Determining the Core Primitive for Optimally Secure Ratcheting

IACR Asiacrypt 2020

2020-12-29

1 Horst Görtz Institute for IT Security
Chair for Network and Data Security
Ruhr University Bochum

2 Security and Cryptography Laboratory
École polytechnique fédérale de Lausanne

Fatih Balli[2], **Paul Rösler**[1], Serge Vaudenay[2]

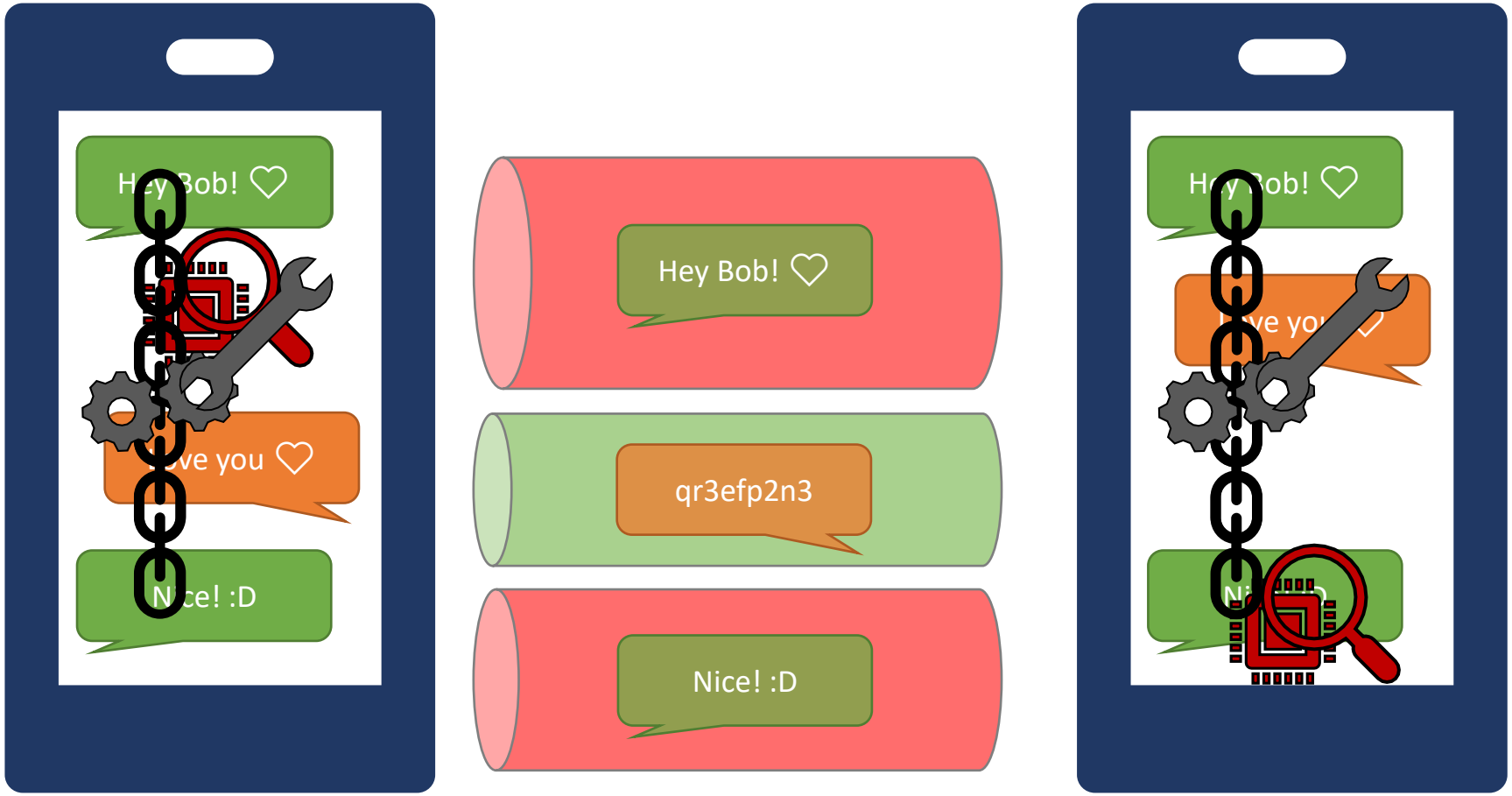
Ratcheting

“Ratchet”-Mechanism:

- Invalidate old secrets
- Sample and include new secrets
- Origin:

Simple construction:

- Repetition and mix of key exchanges (e.g., DH)



Ratcheted Key Exchange (RKE)

URKE: init snd rcv

deterministic

"Ratchet"-Mechanism:

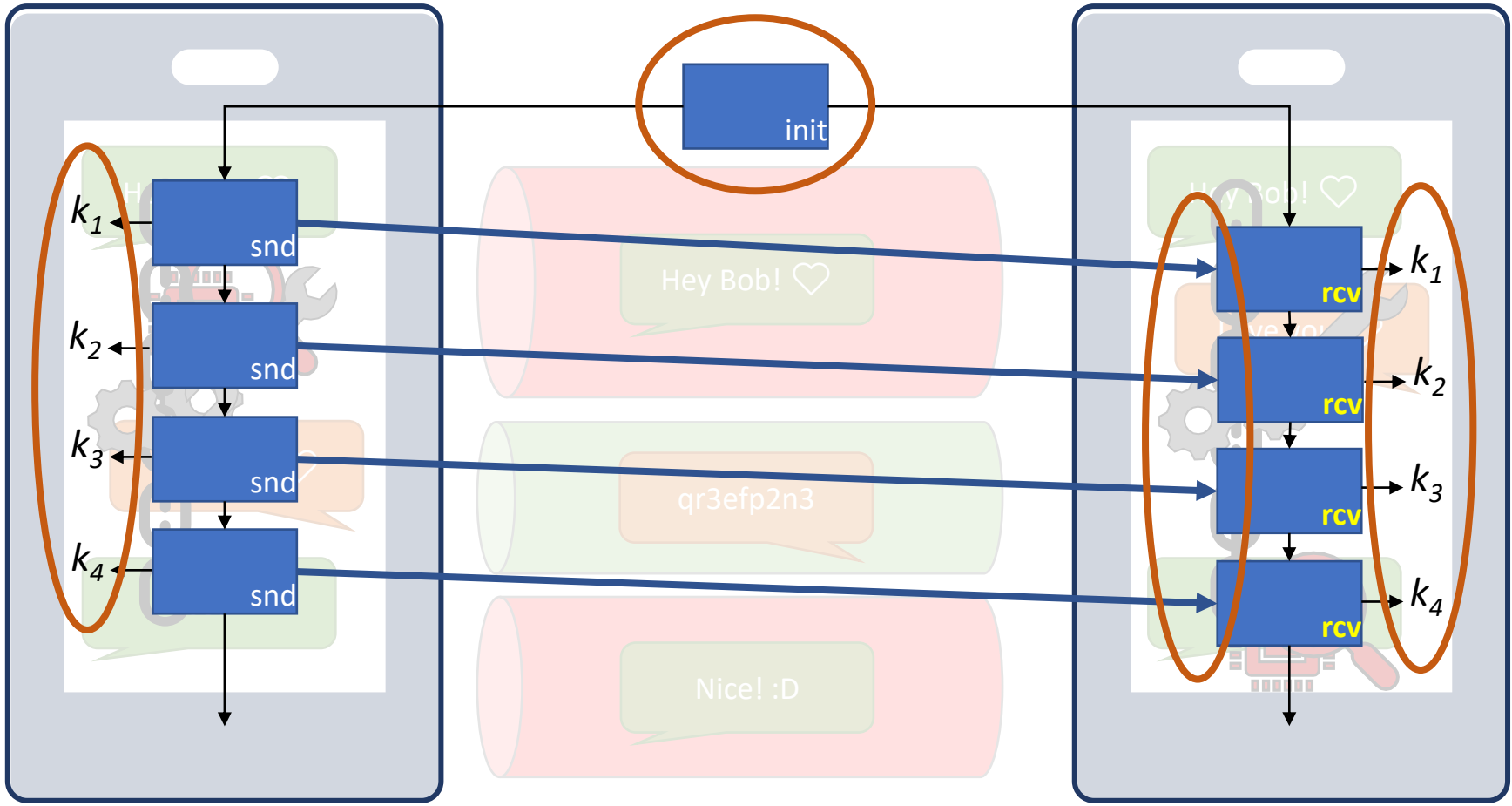
- Invalidate old secrets
- Sample and include new secrets
- Origin:

Simple construction:

- Repetition and mix of key exchanges (e.g., DH)

Simplifications here:

- Abstract initialization
- Continuous key exchange
- Unidirectional communication

