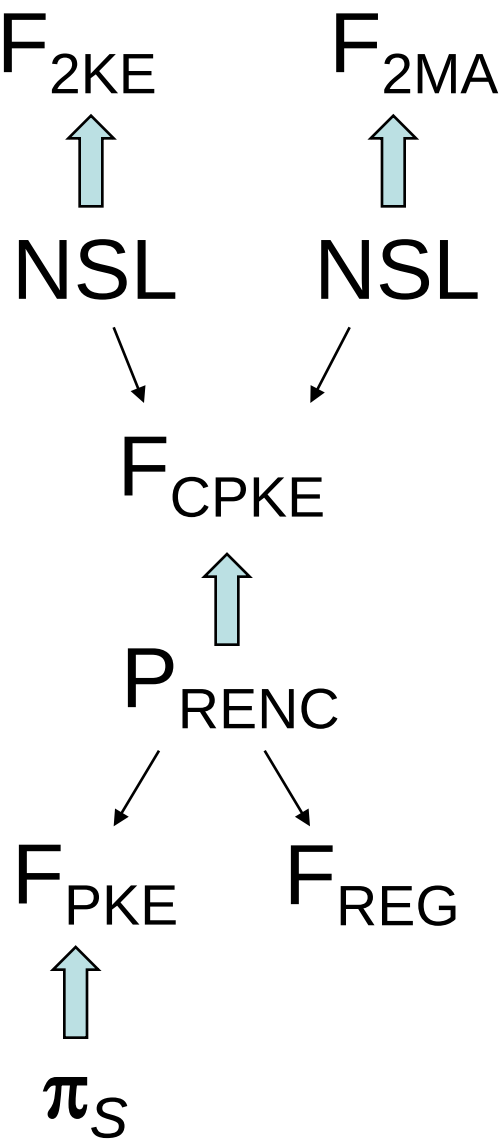
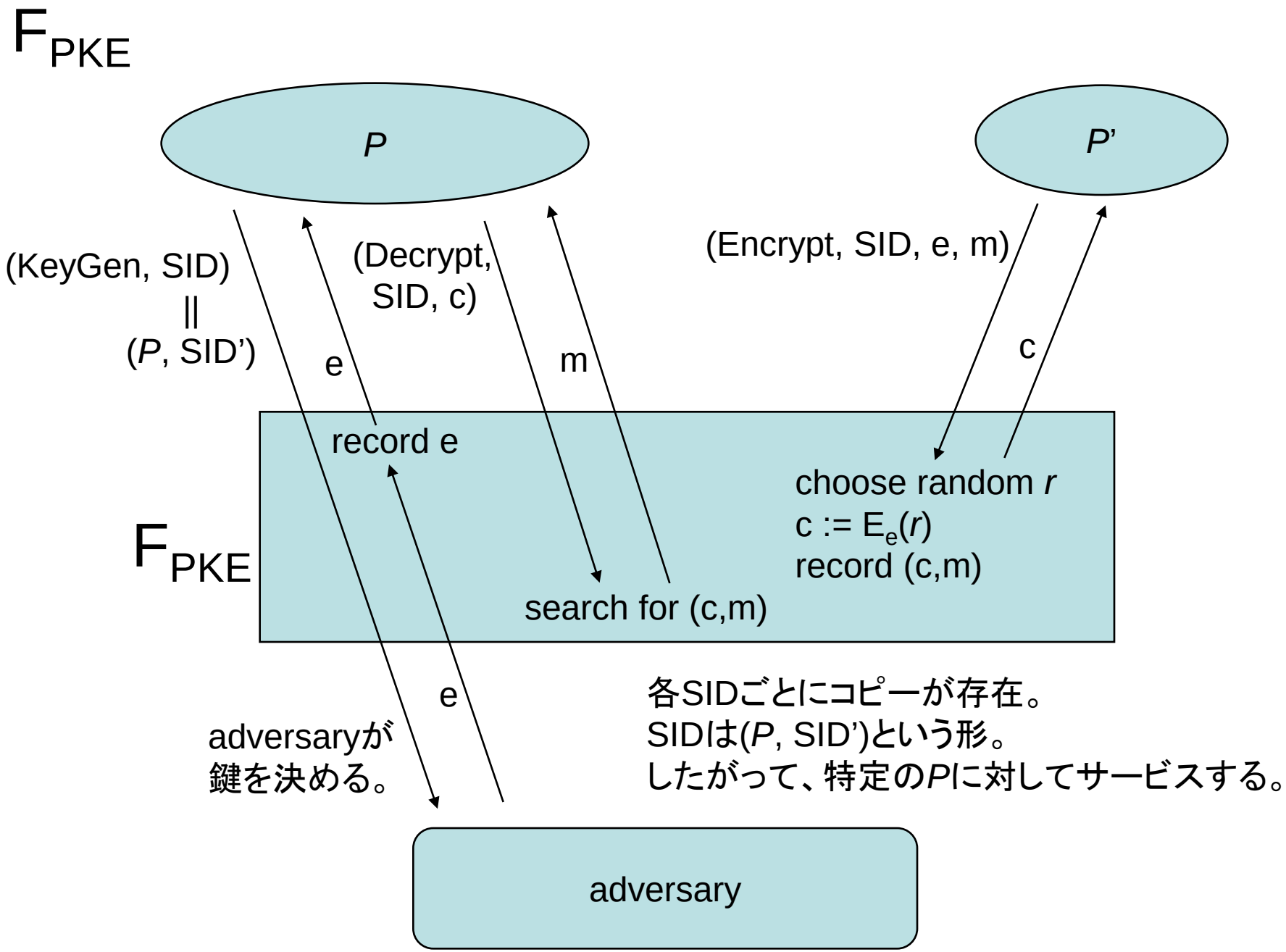
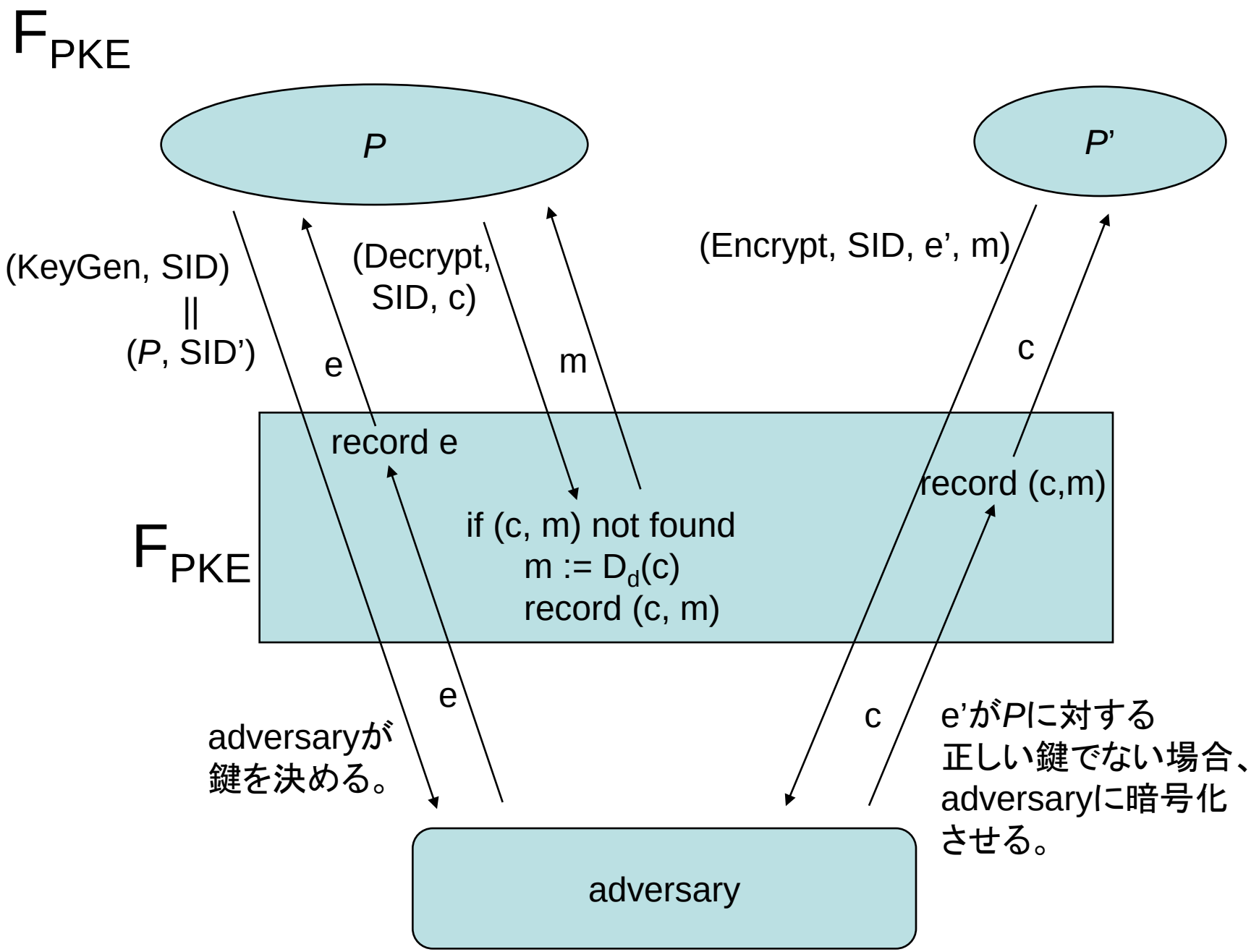
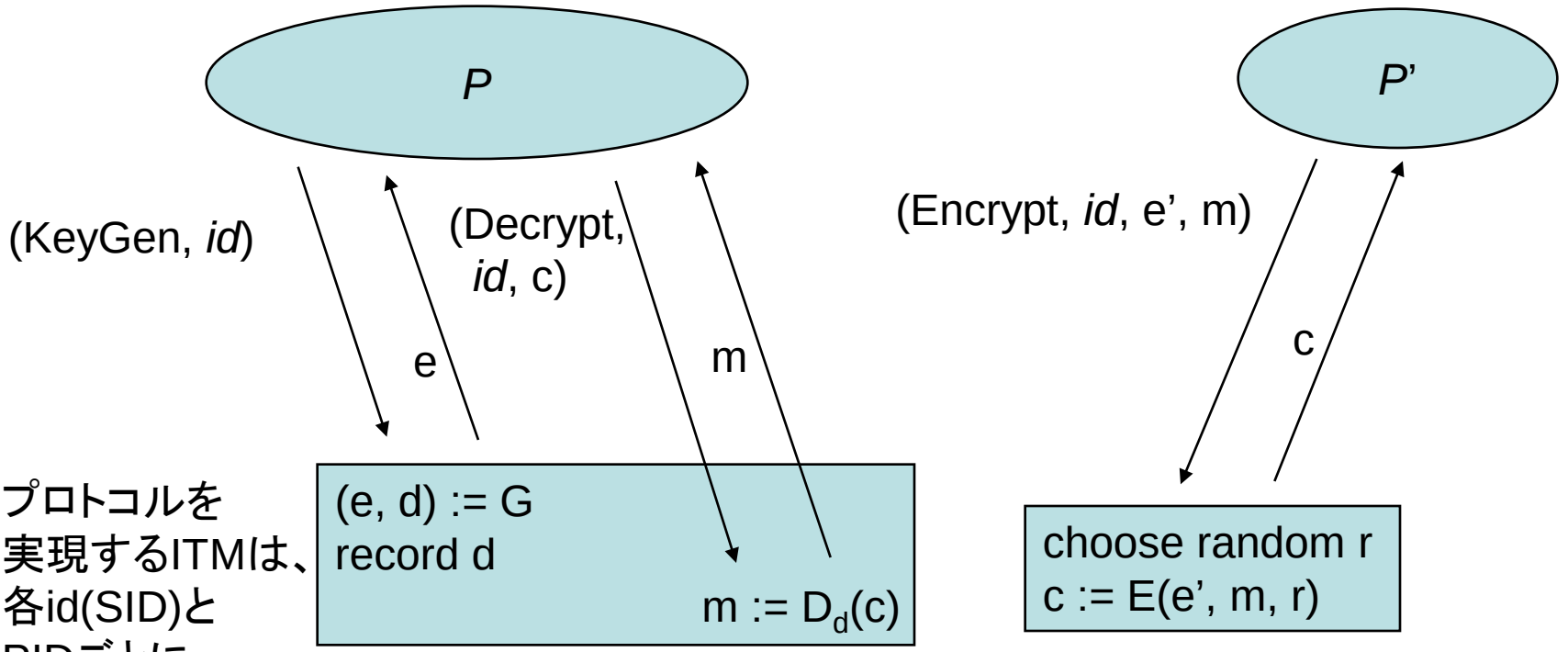

