i^{(3)})$

G_0

st…

ar…

How to answer Hash or Decryption queries?

Assume $\mathcal{A}$ makes a valid $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ query:

<u>Case 1:</u>

$H(k_i)$ is not defined

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathsf{Dec}(c_i^{(1)},\cdot,\cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathsf{Enc}_{pk}(m_i)}$

$r_i \overset{\$}{\leftarrow} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathsf{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathsf{Tag}_{k^{mac}}(c_i^{(2)})$

$^{(2)}, c_i^{(3)})$

How to answer Hash or Decryption queries?

Assume $\mathcal{A}$ makes a valid $\mathsf{Dec}(c_i^{(1)},\cdot,\cdot)$ query:

<u>Case 1:</u>

$H(k_i)$ is not defined

<u>Case 2:</u>

$H(k_i)$ is defined

G_0

st

ar

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

$\ldots^{(2)}, c_i^{(3)})$

How to answer Hash or Decryption queries?

Assume $\mathcal{A}$ makes a valid $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ query:

Case 1:
$H(k_i)$ is not defined $\rightsquigarrow$ k_i^{mac} still uniform, use MAC security

Case 2:
$H(k_i)$ is defined

G_3

G_0

st...
ar...

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} \mathit{Coins}$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := \mathit{uniform}$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

$\ldots{}^{(2)}, c_i^{(3)})$

G_0

st…
ar…

How to answer Hash or Decryption queries?
Assume $\mathcal{A}$ makes a valid $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ query:

<u>Case 1:</u>
$H(k_i)$ is not defined $\rightsquigarrow$ k_i^{mac} still uniform, use MAC security

<u>Case 2:</u>
$H(k_i)$ is defined $\rightsquigarrow$ $\mathcal{A}$ decapsulated $c_i^{(1)}$, use KEM security

G_3

G_4

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 = G_2$$

statistical argument

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.
G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.
G_2: Replace $c_i^{(2)}$ with uniform randomness.
G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.

$\underline{\mathrm{Enc}_{pk}(m_i)}$
$r_i \stackrel{\$}{\leftarrow} \mathit{Coins}$
$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$
$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$
$c_i^{(2)} := \mathit{uniform}$
$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$
Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 \quad = \quad G_2 \qquad G_3$$

statistical argument

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \stackrel{\$}{\leftarrow} \mathit{Coins}$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := \mathit{uniform}$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 = G_2 \approx_c G_3$$

statistical argument ($G_0 \approx_s G_1$)

MAC security ($G_2 \approx_c G_3$)

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.

G_4: Abort if $\mathcal{A}$ queries $H(k_i)$.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

$$G_0 \approx_s G_1 \;=\; G_2 \approx_c G_3 \qquad G_4$$

statistical argument

MAC security

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.

G_4: Abort if $\mathcal{A}$ queries $H(k_i)$.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

KEM security

$$G_0 \approx_s G_1 = G_2 \approx_c G_3 \approx_c G_4$$

statistical argument

MAC security

Tweaking $\mathcal{A}$ in a Sequence of Games

G_0: *real* SIM-SO-CCA game.

G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.

G_2: Replace $c_i^{(2)}$ with uniform randomness.

G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.

G_4: Abort if $\mathcal{A}$ queries $H(k_i)$.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

KEM security

$$G_0 \approx_s G_1 = G_2 \approx_c G_3 \approx_c G_4$$

statistical argument

MAC security

Oracle patching technique. [CS03] (Needs PCA KEM)

Tweaking $\mathcal{A}$ in a Sequence of Games

- G_0: *real* SIM-SO-CCA game.
- G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.
- G_2: Replace $c_i^{(2)}$ with uniform randomness.
- G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.
- G_4: Abort if $\mathcal{A}$ queries $H(k_i)$.
- G_5: *ideal* SIM-SO-CCA game.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} \mathit{Coins}$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := \mathit{uniform}$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

KEM security

$$G_0 \approx_s G_1 = G_2 \approx_c G_3 \approx_c G_4 \quad G_5$$

statistical argument

MAC security

Oracle patching technique. [CS03] (Needs PCA KEM)

Tweaking $\mathcal{A}$ in a Sequence of Games

- G_0: *real* SIM-SO-CCA game.
- G_1: Abort if $\mathcal{A}$ queries $H(k_i)$ or $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ before sending $\mathfrak{D}$.
- G_2: Replace $c_i^{(2)}$ with uniform randomness.
- G_3: Abort if $\mathcal{A}$ issues a valid decryption query $\mathrm{Dec}(c_i^{(1)}, \cdot, \cdot)$ and $H(k_i)$ is not yet defined.
- G_4: Abort if $\mathcal{A}$ queries $H(k_i)$.
- G_5: *ideal* SIM-SO-CCA game.

$\underline{\mathrm{Enc}_{pk}(m_i)}$

$r_i \xleftarrow{\$} Coins$

$(k_i, c_i^{(1)}) \leftarrow \mathrm{Encap}_{pk}(r_i)$

$(k_i^{sym}, k_i^{mac}) \leftarrow H(k_i)$

$c_i^{(2)} := uniform$

$c_i^{(3)} := \mathrm{Tag}_{k_i^{mac}}(c_i^{(2)})$

Return $(c_i^{(1)}, c_i^{(2)}, c_i^{(3)})$

KEM security

$$G_0 \approx_s G_1 = G_2 \approx_c G_3 \approx_c G_4 = G_5$$

statistical argument

MAC security

Oracle patching technique. [CS03] (Needs PCA KEM)

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security
- Employs OT-MAC – can be constructed information-theoretically

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security
- Employs OT-MAC – can be constructed information-theoretically
- Can be instantiated with any OW-PCA secure KEM, such as RSA KEM or DH KEM, latter gives instantiation of DHIES.

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security
- Employs OT-MAC – can be constructed information-theoretically
- Can be instantiated with any OW-PCA secure KEM, such as RSA KEM or DH KEM, latter gives instantiation of DHIES.
- Same proof-ideas can be applied to prove RSA-OAEP SIM-SO-CCA secure in the ROM.

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security
- Employs OT-MAC – can be constructed information-theoretically
- Can be instantiated with any OW-PCA secure KEM, such as RSA KEM or DH KEM, latter gives instantiation of DHIES.
- Same proof-ideas can be applied to prove RSA-OAEP SIM-SO-CCA secure in the ROM.

Certain tranformations give SIM-SO-CCA security for free in the ROM

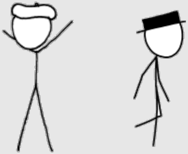


Image source: xkcd.com

Summary

- SIM-SO-CCA secure PKE from generic construction in the ROM
- No additional assumptions compared to proof for IND-CCA security
- Employs OT-MAC – can be constructed information-theoretically
- Can be instantiated with any OW-PCA secure KEM, such as RSA KEM or DH KEM, latter gives instantiation of DHIES.
- Same proof-ideas can be applied to prove RSA-OAEP SIM-SO-CCA secure in the ROM.

Thank you!

Certain tranformations give SIM-SO-CCA security for free in the ROM

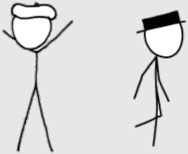


Image source: xkcd.com