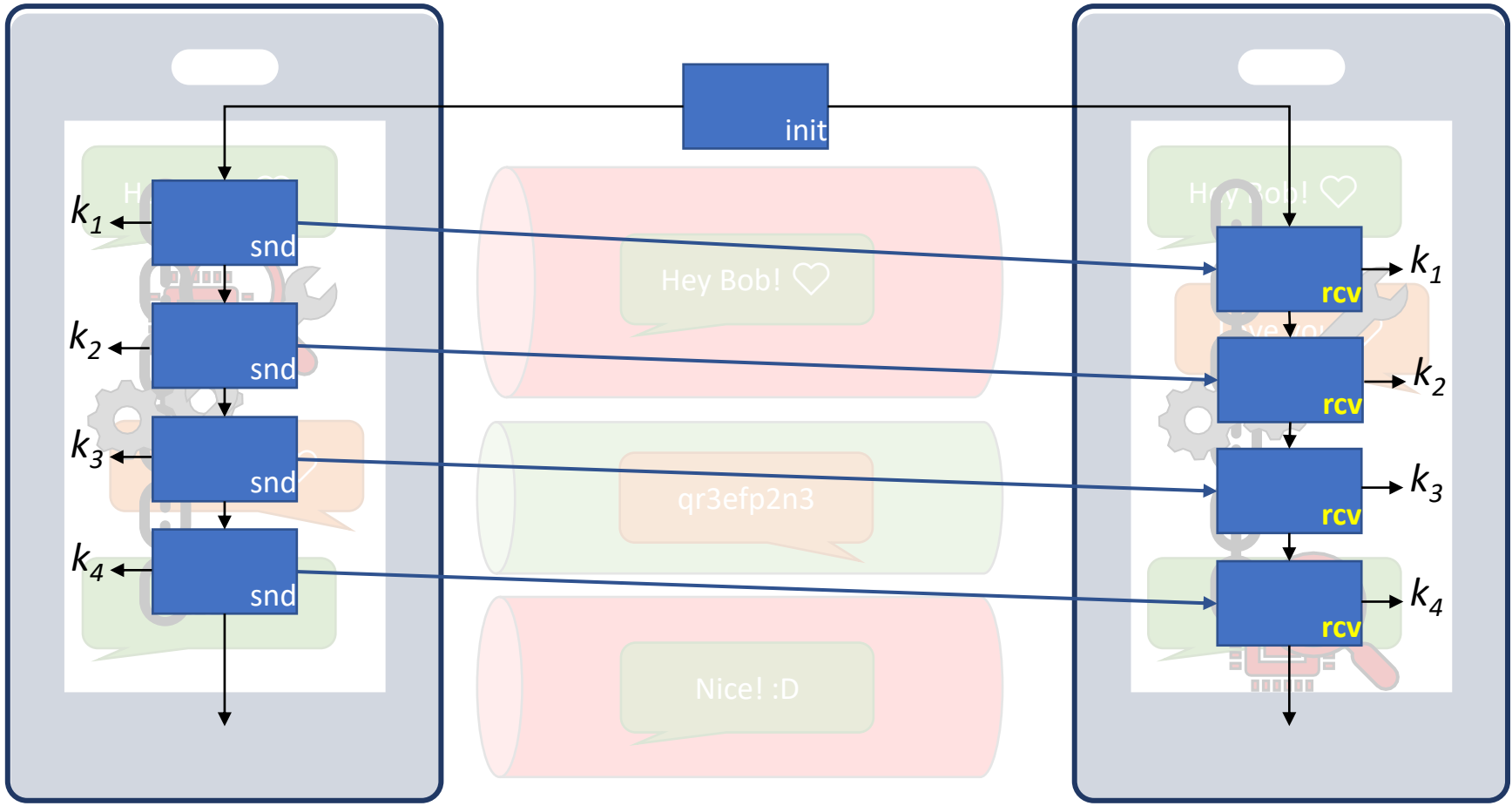


Optimally Secure RKE

URKE: init snd rcv

Theoretic work [P**R**18,JS18]:

- Optimal security
  → Secure key whenever possible
- Certain conditions
  Standard PKC sufficient
- Bidirectional
  Heavier™ tools used