F_{2KE}
F_{2MA}
NSL
NSL
F_{CPKE}
P_{RENC}
F_{PKE}
F_{REG}
π_S


F_{PKE}
P
P'
(KeyGen, SID)
||
(P, SID')
(Decrypt,
SID, c)
(Encrypt, SID, e, m)
e
m
c
record e
choose random r
c := $E_e(r)$
record (c,m)
F_{PKE}
search for (c,m)
e
adversaryが
鍵を決める。
各SIDごとにコピーが存在。
SIDは(P, SID')という形。
したがって、特定のPに対してサービスする。
adversary

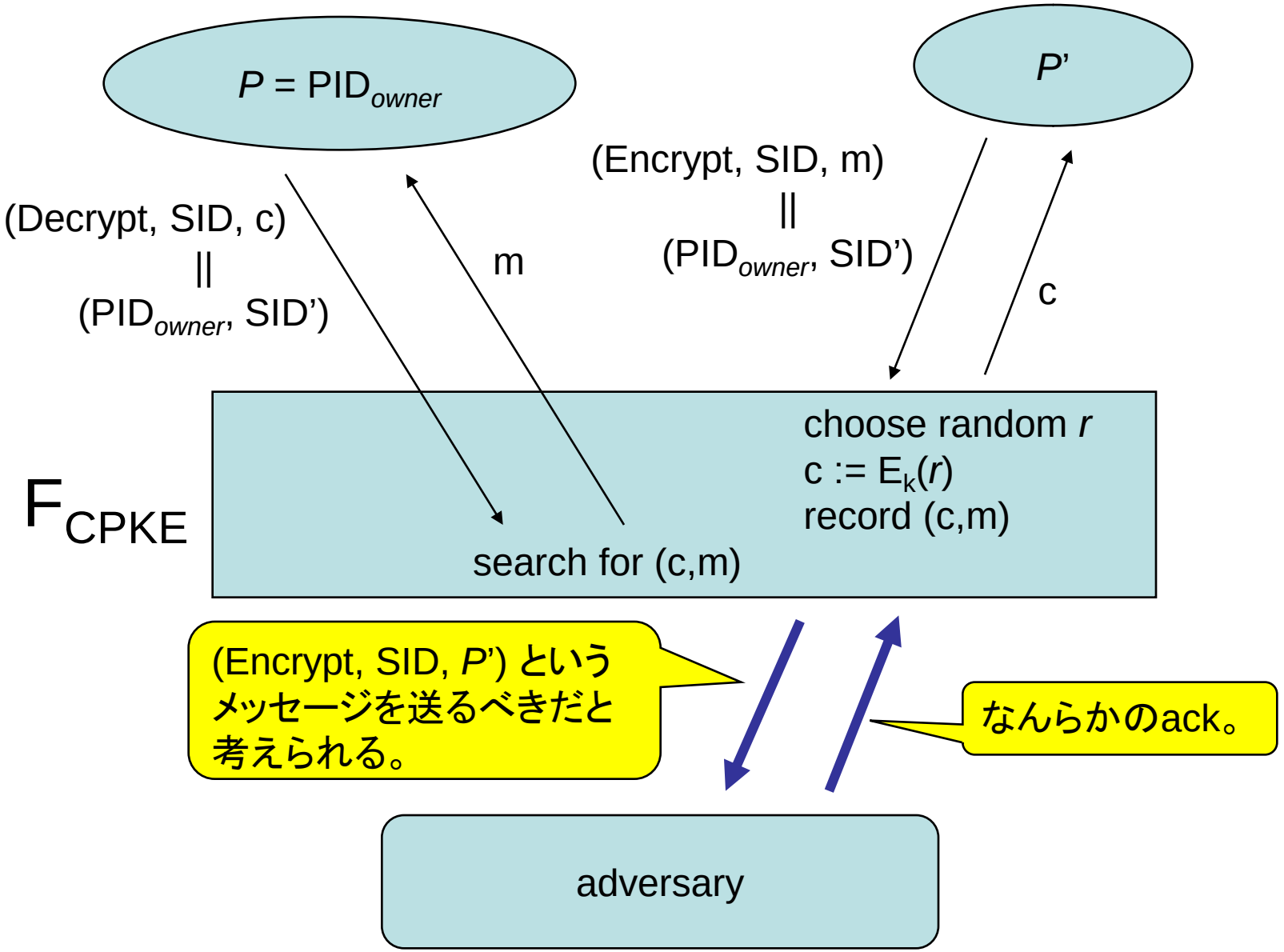

F_{PKE}
P
P'
(KeyGen, SID)
||
(P, SID')
(Decrypt,
SID, c)
(Encrypt, SID, e', m)
e
m
c
record e
record (c,m)
F_{PKE}
if (c, m) not found
m := D_d(c)
record (c, m)
e
c
adversaryが
鍵を決める。
e'がPに対する
正しい鍵でない場合、
adversaryに暗号化
させる。
adversary

