Joint State Theorems for Public-Key Encryption and Digital Signature Functionalities with Local Computation

Max Tuengerthal

joint work with Ralf Küsters

ETH Zurich

FormaCrypt Meeting - 30. November 2007

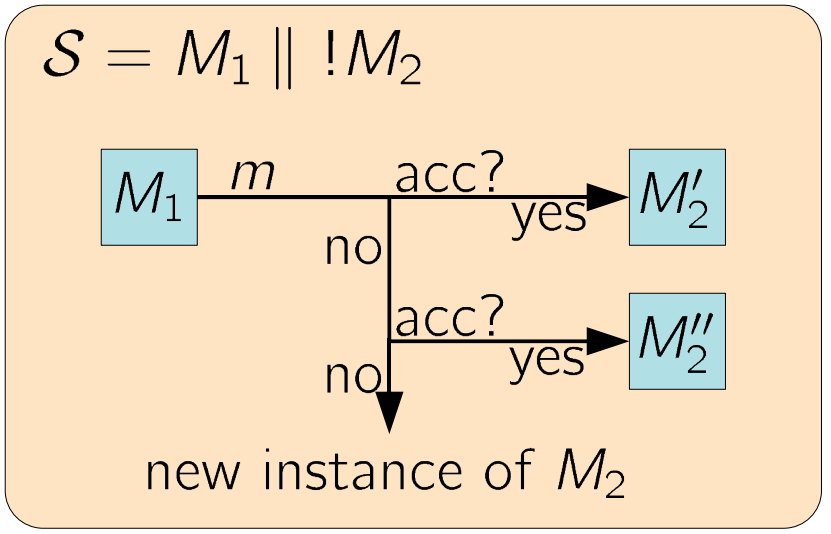
IITM Model [Küsters 2006]

- Users/Machines modeled by PPT-IITMs
  - Inexhaustable ITM (IITM)
  - IITMs connect via tapes
- System of IITMs $\mathcal{S} ::= M \mid (\mathcal{S} \,\|\, \mathcal{S}) \mid !\mathcal{S}$
- Characteristics:
  - Asynchronous communication
  - Model independent of security definition
  - Simple and yet expressive

IITM Model [Küsters 2006]

- Users/Machines modeled by PPT-IITMs
  - Inexhaustable ITM (IITM)
  - IITMs connect via tapes
- System of IITMs $\mathcal{S} ::= M \mid (\mathcal{S} \,\|\, \mathcal{S}) \mid !\mathcal{S}$
- Characteristics:
  - Asynchronous communication
  - Model independent of security definition
  - Simple and yet expressive
- Generic addressing method for multiple instances

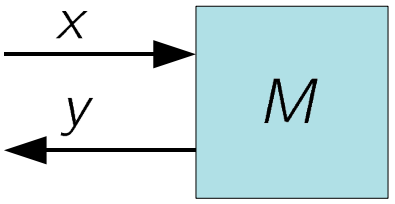
Two modes: CheckAddress
Compute

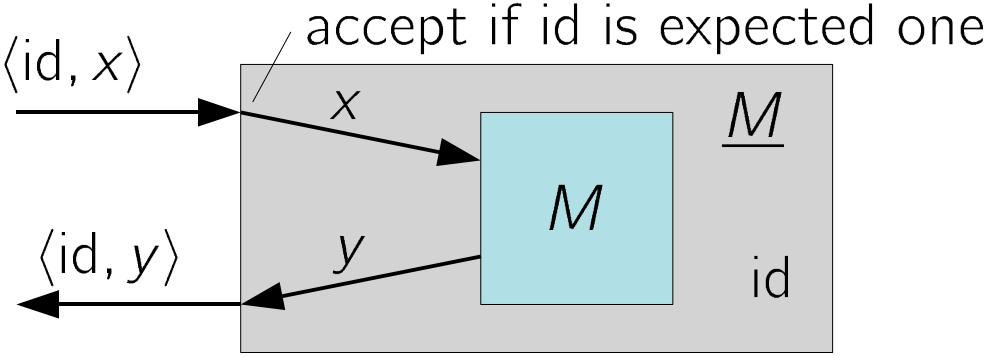


IITM Model [Küsters 2006]

- Users/Machines modeled by PPT-IITMs
  - Inexhaustable ITM (IITM)
  - IITMs connect via tapes
- System of IITMs $\mathcal{S} ::= M \mid (\mathcal{S} \,\|\, \mathcal{S}) \mid !\mathcal{S}$
- Characteristics:
  - Asynchronous communication
  - Model independent of security definition
  - Simple and yet expressive
- Generic addressing method for multiple instances

ID version: $\underline{M}$



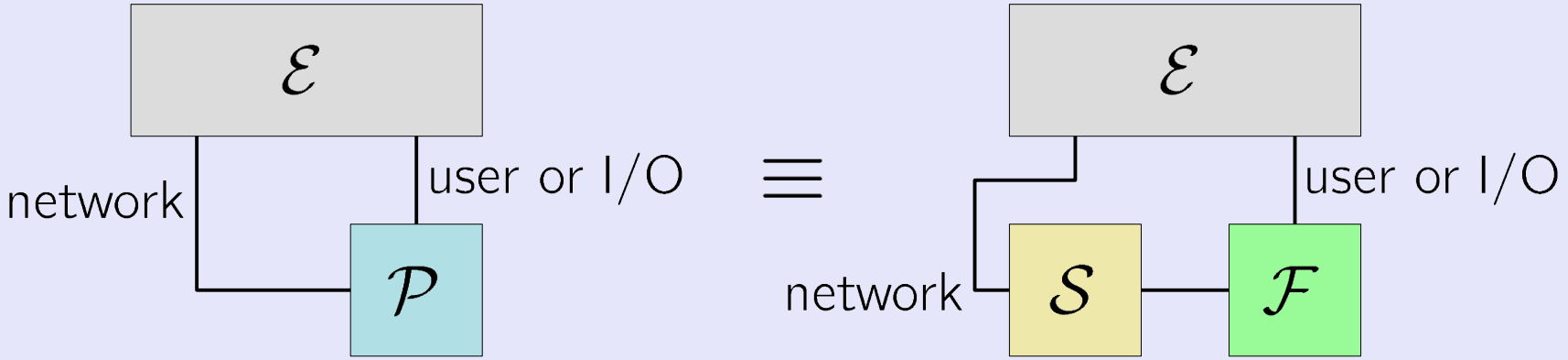


Note: ID can be SID or PID

Security Definition

Definition (Strong Simulatability)