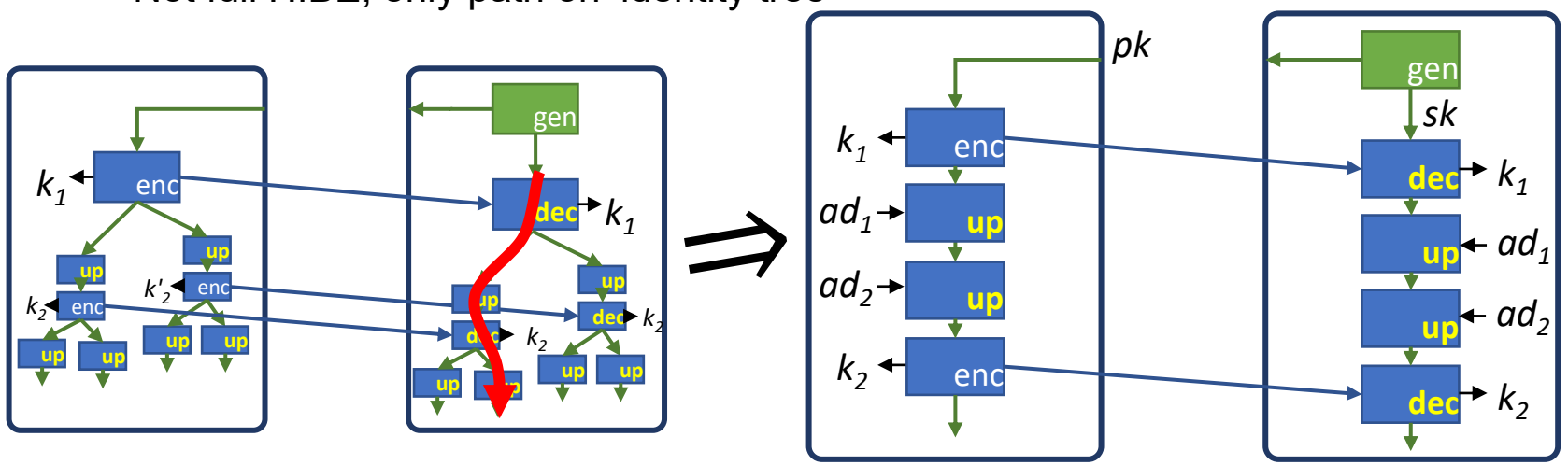


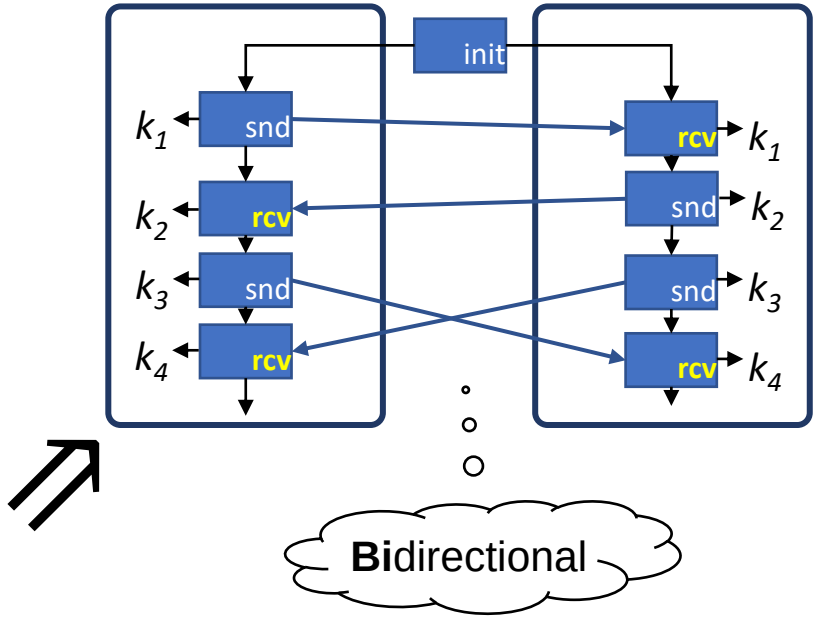
Key-Updatable PKE

U/BRKE: init snd rcv
kuKEM: gen enc dec up

Theoretic work [P**R**18,JS18]:

- Optimal security
- Bidirectional
  - Heavier™ tools used
- Key-updatable KEM
  - Update pk and sk independently and forward securely
- Based on (expensive) HIBE
  - Not full HIBE, only path on ‘identity tree’





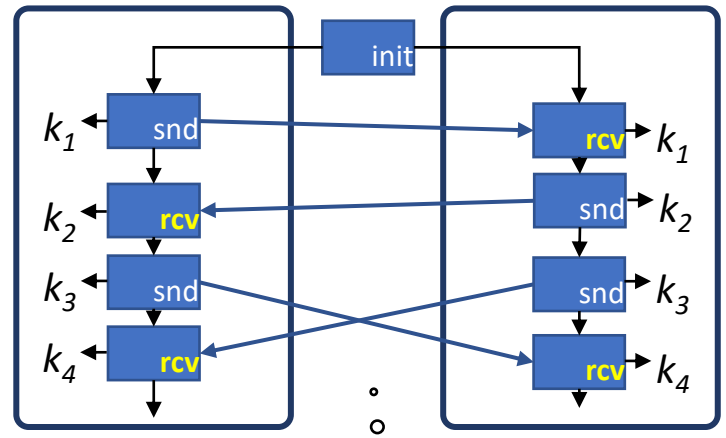
Research Question

Theoretic work [P**R**18,JS18]:

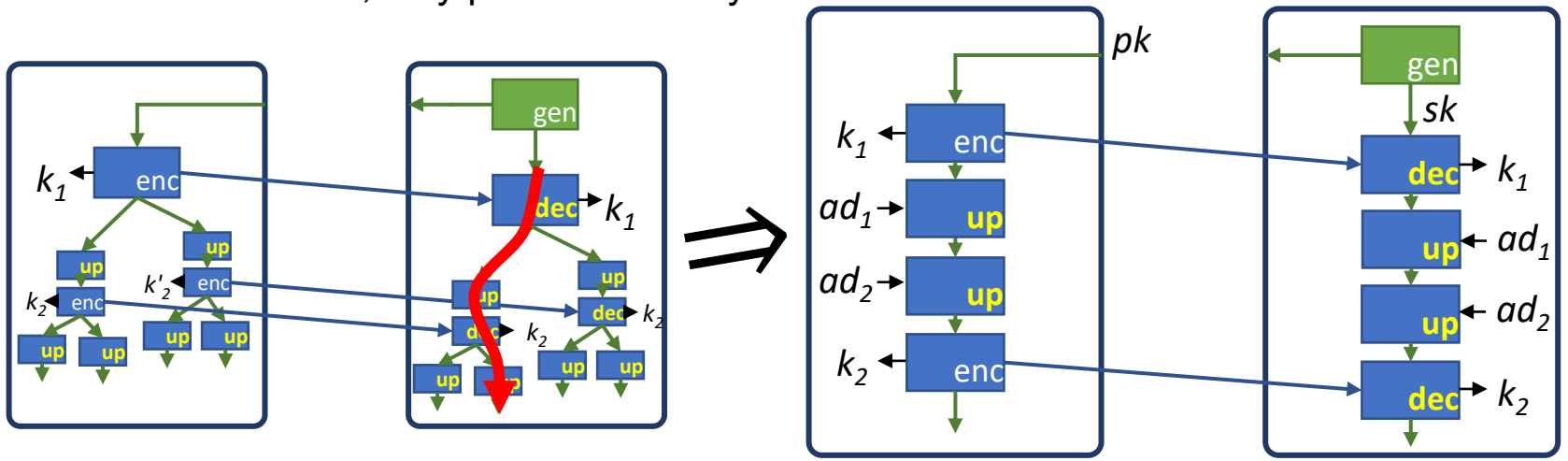
- Optimal security
- Bidirectional
  - Heavier™ tools used
- Key-updatable KEM
  - Update pk and sk independently and forward securely
- Based on (expensive) HIBE
  - Not full HIBE, only path on ‘identity tree’

Question:
Under which conditions
is this necessary?