π_S

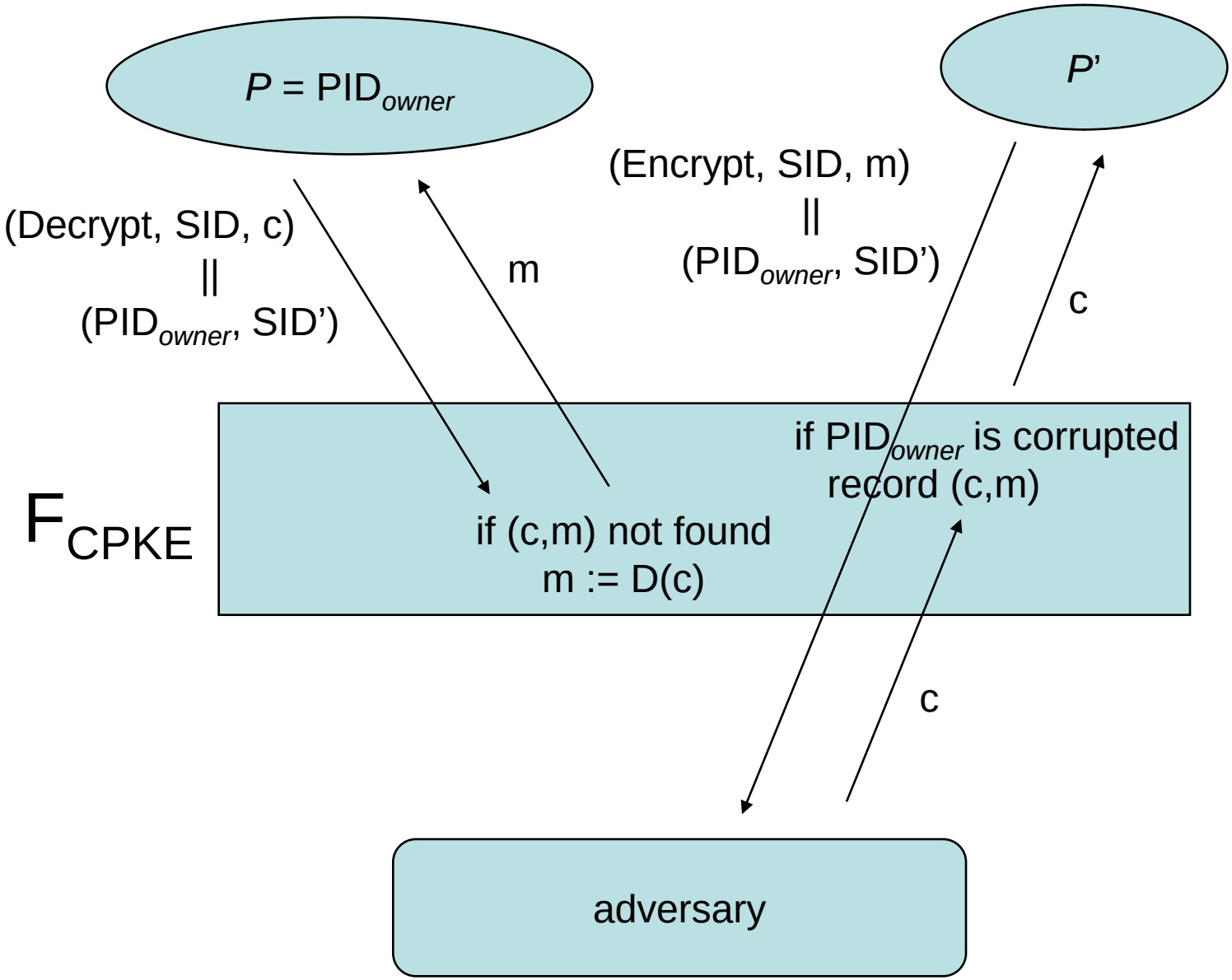


Theorem: *S*=(G, E, D)はIND-CCA2
if and only if π_SはF_{PKE}をsecurely realize

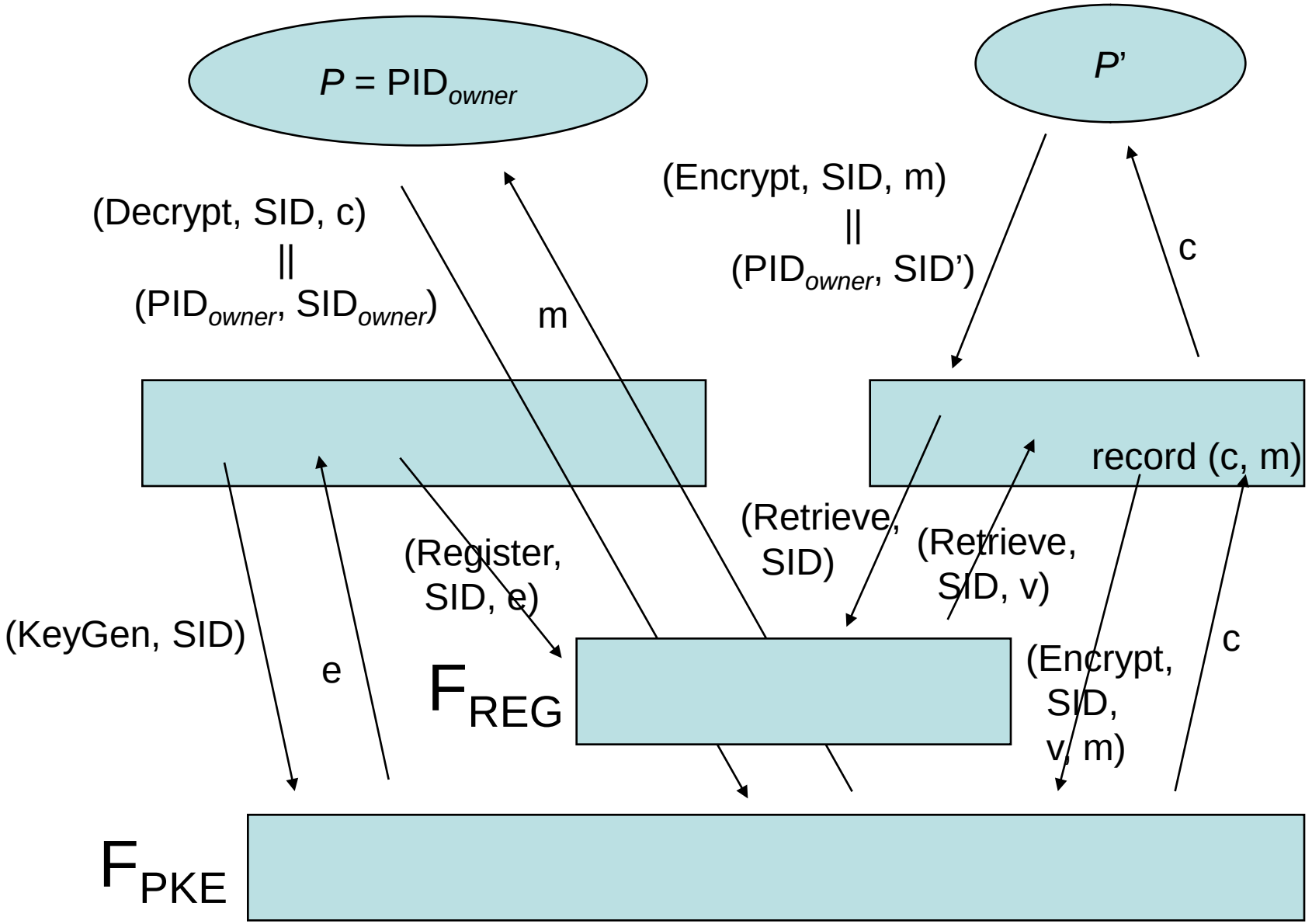
F_{CPKE}


P = PID_owner
P'
(Encrypt, SID, m)
||
(PID_owner, SID')
(Decrypt, SID, c)
||
(PID_owner, SID')
m
c
F_CPKE
choose random r
c := E_k(r)
record (c,m)
search for (c,m)
(Encrypt, SID, P') というメッセージを送るべきだと考えられる。
なんらかのack。
adversary