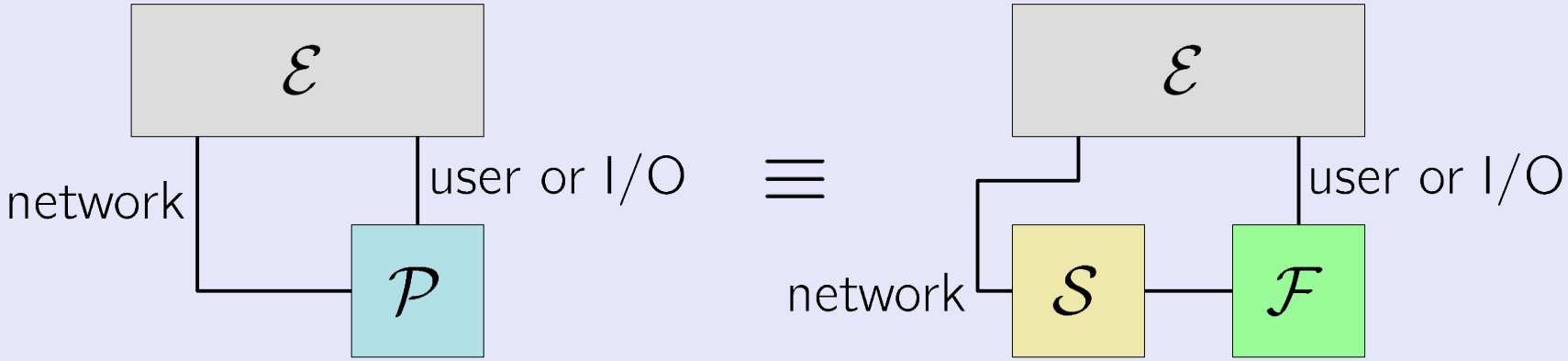
$\mathcal{P} \leq^{\mathsf{SS}} \mathcal{F}$ iff $\exists$ Sim. $\mathcal{S}$ $\forall$ Env. $\mathcal{E}$: $\mathcal{E} \,\|\, \mathcal{P} \equiv \mathcal{E} \,\|\, \mathcal{S} \,\|\, \mathcal{F}$



Security Definition

Definition (Strong Simulatability)

$\mathcal{P} \leq^{\mathrm{SS}} \mathcal{F}$ iff $\exists$ Sim. $\mathcal{S}$ $\forall$ Env. $\mathcal{E}$: $\mathcal{E} \,\|\, \mathcal{P} \equiv \mathcal{E} \,\|\, \mathcal{S} \,\|\, \mathcal{F}$



- Transitive and reflexive
- Notion conceptual equiv. to blackbox simulatability [Pfitzmann and Waidner 2001], UC [Canetti 2001]
- Corruption described in the formulation of $\mathcal{F}/\mathcal{P}$

Composition Theorem

Composition Theorem [Küsters 2006]

$\mathcal{P}_1 \leq^{\mathsf{SS}} \mathcal{F}_1$, $\mathcal{P}_2 \leq^{\mathsf{SS}} \mathcal{F}_2$ implies

$$\mathcal{P}_1 \,\|\, \mathcal{P}_2 \leq^{\mathsf{SS}} \mathcal{F}_1 \,\|\, \mathcal{F}_2 \quad \text{and} \quad !\underline{\mathcal{P}_1} \leq^{\mathsf{SS}} !\underline{\mathcal{F}_1}$$

- Connection between $\mathcal{F}_1/\mathcal{P}_1$ and $\mathcal{F}_2/\mathcal{P}_2$ arbitrary

Composition Theorem

Composition Theorem [Küsters 2006]

$\mathcal{P}_1 \leq^{\mathrm{SS}} \mathcal{F}_1$, $\mathcal{P}_2 \leq^{\mathrm{SS}} \mathcal{F}_2$ implies

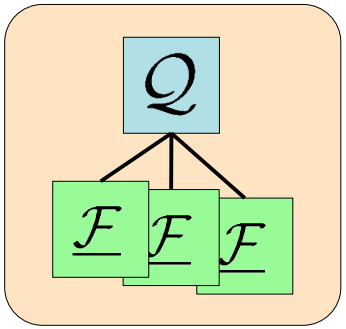
$$\mathcal{P}_1 \,\|\, \mathcal{P}_2 \leq^{\mathrm{SS}} \mathcal{F}_1 \,\|\, \mathcal{F}_2 \quad \text{and} \quad !\underline{\mathcal{P}_1} \leq^{\mathrm{SS}} !\underline{\mathcal{F}_1}$$

- Connection between $\mathcal{F}_1/\mathcal{P}_1$ and $\mathcal{F}_2/\mathcal{P}_2$ arbitrary

Corollary

$$\mathcal{P} \leq^{\mathrm{SS}} \mathcal{F} \quad \text{implies} \quad \mathcal{Q} \,\|\, !\underline{\mathcal{P}} \leq^{\mathrm{SS}} \mathcal{Q} \,\|\, !\underline{\mathcal{F}}$$

- $\mathcal{Q}$ uses multiple instances of $\mathcal{F}/\mathcal{P}$
- Corresponds to comp. thm. in UC model



Motivating Joint State

Example: Protocol $\mathcal{P}$ that uses public key encryption (PKE)