U/BRKE: init snd rcv
kuKEM: gen enc dec up



Bidirectional



Research Question

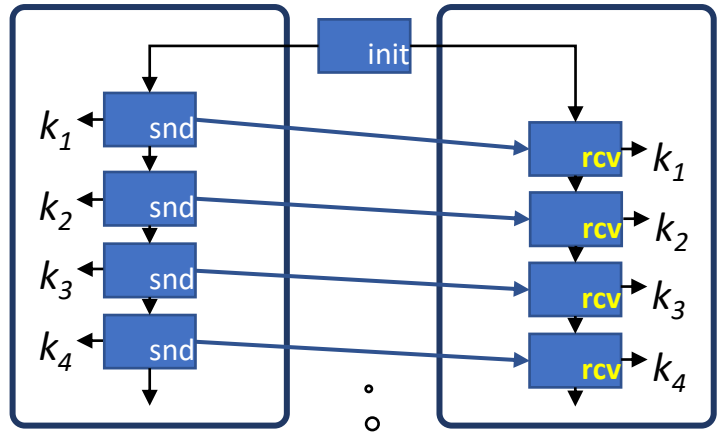
Theoretic work [P**R**18,JS18]:

- Optimal security
- Bidirectional
  - Heavier™ tools used
- Key-updatable KEM
  - Update pk and sk independently and forward securely
- Based on (expensive) HIBE
  - Not full HIBE, only path on ‘identity tree’

Both realistic:
DualEC,
little entropy, …

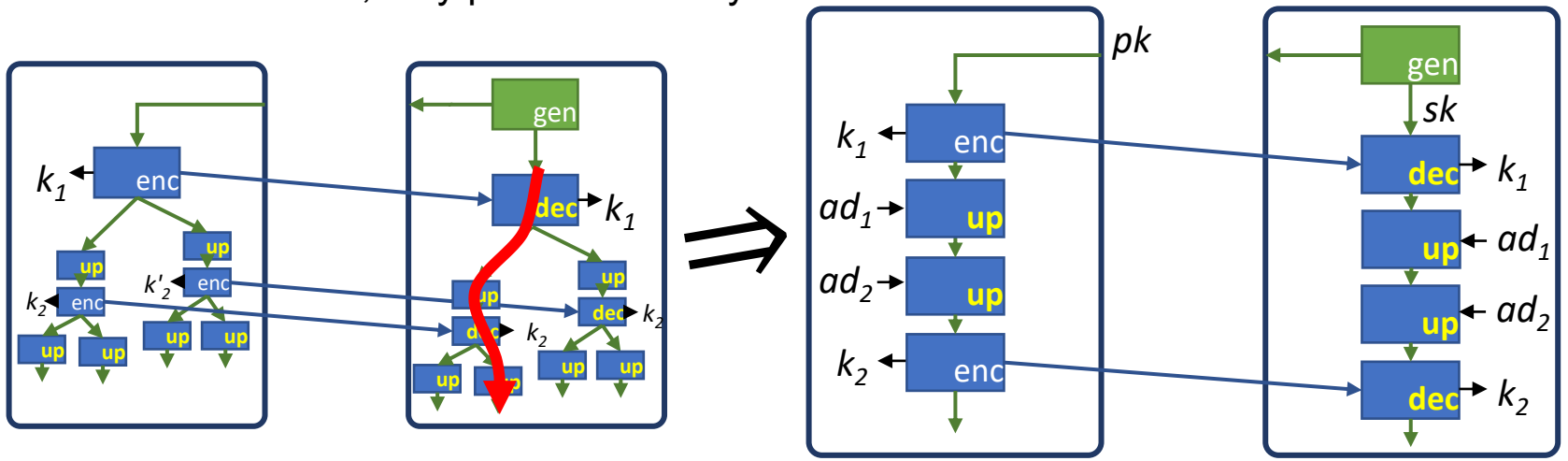
Answer:
Optimal security under
state exposures and
randomness manipulation