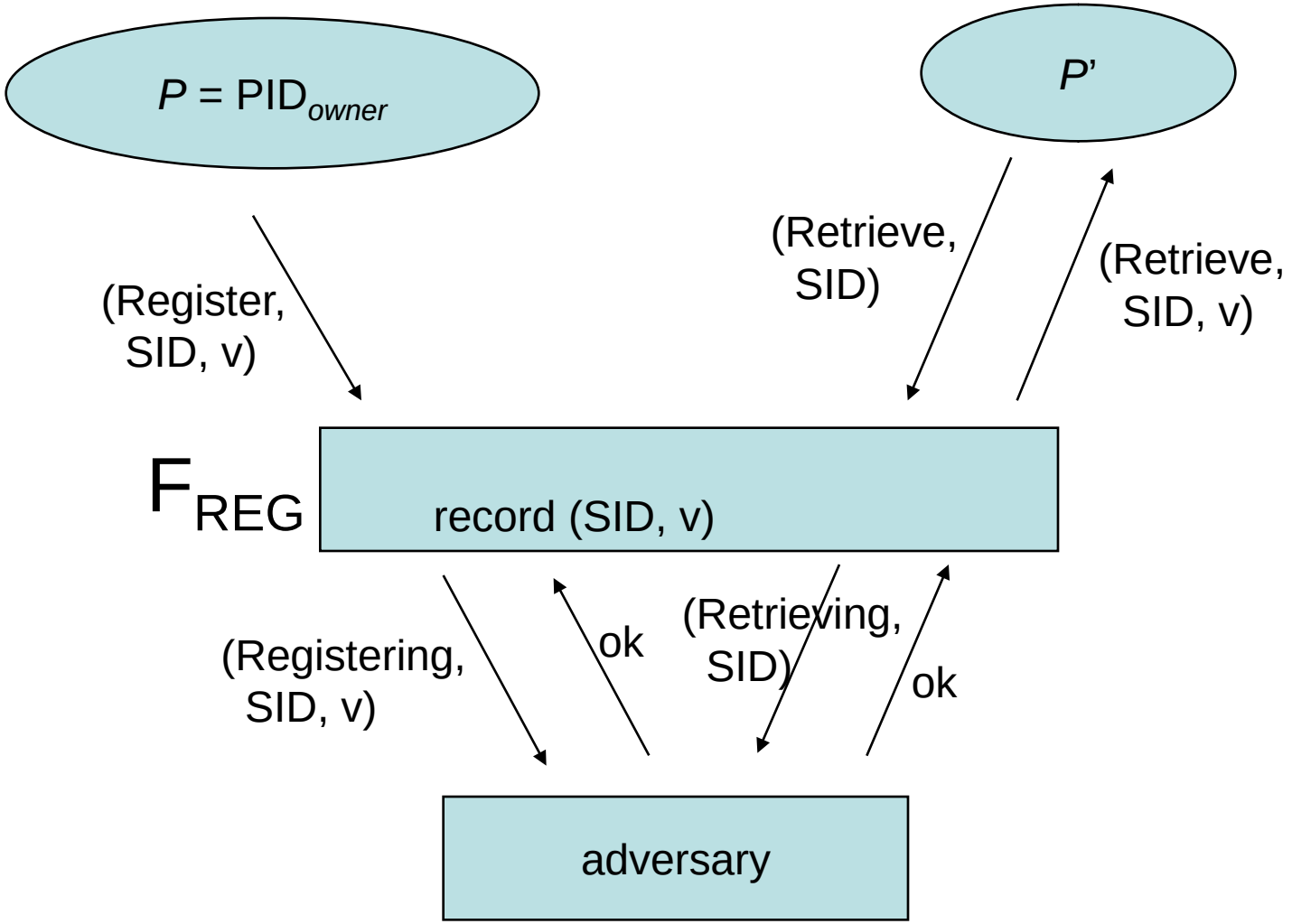
F_{CPKE}

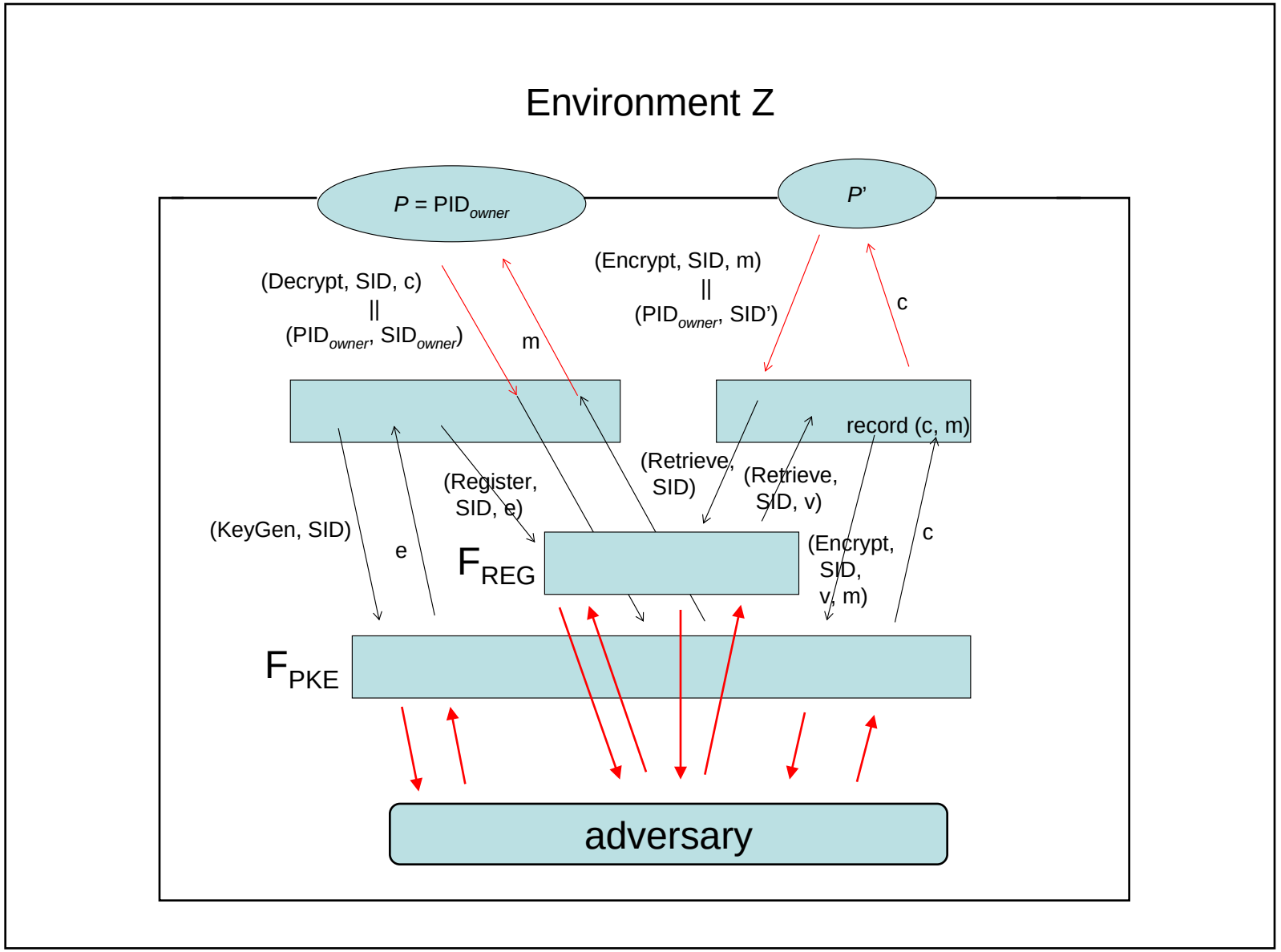

P = PID_owner
P'
(Encrypt, SID, m)
||
(PID_owner, SID')
(Decrypt, SID, c)
||
(PID_owner, SID')
m
c
F_CPKE
if PID_owner is corrupted
record (c,m)
if (c,m) not found
m := D(c)
c
adversary