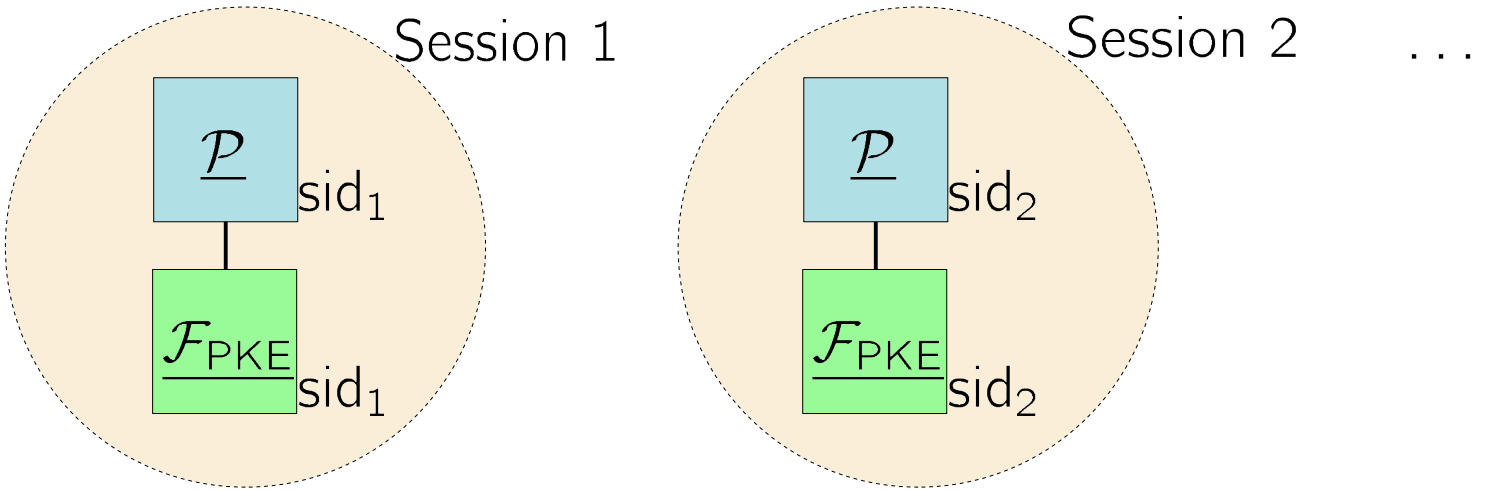
- Replace use of PKE by calls to ideal functionality $\mathcal{F}_{\text{PKE}}$
  - Simplifies security analysis
  - Use composition theorem to obtain security of $\mathcal{P}$
- Analyze only single instance (session) of $\mathcal{P}$
  - Security of multiple sessions ($!\underline{\mathcal{P}}$) by composition theorem

Motivating Joint State

Example: Protocol $\mathcal{P}$ that uses public key encryption (PKE)

- Replace use of PKE by calls to ideal functionality $\mathcal{F}_{\text{PKE}}$
  - Simplifies security analysis
  - Use composition theorem to obtain security of $\mathcal{P}$
- Analyze only single instance (session) of $\mathcal{P}$
  - Security of multiple sessions ($!\underline{\mathcal{P}}$) by composition theorem

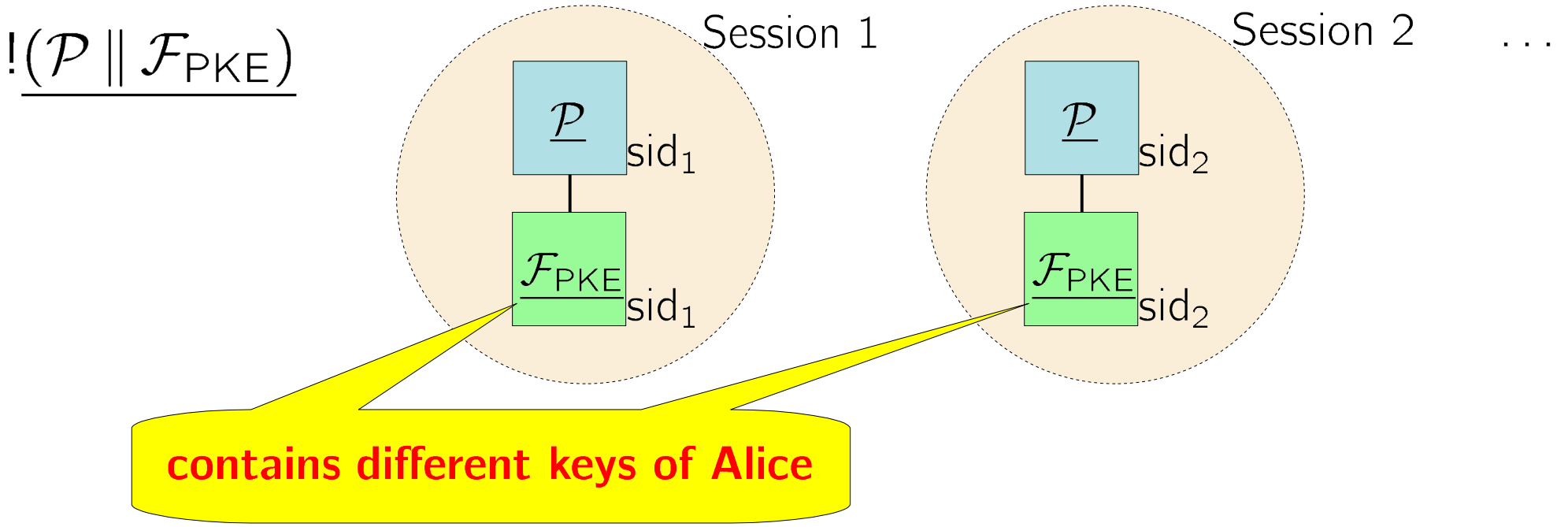
$!\underline{(\mathcal{P} \,\|\, \mathcal{F}_{\text{PKE}})}$



Motivating Joint State

Example: Protocol $\mathcal{P}$ that uses public key encryption (PKE)

- Replace use of PKE by calls to ideal functionality $\mathcal{F}_{\text{PKE}}$
  - Simplifies security analysis
  - Use composition theorem to obtain security of $\mathcal{P}$
- Analyze only single instance (session) of $\mathcal{P}$
  - Security of multiple sessions ($!\underline{\mathcal{P}}$) by composition theorem