URKE: init snd rcv
kuKEM: gen enc dec up



Unidirectional

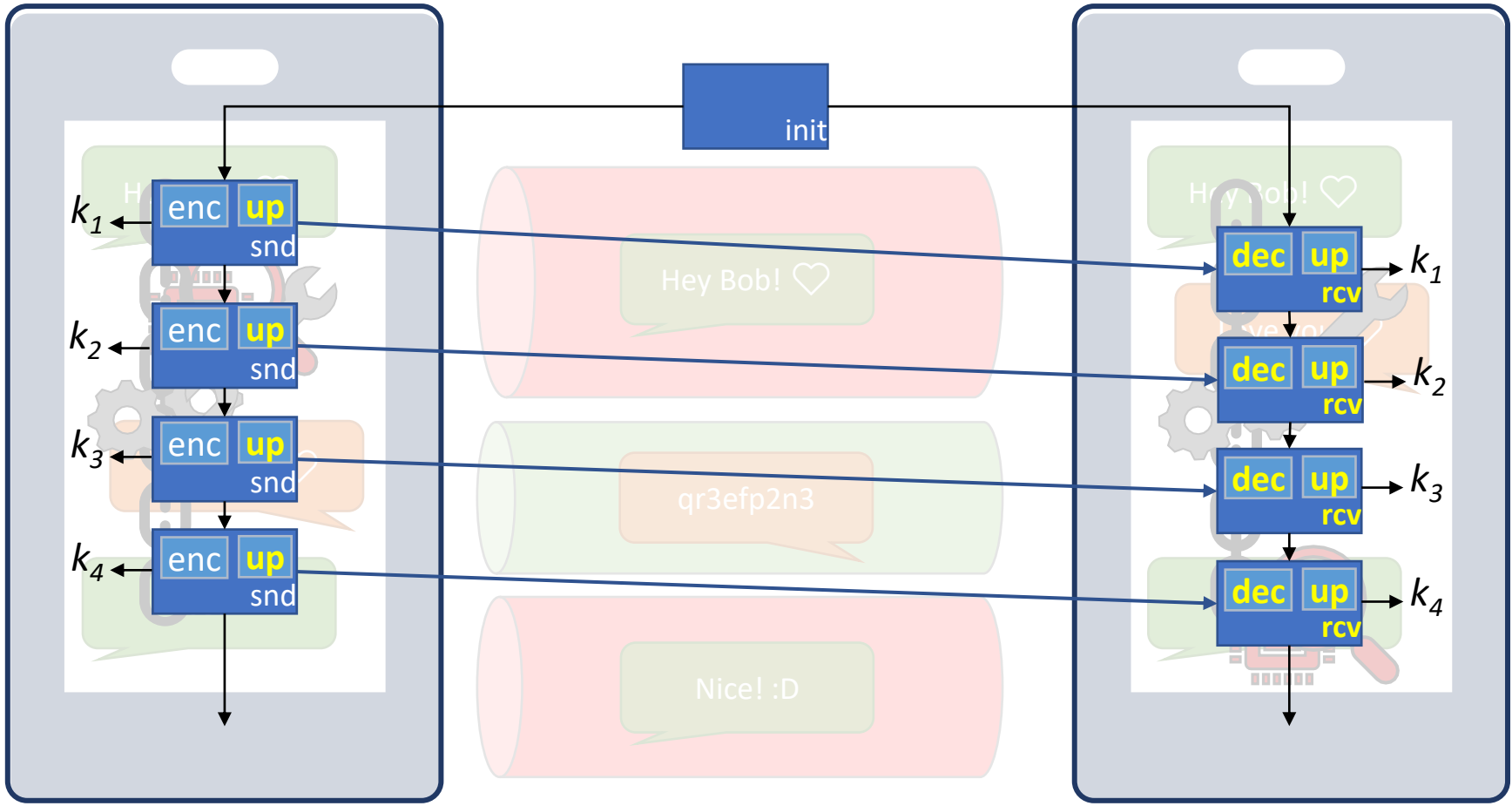
How?
Generically build
URKE from kuKEM
and kuKEM from URKE



kuKEM → Unidirectional RKE

URKE: init snd rcv
kuKEM: gen enc dec up

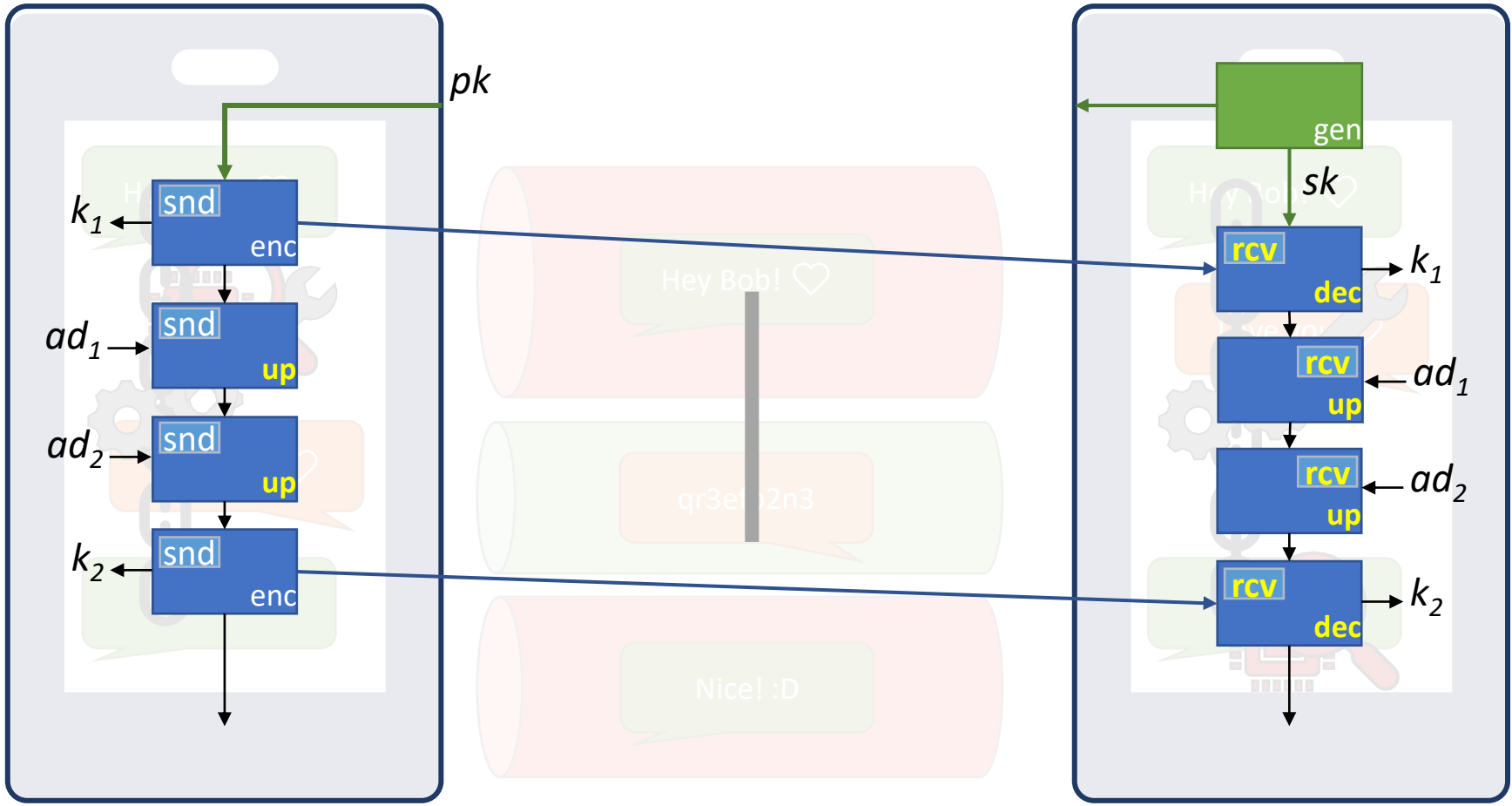
- Encapsulate symmetric key
- Update pk and sk independently wrt. transcript
  - → Diverge states on active attacks
  - → Update pk without revealing sk under manipulated randomness
- Plus random oracle and message authentication code