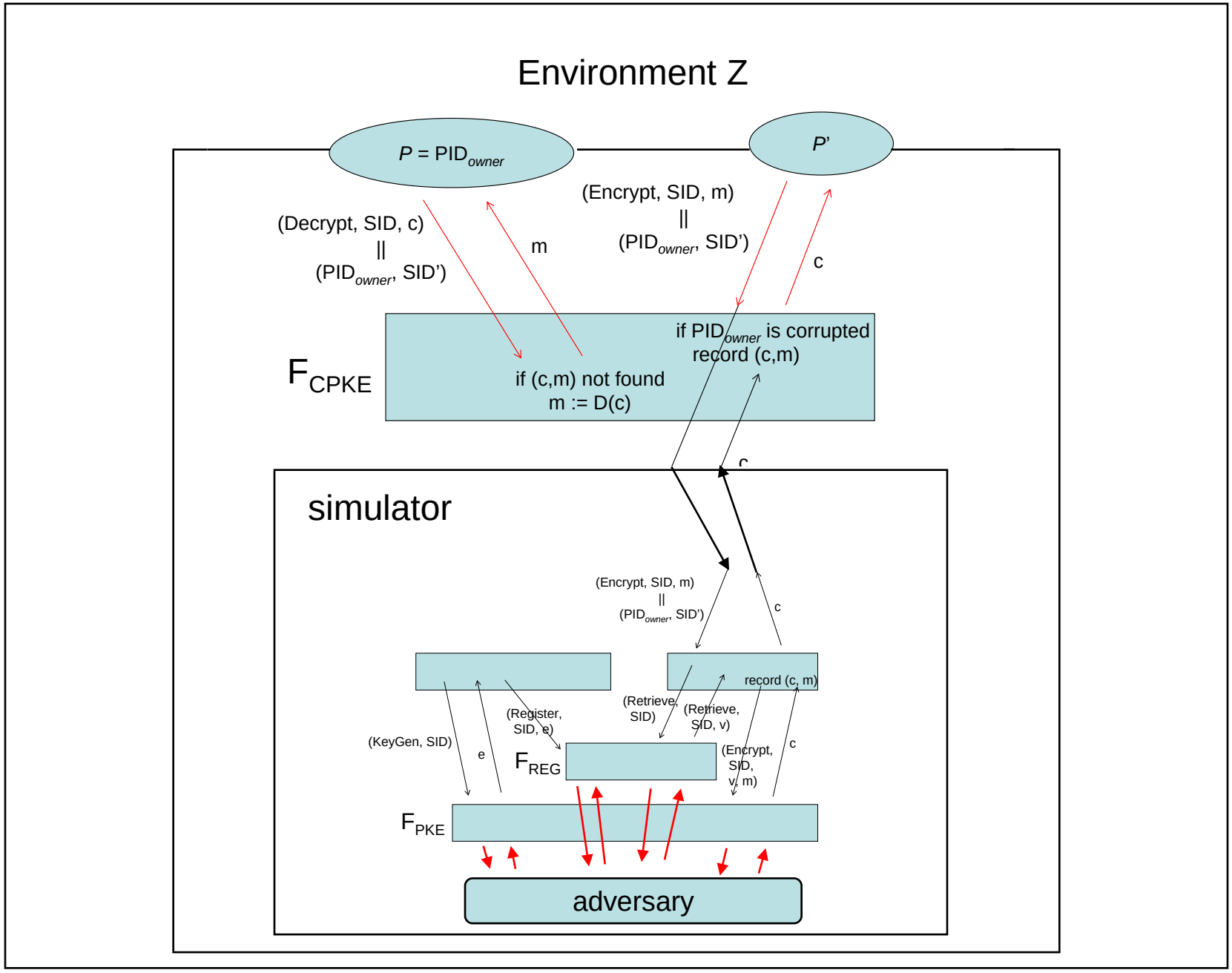
P_{RENC}

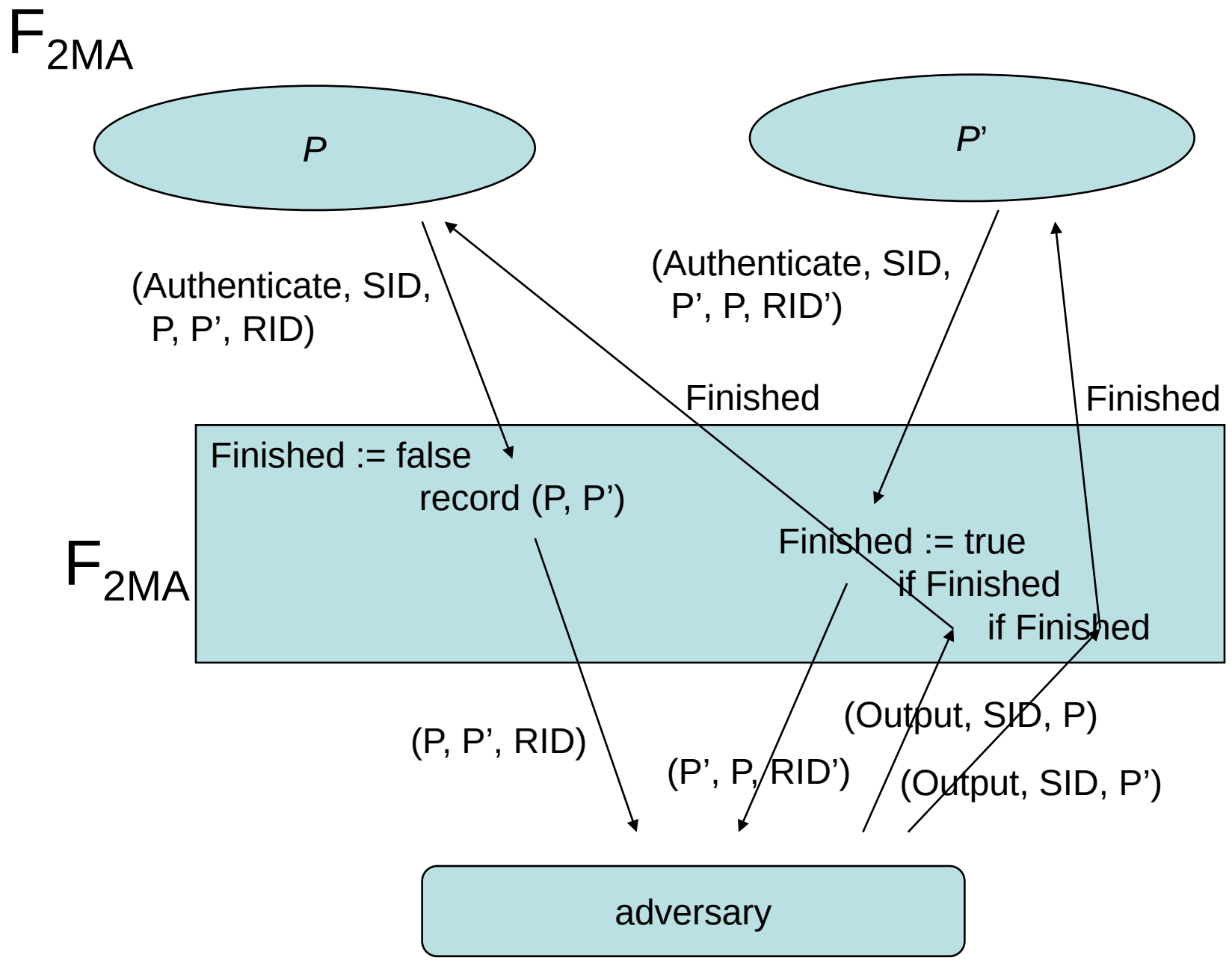

P = PID_owner
P'
(Decrypt, SID, c)
||
(PID_owner, SID_owner)
m
(Encrypt, SID, m)
||
(PID_owner, SID')
c
record (c, m)
(Retrieve, SID)
(Retrieve, SID, v)
(Register, SID, e)
(KeyGen, SID)
e
F_REG
(Encrypt, SID, v, m)
c
F_PKE