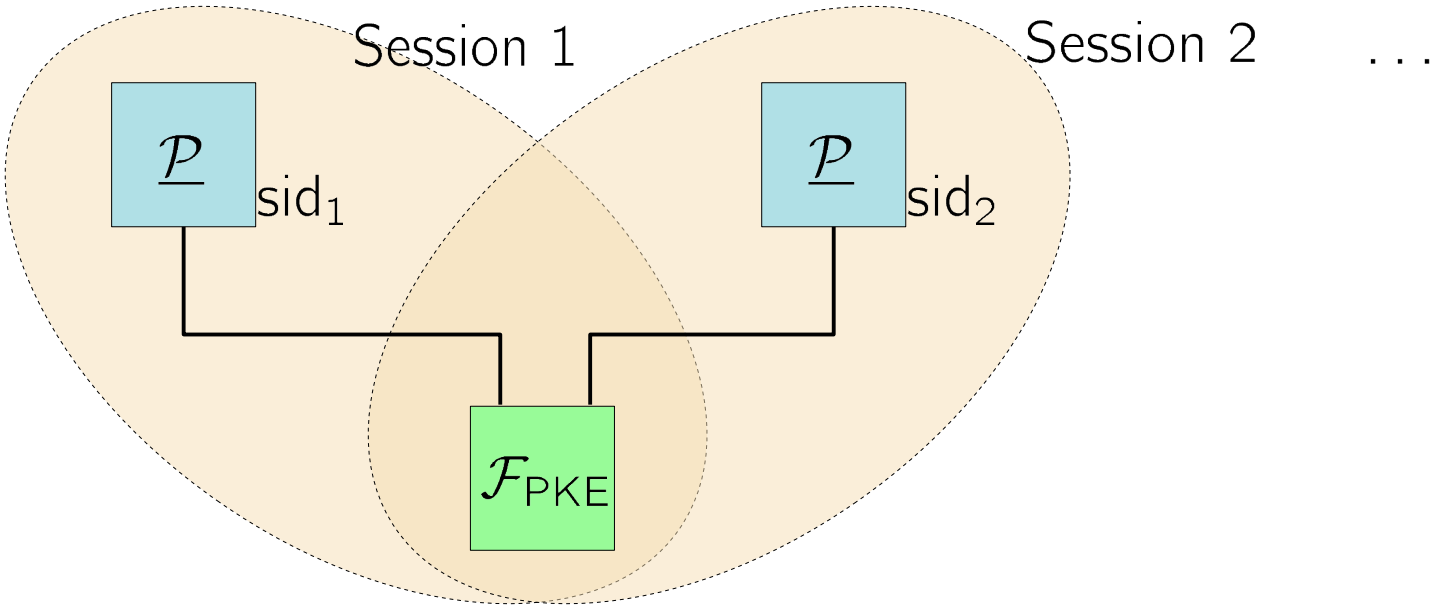


Motivating Joint State

Example: Protocol $\mathcal{P}$ that uses public key encryption (PKE)

- Replace use of PKE by calls to ideal functionality $\mathcal{F}_{\text{PKE}}$
  - Simplifies security analysis
  - Use composition theorem to obtain security of $\mathcal{P}$
- Analyze only single instance (session) of $\mathcal{P}$
  - Security of multiple sessions ($!\underline{\mathcal{P}}$) by composition theorem

$!\underline{\mathcal{P}} \,\|\, \mathcal{F}_{\text{PKE}}$



cannot simply use one instance of $\mathcal{F}_{\text{PKE}}$ in multiple sessions

(General) Joint State Theorem – UC Model

Joint State Theorem in UC model [Canetti and Rabin 2003]

$\mathcal{F}$ ideal functionality

$\mathcal{Q}$ protocol that uses multiple instances of $\mathcal{F}$

$\widehat{\mathcal{P}}$ realization of $\widehat{\mathcal{F}} \approx !\underline{\mathcal{F}}$ – multi session version

Then $\mathcal{Q}^{[\widehat{\mathcal{P}}]}$ realizes $\mathcal{Q}$ using multiple instances of $\mathcal{F}$

(General) Joint State Theorem – UC Model

Joint State Theorem in UC model [Canetti and Rabin 2003]

$\mathcal{F}$ ideal functionality

$\mathcal{Q}$ protocol that uses multiple instances of $\mathcal{F}$

$\widehat{\mathcal{P}}$ realization of $\widehat{\mathcal{F}} \approx !\underline{\mathcal{F}}$ – multi session version

Then $\mathcal{Q}^{[\widehat{\mathcal{P}}]}$ realizes $\mathcal{Q}$ using multiple instances of $\mathcal{F}$

- Good conceptual idea,
  but technical subtleties and limitations of model:
  - JUC operation ($\mathcal{Q}^{[\widehat{\mathcal{P}}]}$) needs to be defined explicitely
  - $!\underline{\mathcal{F}}$ cannot be stated directly in UC model ($\widehat{\mathcal{F}}$ is single ITM)
  - $\widehat{\mathcal{F}}$ is only approximation of $!\underline{\mathcal{F}}$ (exhaution of ITMs)
    Difficult (sometimes impossible) to realize $\widehat{\mathcal{F}}$

(General) Joint State Theorem – IITM Model

Joint State Theorem in IITM model

$\mathcal{F}$ ideal functionality

$\mathcal{Q}$ protocol that connects to $!\underline{\mathcal{F}}$ (multi session)

$\widehat{\mathcal{P}} \quad \leq^{\mathsf{SS}} \quad !\underline{\mathcal{F}}$

Then $\quad \mathcal{Q} \,\|\, \widehat{\mathcal{P}} \quad \leq^{\mathsf{SS}} \quad \mathcal{Q} \,\|\, !\underline{\mathcal{F}}$

(General) Joint State Theorem – IITM Model

Joint State Theorem in IITM model

$\mathcal{F}$ ideal functionality

$\mathcal{Q}$ protocol that connects to $!\underline{\mathcal{F}}$ (multi session)