Unidirectional RKE → kuKEM

URKE: init snd rcv
kuKEM: gen enc dec up

- Send to establish symmetric key
- Send to update pk
  - → Receive to update sk?!
  - → How to update synchronously?
  - → Send is probabilistic and outputs ciphertext



Unidirectional RKE → kuKEM

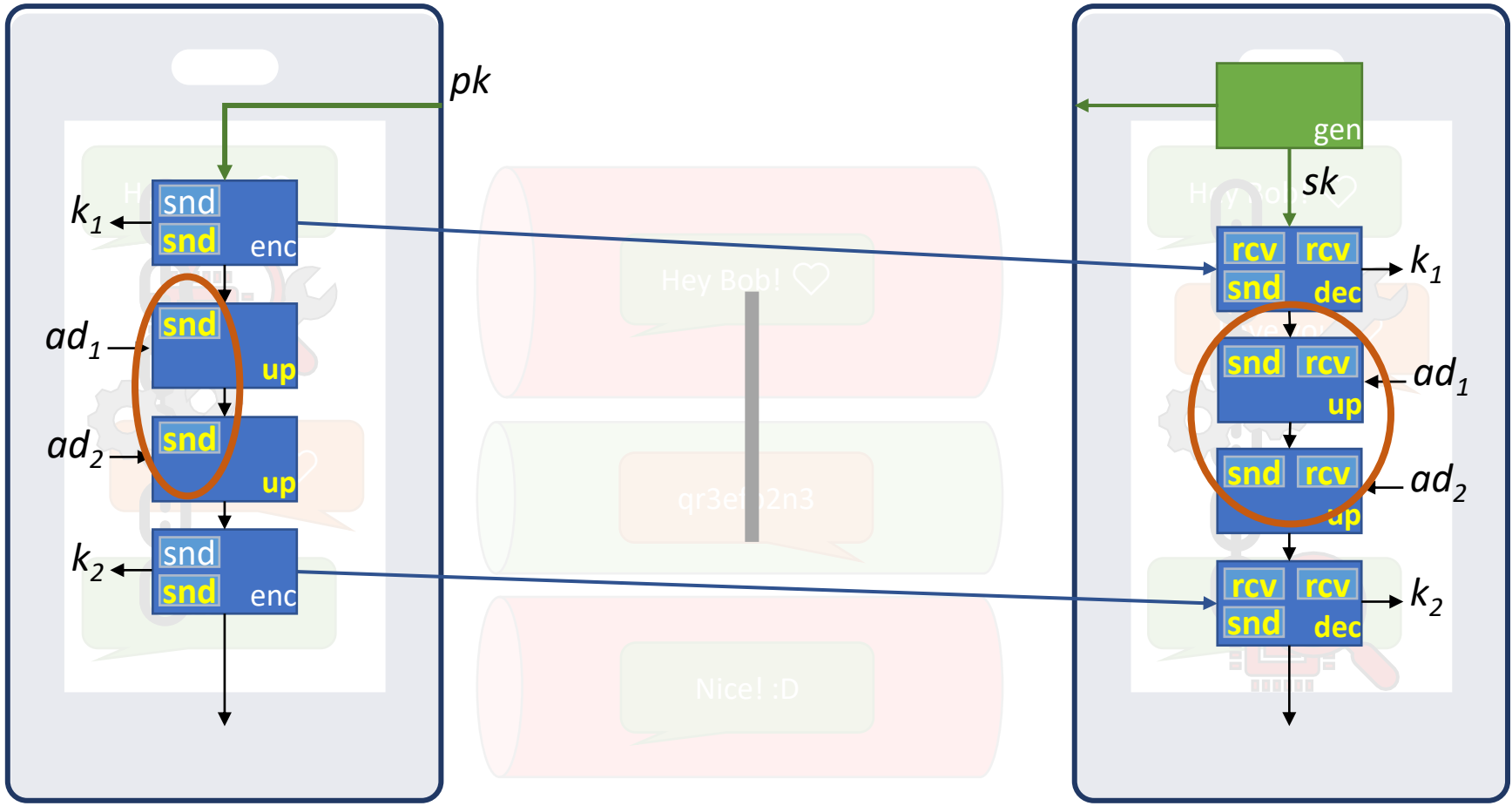
URKE: init snd rcv
kuKEM: gen enc dec up