F_{REG}

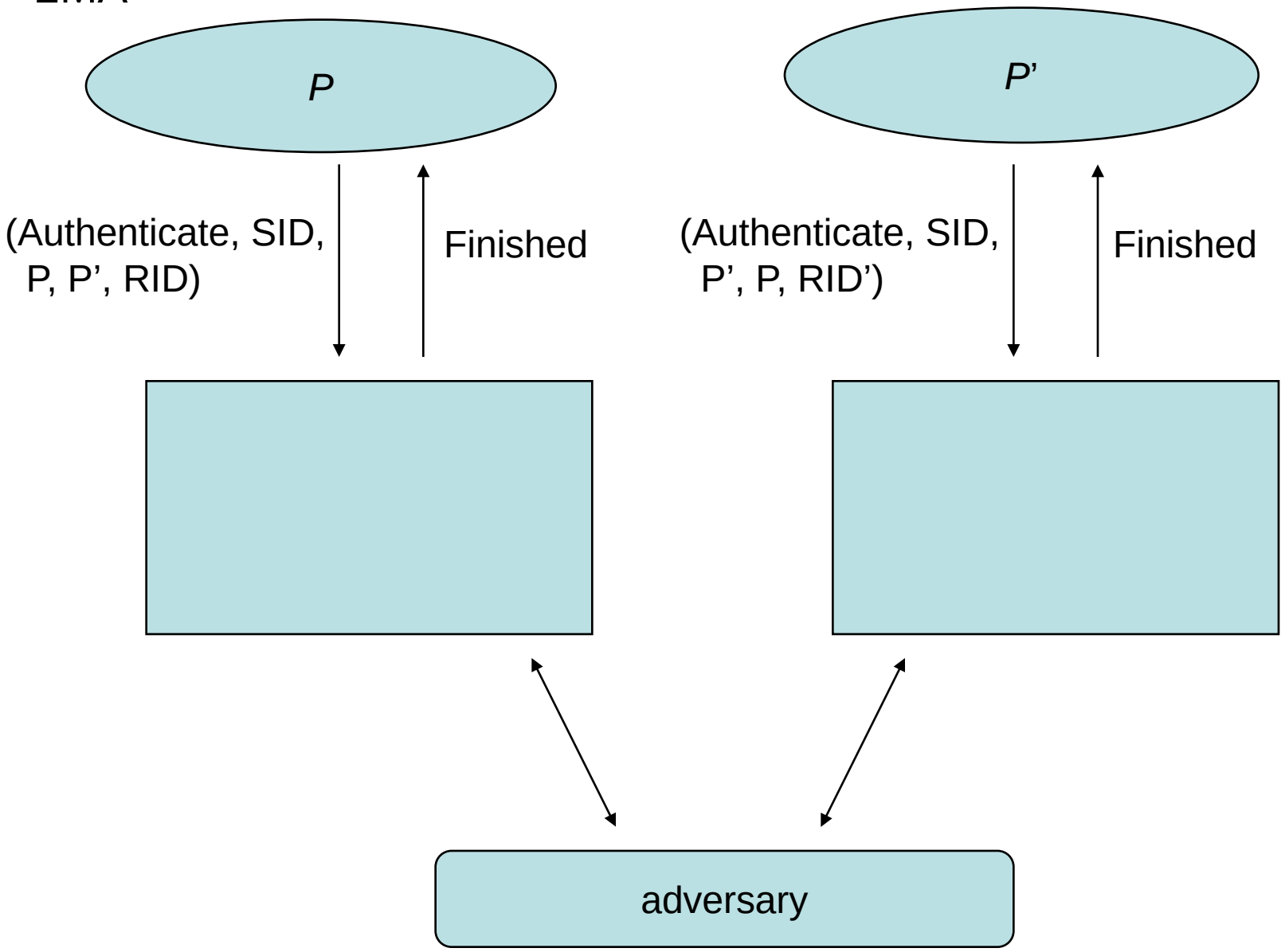

P = PID_owner
P'
(Register, SID, v)
(Retrieve, SID)
(Retrieve, SID, v)
F_REG
record (SID, v)
(Registering, SID, v)
ok
(Retrieving, SID)
ok
adversary

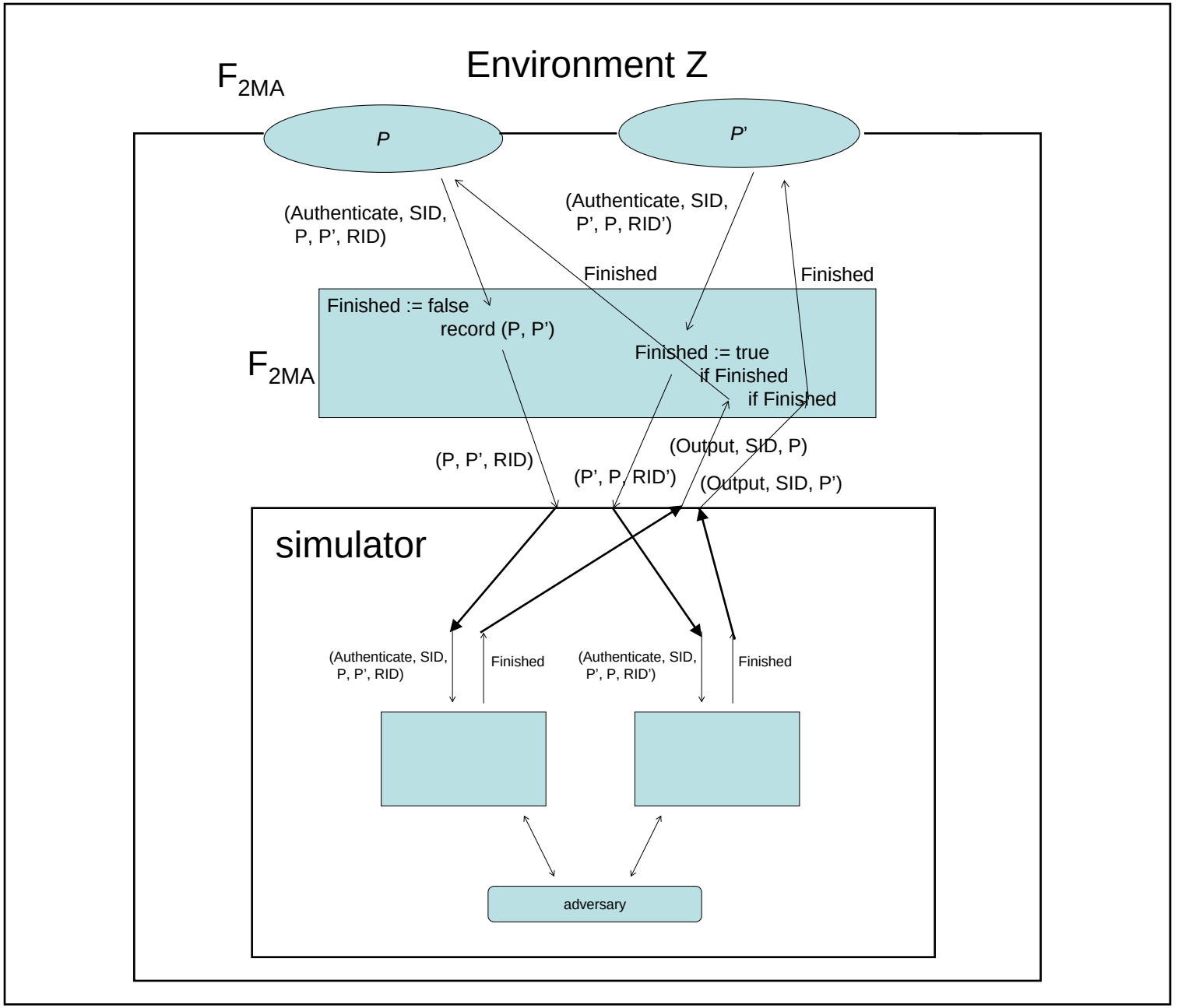

Environment Z
P = PID_owner
P'
(Decrypt, SID, c)
||
(PID_owner, SID_owner)
m
(Encrypt, SID, m)
||
(PID_owner, SID')
c
record (c, m)
(Retrieve, SID)
(Retrieve, SID, v)
(Register, SID, e)
(KeyGen, SID)
e
F_REG
(Encrypt, SID, v, m)
c
F_PKE
adversary