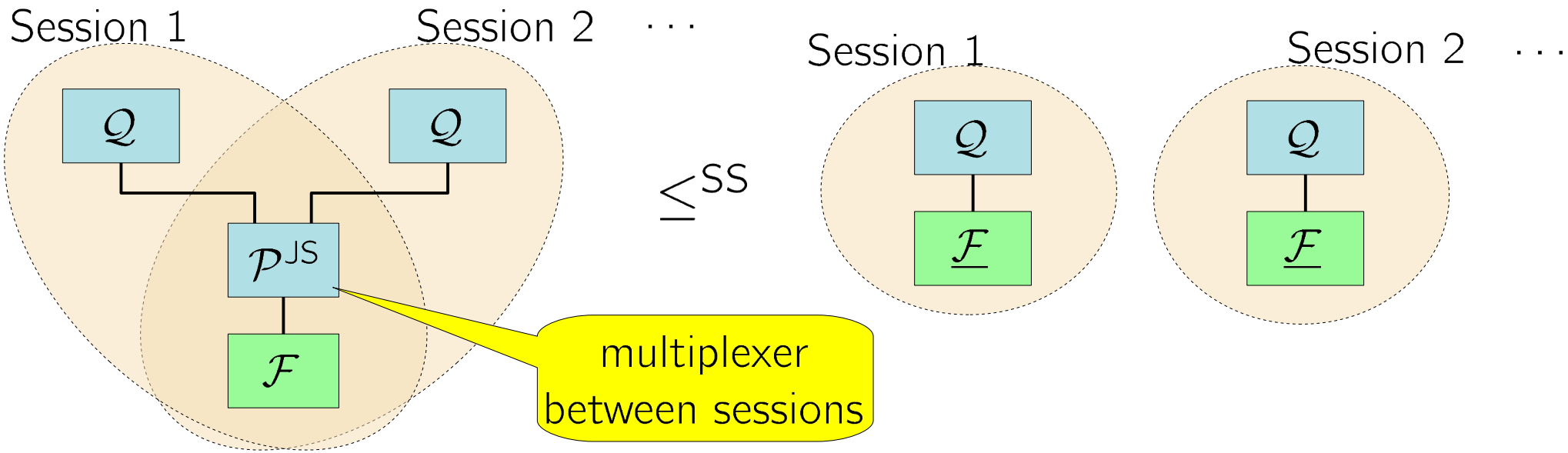
$\widehat{\mathcal{P}} \quad \leq^{\mathsf{SS}} \quad !\underline{\mathcal{F}}$

Then $\quad \mathcal{Q} \,\|\, \widehat{\mathcal{P}} \quad \leq^{\mathsf{SS}} \quad \mathcal{Q} \,\|\, !\underline{\mathcal{F}}$

- Proof: $\mathcal{Q} \leq^{\mathsf{SS}} \mathcal{Q},\ \widehat{\mathcal{P}} \leq^{\mathsf{SS}} !\underline{\mathcal{F}} \overset{\text{comp. thm.}}{\Longrightarrow} \mathcal{Q} \,\|\, \widehat{\mathcal{P}} \leq^{\mathsf{SS}} \mathcal{Q} \,\|\, !\underline{\mathcal{F}}$
- No new composition operation (like $\mathcal{Q}^{[\widehat{\mathcal{P}}]}$) needed
- Elegant and rigorous formulation

- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$

- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathrm{JS}}$ such that $\mathcal{P}^{\mathrm{JS}} \,\|\, \mathcal{F} \leq^{\mathrm{SS}} !\underline{\mathcal{F}}$ and $\mathcal{P}^{\mathrm{JS}}$ is "sufficiently simple"

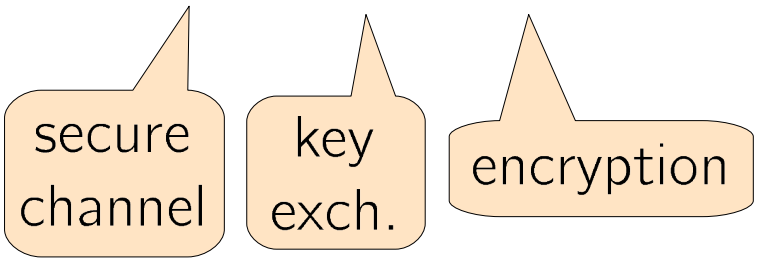


- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathsf{JS}}$ such that $\mathcal{P}^{\mathsf{JS}} \,\|\, \mathcal{F} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  and $\mathcal{P}^{\mathsf{JS}}$ is "sufficiently simple"
- $\mathcal{P}^{\mathsf{JS}}$ is called **joint state realization** for $\mathcal{F}$

- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathsf{JS}}$ such that $\mathcal{P}^{\mathsf{JS}} \,\|\, \mathcal{F} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  and $\mathcal{P}^{\mathsf{JS}}$ is "sufficiently simple"
- $\mathcal{P}^{\mathsf{JS}}$ is called **joint state realization** for $\mathcal{F}$
- Consequences:
  - Reuse of realization: $\mathcal{P} \leq^{\mathsf{SS}} \mathcal{F} \overset{\text{comp. thm.}}{\Longrightarrow} \mathcal{P}^{\mathsf{JS}} \,\|\, \mathcal{P} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$

- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathsf{JS}}$ such that $\mathcal{P}^{\mathsf{JS}} \,\|\, \mathcal{F} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  and $\mathcal{P}^{\mathsf{JS}}$ is "sufficiently simple"
- $\mathcal{P}^{\mathsf{JS}}$ is called **joint state realization** for $\mathcal{F}$
- Consequences:
  - Reuse of realization: $\mathcal{P} \leq^{\mathsf{SS}} \mathcal{F} \overset{\text{comp. thm.}}{\Longrightarrow} \mathcal{P}^{\mathsf{JS}} \,\|\, \mathcal{P} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  - Iterative application:

$$\mathcal{Q} \,\|\, !\underline{(\mathcal{Q}' \,\|\, !\underline{\mathcal{F}})}$$



- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathsf{JS}}$ such that $\mathcal{P}^{\mathsf{JS}} \| \mathcal{F} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$ and $\mathcal{P}^{\mathsf{JS}}$ is "sufficiently simple"
- $\mathcal{P}^{\mathsf{JS}}$ is called **joint state realization** for $\mathcal{F}$
- Consequences:
  - Reuse of realization: $\mathcal{P} \leq^{\mathsf{SS}} \mathcal{F} \overset{\text{comp. thm.}}{\Longrightarrow} \mathcal{P}^{\mathsf{JS}} \| \mathcal{P} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  - Iterative application:

$$\mathcal{Q} \| !\underline{(\mathcal{Q}' \| !\underline{\mathcal{F}})} \geq^{\mathsf{SS}} \mathcal{Q} \| !\underline{(\mathcal{Q}' \| \mathcal{P}^{\mathsf{JS}} \| \mathcal{F})} \overset{\text{by def.}}{=} \mathcal{Q} \| !\underline{\mathcal{Q}'} \| !\underline{\mathcal{P}^{\mathsf{JS}}} \| !\underline{\mathcal{F}}$$