- Send to establish symmetric key
- Send to update pk
  - → Receive to update sk?!
  - → How to update synchronously?
  - → Send is probabilistic and outputs ciphertext

→De-randomize snd

- Use snd to update pk
- Use snd and rcv to update sk
- → Replay snd in dec to generate ciphertext

- Additional snd-rcv update in encapsulation



Contributions

Theoretic work [P**R**18,JS18]:

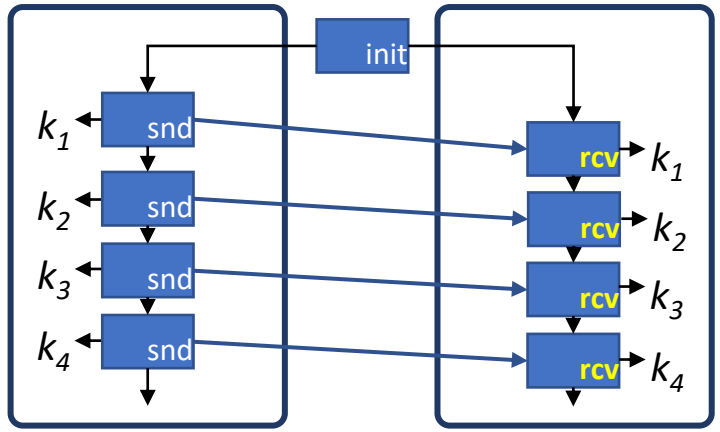
- Optimal security
- Bidirectional
  - Heavier™ tools used
- Key-updatable KEM
  - Update pk and sk independently and forward securely
- Based on (expensive) HIBE
  - Not full HIBE, only path on ‘identity tree’

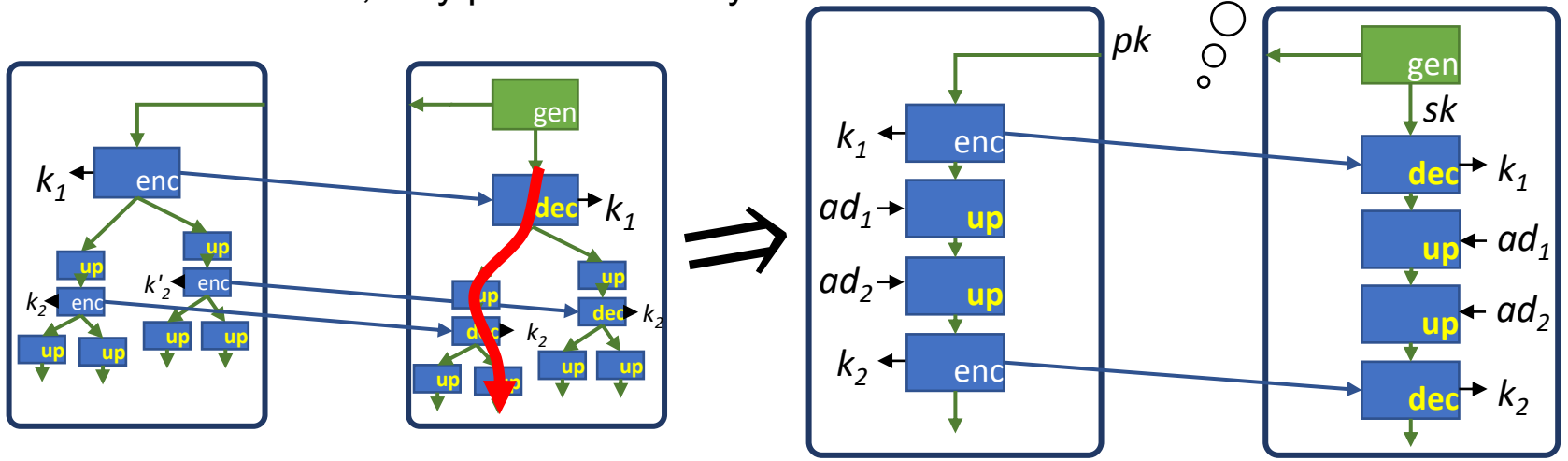
Answer:
Optimal security under
state exposures and
randomness manipulation

Both realistic:
DualEC,
little entropy, …

Core primitive for ratcheting
→Easier to build

URKE: init snd rcv
kuKEM: gen enc dec up





ia.cr/2020/148

@roeslpa