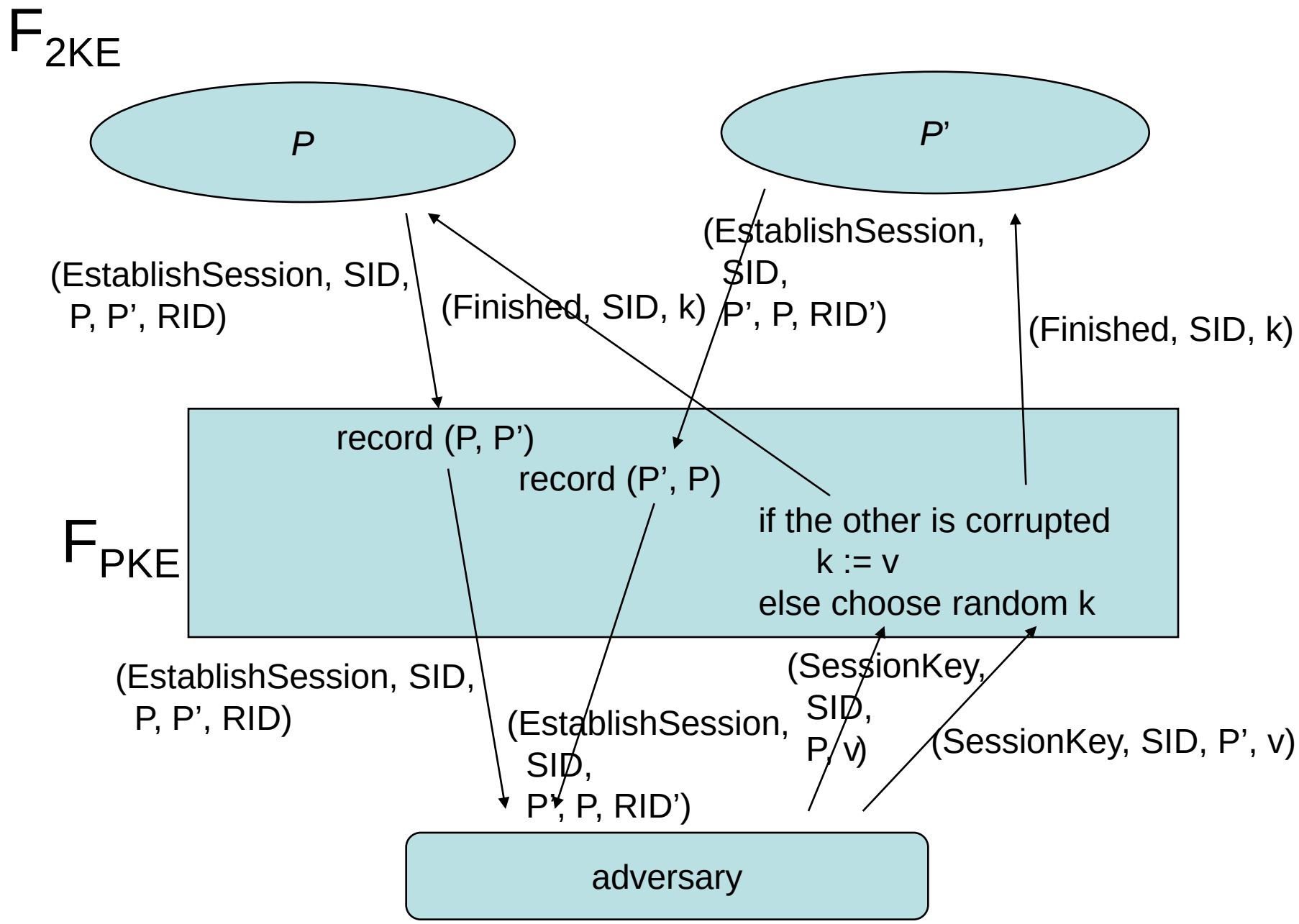

Environment Z
P = PID_owner
P'
(Decrypt, SID, c)
||
(PID_owner, SID')
m
(Encrypt, SID, m)
||
(PID_owner, SID')
c
F_CPKE
if PID_owner is corrupted
record (c,m)
if (c,m) not found
m := D(c)
c
simulator
(Encrypt, SID, m)
||
(PID_owner, SID')
c
record (c, m)
(Retrieve, SID)
(Retrieve, SID, v)
(Register, SID, e)
(KeyGen, SID)
e
F_REG
(Encrypt, SID, v, m)
c
F_PKE
adversary


F2MA
P
P'
(Authenticate, SID, P, P', RID)
(Authenticate, SID, P', P, RID')
Finished
Finished
Finished := false
record (P, P')
F2MA
Finished := true
if Finished
if Finished
(Output, SID, P)
(P, P', RID)
(P', P, RID')
(Output, SID, P')
adversary