JS + comp. thm.

secure channel

key exch.

encryption

- Joint State theorem itself does not yield practical realization
- $\widehat{\mathcal{P}}$ not necessarily better than $!\underline{\mathcal{P}}$
- Solution: Find $\mathcal{P}^{\mathsf{JS}}$ such that $\mathcal{P}^{\mathsf{JS}} \| \mathcal{F} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  and $\mathcal{P}^{\mathsf{JS}}$ is "sufficiently simple"
- $\mathcal{P}^{\mathsf{JS}}$ is called **joint state realization** for $\mathcal{F}$
- Consequences:
  - Reuse of realization: $\mathcal{P} \leq^{\mathsf{SS}} \mathcal{F} \overset{\text{comp. thm.}}{\Longrightarrow} \mathcal{P}^{\mathsf{JS}} \| \mathcal{P} \leq^{\mathsf{SS}} !\underline{\mathcal{F}}$
  - Iterative application:

$$\mathcal{Q} \| !\underline{(\mathcal{Q}' \| !\underline{\mathcal{F}})} \geq^{\mathsf{SS}} \mathcal{Q} \| !\underline{(\mathcal{Q}' \| \mathcal{P}^{\mathsf{JS}} \| \mathcal{F})} \overset{\text{by def.}}{=} \mathcal{Q} \| !\underline{\mathcal{Q}'} \| !\underline{\mathcal{P}^{\mathsf{JS}}} \| !\underline{\mathcal{F}}$$

$$\geq^{\mathsf{SS}} \mathcal{Q} \| !\underline{\mathcal{Q}'} \| \underbrace{!\underline{\mathcal{P}^{\mathsf{JS}}} \| \mathcal{P}^{\mathsf{JS}}}_{\text{joint state realization}} \| \mathcal{F}$$

JS + comp. thm. (secure channel, key exch.) — JS + comp. thm. (encryption)

Application of JS Theorem to PKE

Ideal Functionality for PKE: $\mathcal{F}_{\mathsf{PKE}}(l)$

$l : \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0 : m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

Ideal Functionality for PKE: $\mathcal{F}_{\mathsf{PKE}}(l)$

$l\colon \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0\colon m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  - $c := e(k_e, l(m))$
  - if $d(c) \neq l(m)$ then return **error**
  - else store (m, c), return c
- else
  - return $c := e(k', m)$

Ideal Functionality for PKE: $\mathcal{F}_{\mathsf{PKE}}(l)$

$l\colon \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0\colon m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c

  else
  return $c := e(k', m)$

Decryption:
- recv $(\mathbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**

  else if (m, c) stored then return m
  else return $d(c)$

Ideal Functionality for PKE: $\mathcal{F}_{\mathsf{PKE}}(l)$

$l\colon \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0\colon m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c
  else
  return $c := e(k', m)$

Decryption:
- recv $(\mathbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**
  else if (m, c) stored then return m
  else return $d(c)$

Static Corruption:
- recv $(\mathbf{Corrupt})$ from adversary

Ideal Functionality for PKE: $\mathcal{F}_{\mathrm{PKE}}(l)$

$l\colon \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0\colon m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c

  else
  return $c := e(k', m)$

Decryption:
- recv $(\mathbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**

  else if (m, c) stored then return m
  else return $d(c)$

Static Corruption:
- recv $(\mathbf{Corrupt})$ from adversary

- Non-interactive formulation, Advantage: Nested encryption

Ideal Functionality for PKE: $\mathcal{F}_{\mathrm{PKE}}(l)$

$l\colon \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0\colon m \mapsto 0^{|m|}$

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

no restrictions to e, d, k_e
$\Rightarrow$ quantification over all alg.
$\Rightarrow$ easier proofs/application

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c
  else
  return $c := e(k', m)$

Decryption:
- recv $(\mathbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**
  else if (m, c) stored then return m
  else return $d(c)$

Static Corruption:
- recv $(\mathbf{Corrupt})$ from adversary

- Non-interactive formulation, Advantage: Nested encryption

Ideal Functionality for PKE: $\mathcal{F}_{\text{PKE}}$

$l : \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0 : m \mapsto 0^{|m|}$

identifying k_e with e does not work for JS

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

no restrictions to e, d, k_e
$\Rightarrow$ quantification over all alg.
$\Rightarrow$ easier proofs/application

Encryption:
- recv $(\textbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c
  else
  return $c := e(k', m)$

Decryption:
- recv $(\textbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**
  else if (m, c) stored then return m
  else return $d(c)$

Static Corruption:
- recv $(\textbf{Corrupt})$ from adversary

- Non-interactive formulation, Advantage: Nested encryption

Ideal Functionality for PKE: $\mathcal{F}_{\mathrm{PKE}}$

$l : \{0,1\}^* \to \{0,1\}^*$ models leakage, e.g., $l_0 : m \mapsto 0^{|m|}$

identifying k_e with e does not work for JS

Key Generation:
- recv **KeyGen** from decryptor, send **KeyGen** to adversary
- recv (e, d, k_e) from adv., store e, d, k_e
- send k_e to decryptor

no restrictions to e, d, k_e
$\Rightarrow$ quantification over all alg.
$\Rightarrow$ easier proofs/application

Encryption:
- recv $(\mathbf{Enc}, k', m)$ from encryptor
- if $k_e = k'$ and not corrupted then
  $c := e(k_e, l(m))$
  if $d(c) \neq l(m)$ then return **error**
  else store (m, c), return c
  else
  return $c := e(k', m)$

important for JS, missing in other formulations

Decryption:
- recv $(\mathbf{Dec}, c)$ from decryptor
- if $(m, c), (m', c)$ stored, $m \neq m'$ then return **error**
  else if (m, c) stored then return m
  else return $d(c)$

Static Corruption:
- recv $(\mathbf{Corrupt})$ from adversary

- Non-interactive formulation, Advantage: Nested encryption

Realizing $\mathcal{F}_{\mathsf{PKE}}$ by Encryption Scheme

- $\Sigma = (\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ – public-key encryption scheme
- $\mathcal{P}_\Sigma$ describes Σ as protocol in IITM model