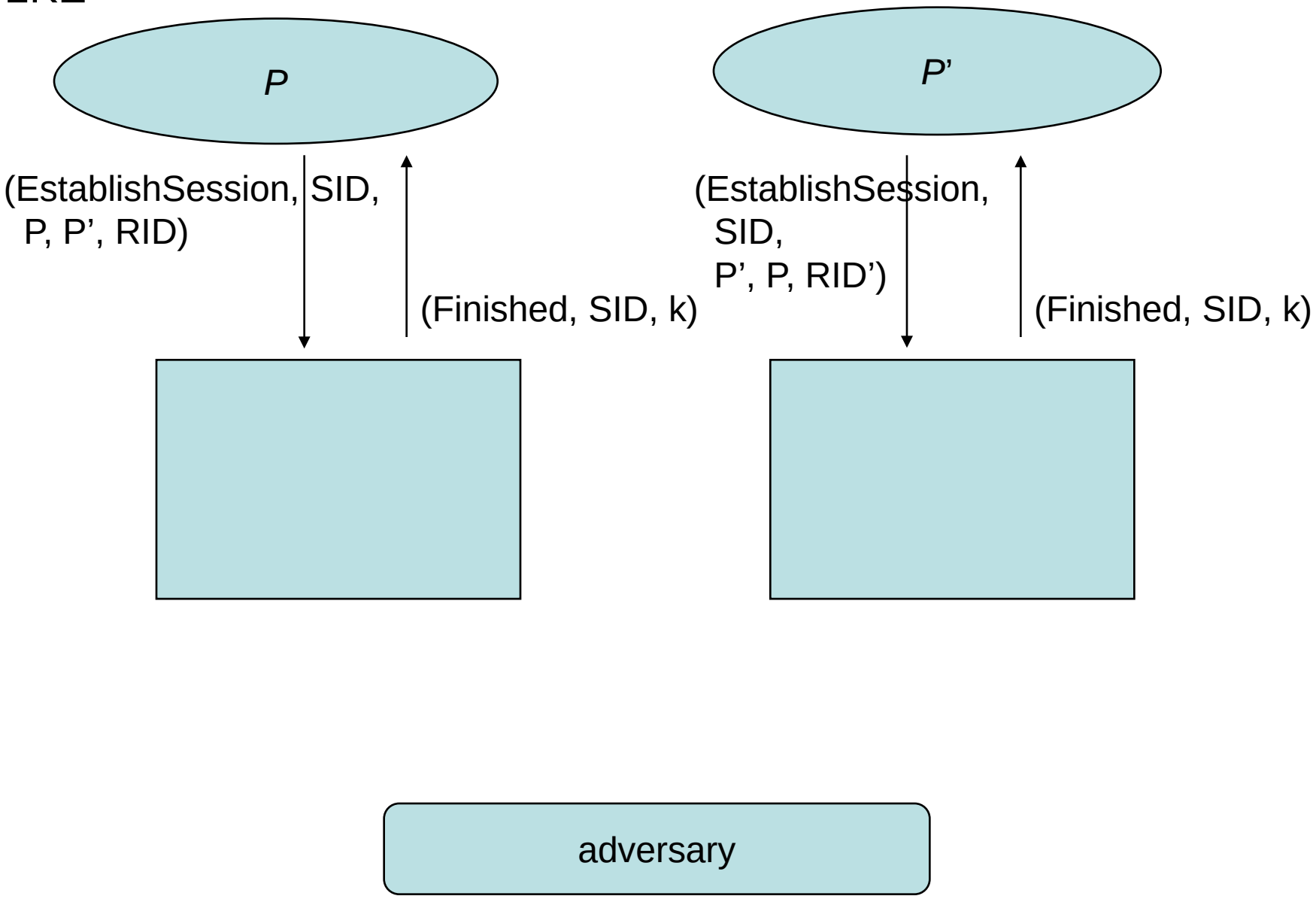
F_{2MA}を実現するプロトコル

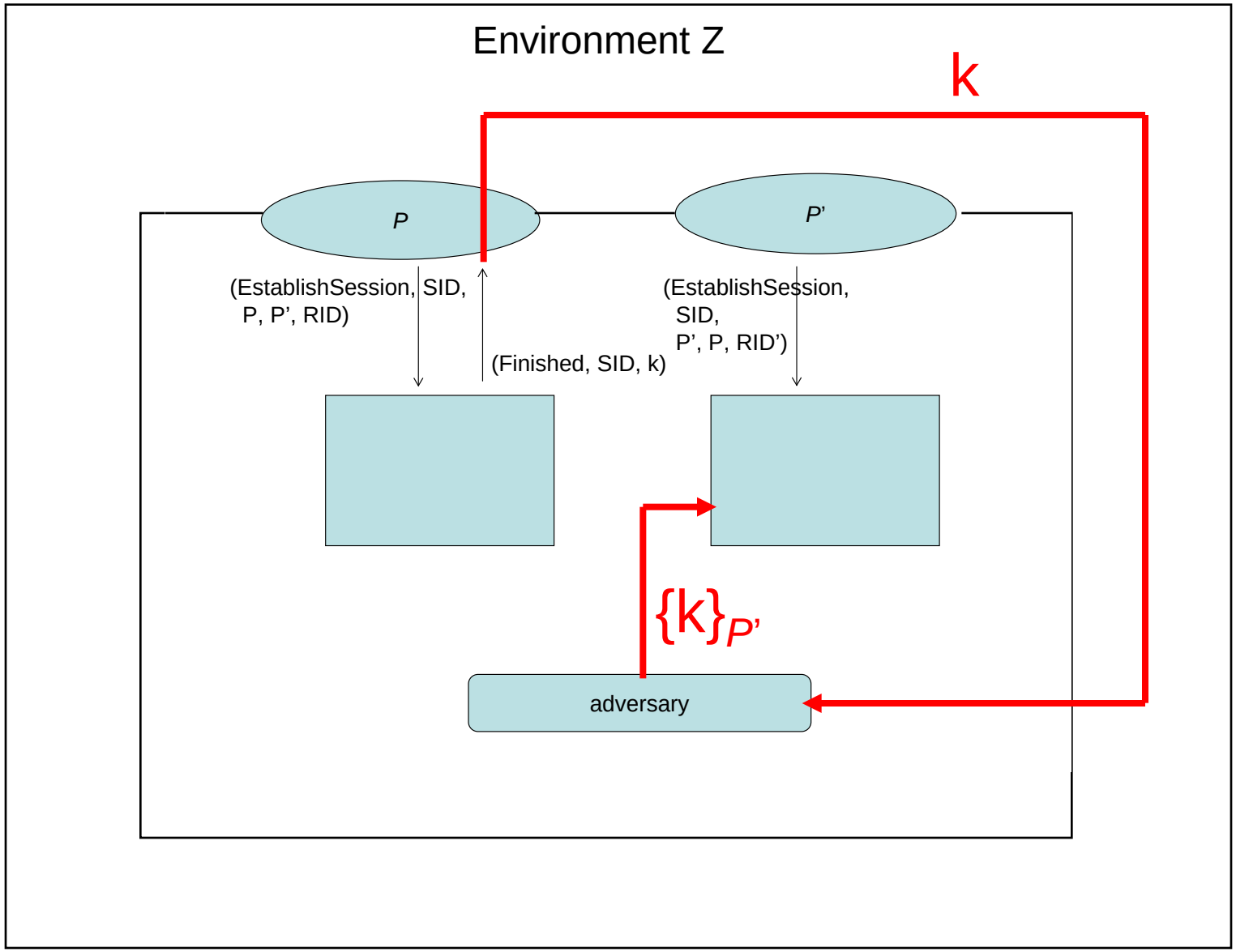



Environment Z
F2MA
P
P'
(Authenticate, SID, P, P', RID)
(Authenticate, SID, P', P, RID')
Finished
Finished
Finished := false
record (P, P')
F2MA
Finished := true
if Finished
if Finished
(Output, SID, P)
(P, P', RID)
(P', P, RID')
(Output, SID, P')
simulator
(Authenticate, SID, P, P', RID)
Finished
(Authenticate, SID, P', P, RID')
Finished
adversary


F2KE
P
P'
(EstablishSession, SID, P, P', RID)
(Finished, SID, k)
(EstablishSession, SID, P', P, RID')
(Finished, SID, k)
record (P, P')
record (P', P)
FPKE
if the other is corrupted
k := v
else choose random k
(EstablishSession, SID, P, P', RID)
(EstablishSession, SID, P', P, RID')
(SessionKey, SID, P, v)
(SessionKey, SID, P', v)
adversary

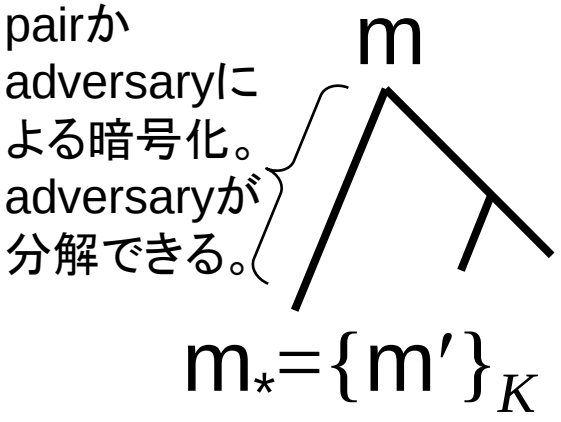
F_{2KE}を実現するプロトコル




Environment Z
k
P
P'
(EstablishSession, SID, P, P', RID)
(Finished, SID, k)
(EstablishSession, SID, P', P, RID')
{k}P'
adversary

Mapping Lemma

- t: プロトコルpのconcrete trace
- *symb*(t): tのsymbolic traceへの翻訳
  - adversaryが閉包にないメッセージmを配布するとき、[“fail”, m]に翻訳される。
  - このとき、mの代わりに以下のような部分メッセージm_*を配布するadversaryを定義可能。
  - m_*は閉包にないノンスであるか、
  - または、$m_*=\{m'\}_K$であって、$\{...\{m'\}_K...\}_{K'}$というメッセージは閉包に含まれる。



KもK'もhonest partyの鍵

Mapping Lemma

- m_*がノンスである場合
  - m_*は暗号化されているはず。
  - F_{CKPE}の定義より、adversaryの観測とm_*は独立。
  - m_*は一様に分布。
  - 従って、m_*を当てる確率はnegligible。
- $m_*=\{m'\}_K$である場合
  - この場合も、F_{CKPE}の定義より、$\{m'\}_K=E_k(r)$は、wide spreadな関数E_kによる乱数rの像なので、m_*は一様に分布。
  - 従って、m_*を当てる確率はnegligible。

2MAに対するsymbolic criterion

P: Dolev-Yaoプロトコル

(Dolev-Yao mutual authentication)

PのDolev-Yao traceが
Pによる出力メッセージ⟨Finished|P|P'|m⟩を
含んでいるならば、それに先んじて、
P'への入力メッセージ⟨Started|P|P'|m′⟩を含む。

2KEに対するsymbolic criterion

(agreement)

P_0とP_1がプロトコルを開始した後、

P_0が$\langle \text{Finished}|\text{m}_0\rangle$を出力し、

P_1が$\langle \text{Finished}|\text{m}_1\rangle$を出力したならば、

$\text{m}_0 = P_0|P_1|R$かつ$\text{m}_0 = P_1|P_0|R$

(real-or-random secrecy)

もとのDolev-Yaoプロトコルを$\mathbf{P}$、

実際の鍵R_rの代わりに乱数R_fを返す

プロトコルを$\mathbf{P}_f$としたとき、

任意のadversary戦略Ψに対して、

$$pattern(\Psi(\mathbf{P})) = pattern(\Psi(\mathbf{P}_f)[R_f \rightarrow R_r])$$

adversary戦略Ψ

- Dolev-Yaoを満たすadversary eventの列
  - receive, enc, dec, pair, extract-l, extract-r, random, name, pubkey, deliver
- intruderストランドのようなもの
- プロトコル$\mathbf{P}$に対してadversary戦略Ψを実行して得られるDolev-Yao traceを、$\Psi(\mathbf{P})$と書く。
- 要するに、symbolicな世界においてadversaryを網羅している。

symbolic criterion → UC

- *symb*(p)のDolev-Yao trace τ' に対して、

  $$pattern(symb(t)) = pattern(symb(t^{F2KE})) = \tau'$$

  という条件のもとで、tとt^{F2KE}の分布は等しい。
  - t^{F2KE}はideal functionalityのtrace
- $pattern(symb(t))$と$pattern(symb(t^{F2KE}))$の分布は等しい。
  - 乱数を固定すると両者は等しい。
  - 等しくない場合には、real-or-random secrecyに違反するΨを構成することができる。