Theorem

$$\Sigma \text{ is CCA-secure} \implies \mathcal{P}_\Sigma \leq^{\mathsf{SS}} \mathcal{F}_{\mathsf{PKE}}(l)$$

for all leakage l with $|l(m)| = |m|$

e.g. leakage $l_0 \colon m \mapsto 0^{|m|}$

Joint State Realization for PKE

- Basic idea [Canetti and Herzog 2006]:
  - Encrypt $\langle \text{sid}, m \rangle$ instead of m
  - Upon decryption check if plaintext is of shape $\langle \text{sid}, m \rangle$ (else error)

Joint State Realization for PKE

- Basic idea [Canetti and Herzog 2006]:
  - Encrypt $\langle \mathsf{sid}, m \rangle$ instead of m
  - Upon decryption check if plaintext is of shape $\langle \mathsf{sid}, m \rangle$ (else error)

Joint State Theorem for PKE

$$\mathcal{P}^{\mathsf{JS}}_{\mathsf{PKE}} \,\|\, \mathcal{F}_{\mathsf{PKE}}(l') \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{PKE}}(l)}$$

where $l'(\langle \mathsf{sid}, m \rangle) = \langle \mathsf{sid}, l(m) \rangle$ (leakage of SID)

Joint State Realization for PKE

- Basic idea [Canetti and Herzog 2006]:
  - Encrypt $\langle \mathsf{sid}, m \rangle$ instead of m
  - Upon decryption check if plaintext is of shape $\langle \mathsf{sid}, m \rangle$ (else error)

Joint State Theorem for PKE

$$\mathcal{P}_{\mathsf{PKE}}^{\mathsf{JS}} \,\|\, \mathcal{F}_{\mathsf{PKE}}(l') \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{PKE}}(l)}$$

$$\text{where } l'(\langle \mathsf{sid}, m \rangle) = \langle \mathsf{sid}, l(m) \rangle \quad \text{(leakage of SID)}$$

- Proof employs leakage and decryption test
- Recall: Σ CCA-secure $\Rightarrow$ $\mathcal{P}_\Sigma \;\leq^{\mathsf{SS}}\; \mathcal{F}_{\mathsf{PKE}}(l'_0)$

$$\Longrightarrow \;\mathcal{P}_{\mathsf{PKE}}^{\mathsf{JS}} \,\|\, \mathcal{P}_\Sigma \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{PKE}}(l_0)}$$

More Results

Joint State Theorem for Replayable PKE

$$\mathcal{P}^{\mathsf{JS}}_{\mathsf{PKE}} \,|\, \mathcal{F}_{\mathsf{RPKE}}(I') \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{RPKE}}(I)}$$

where $I'(\langle \mathsf{sid}, m \rangle) = \langle \mathsf{sid}, I(m) \rangle$ (leakage of SID)

- $\mathcal{F}_{\mathsf{RPKE}}$ **replayable PKE** functionality (relaxation of $\mathcal{F}_{\mathsf{PKE}}$)
  (weakens non-malleability guarantee)
- Related to RCCA [Canetti, Krawczyk and Nielsen 2003]

More Results

Joint State Theorem for Replayable PKE

$$\mathcal{P}^{\mathsf{JS}}_{\mathsf{PKE}} \,|\, \mathcal{F}_{\mathsf{RPKE}}(I') \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{RPKE}}(I)}$$

where $I'(\langle \mathsf{sid}, m\rangle) = \langle \mathsf{sid}, I(m)\rangle$ (leakage of SID)

- $\mathcal{F}_{\mathsf{RPKE}}$ **replayable PKE** functionality (relaxation of $\mathcal{F}_{\mathsf{PKE}}$)
  (weakens non-malleability guarantee)
- Related to RCCA [Canetti, Krawczyk and Nielsen 2003]

Joint State Theorem for Digital Signatures

$$\mathcal{P}^{\mathsf{JS}}_{\mathsf{SIG}} \,|\, \mathcal{F}_{\mathsf{SIG}} \;\leq^{\mathsf{SS}}\; !\underline{\mathcal{F}_{\mathsf{SIG}}}$$

- $\mathcal{F}_{\mathsf{SIG}}$ non-interactive signature functionality

Related Work

- **[Pfitzmann, Waidner '01]**:
  Non-interactive parameterized $\mathcal{F}_{\mathsf{PKE}}$ and $\mathcal{F}_{\mathsf{SIG}}$
  (JS not considered)

- **[Canetti, Rabin '03]**: JS theorem for interactive $\mathcal{F}_{\mathsf{SIG}}$

- **[CKN '03]**: Interactive $\mathcal{F}_{\mathsf{RPKE}}$ (JS not considered)

- **[Canetti '05]**: – Non-interactive $\mathcal{F}_{\mathsf{PKE}}$ and $\mathcal{F}_{\mathsf{SIG}}$
  for JS point to [Canetti, Rabin '03]
  – de facto interactive $\mathcal{F}_{\mathsf{RPKE}}$

- **[Canetti, Herzog '06]**: Basic idea of $\mathcal{P}^{\mathsf{JS}}_{\mathsf{PKE}}$
  no proof, parameterized $\mathcal{F}_{\mathsf{PKE}}$

Conclusion

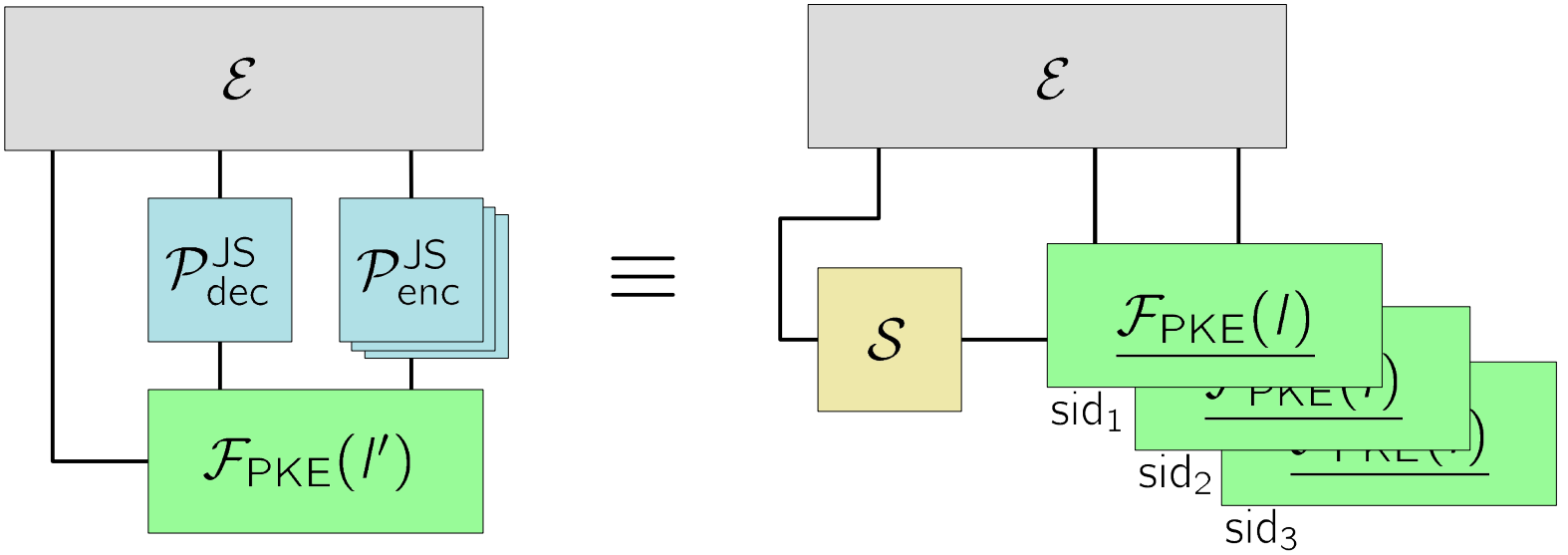
- JS realizations important for practical realizations
- General JS theorem special case of comp. thm. in IITM model
- We presented JS realizations for **non-interactive**
  - **PKE**,
  - **replayable PKE** and
  - **digital signature** functionalities

Thank you for your attention!

Thank you for your attention!

Proof of JS Theorem for PKE:

Define simulator $\mathcal{S}$ s.t. for all $\mathcal{E}$:



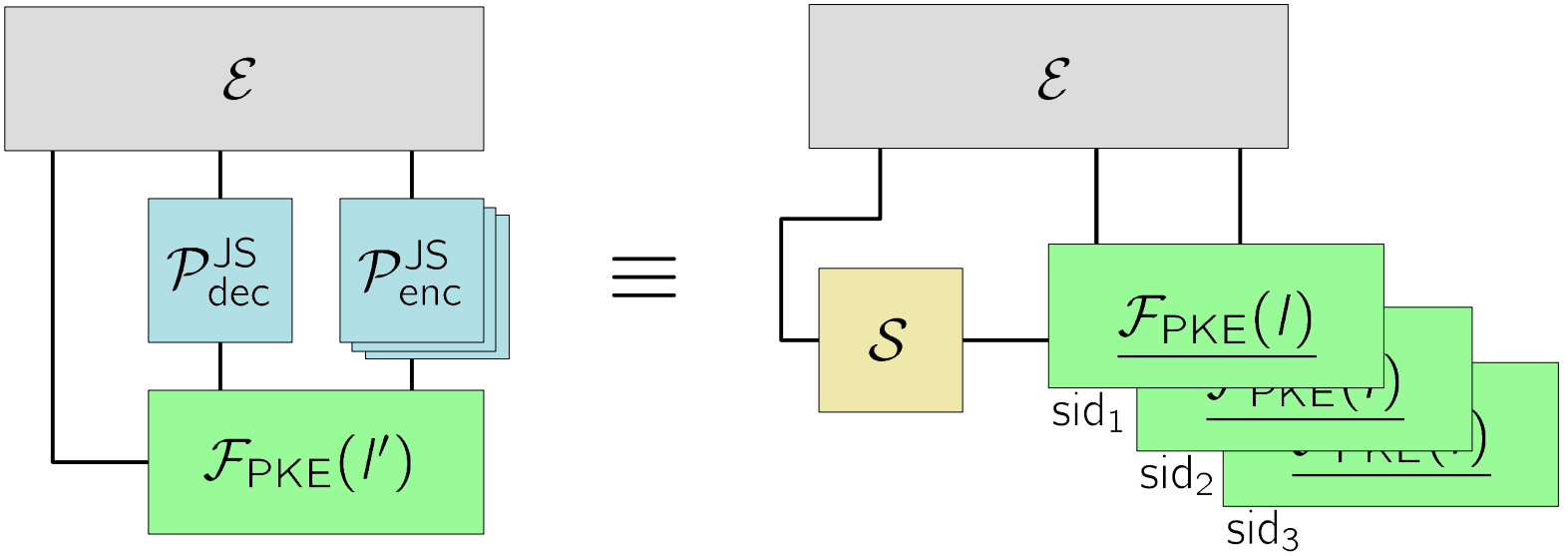
- $\mathcal{S}$ obtains algorithms e, d and key k from $\mathcal{E}$
- $\mathcal{S}$ provides algorithms $e_{\mathsf{sid}}, d_{\mathsf{sid}}$ and key k to $\underline{\mathcal{F}_{\mathsf{PKE}}}[\mathsf{sid}]$

$$e_{\mathsf{sid}}(k, m) = e(k, \langle \mathsf{sid}, m \rangle)$$

$$d_{\mathsf{sid}}(c) = \begin{cases} m & \text{if } d(y) = \langle \mathsf{sid}, x \rangle \\ \perp & \text{otherwise} \end{cases}$$

Proof of JS Theorem for PKE:

Define simulator $\mathcal{S}$ s.t. for all $\mathcal{E}$:



- Main problem: Collisions in JS world
  - two plaintexts from different sessions encrypt to same ciphertext
  - can not occur by **leakage** and **decryption test**

$$\text{dec}(\underbrace{\text{enc}_k(I'(\langle \text{sid}, m\rangle))}_{\text{ciphertext } c}) = I'(\langle \text{sid}, m\rangle) = \langle \text{sid}, I(m)\rangle$$

- Proof holds for unbounded $\mathcal{E} \Rightarrow$ **perfect indistinguishability**