Cryptography analysis of monocypher’s handshake protocol

under MITM attack with small order keys

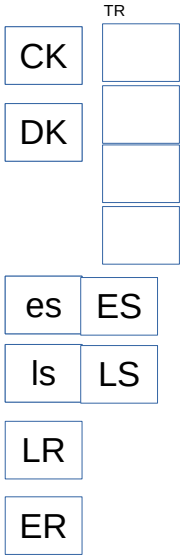
Mike, NOV.2018, CC0

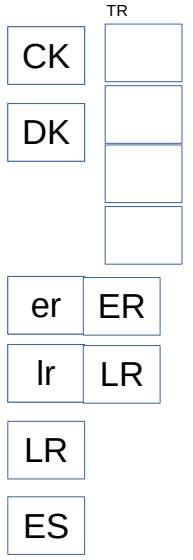
INITIALIZATION
on 3 sides

Sender

MITM

Receiver

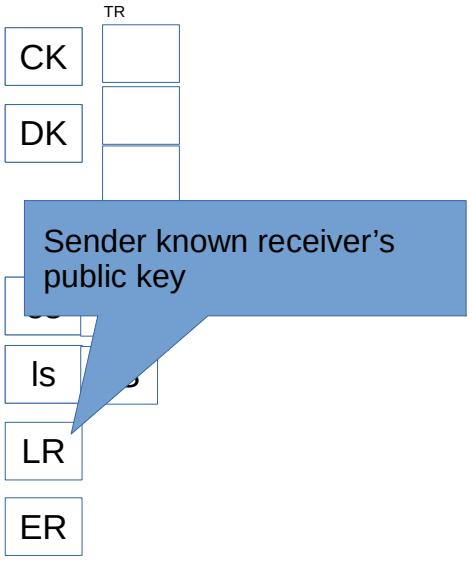


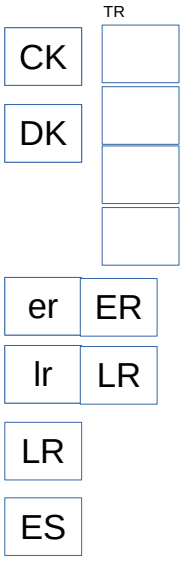


Sender

MITM

Receiver

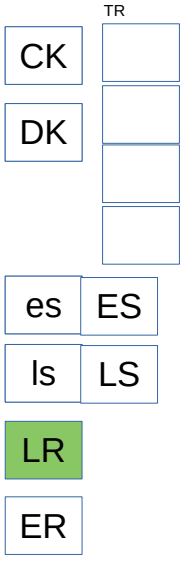


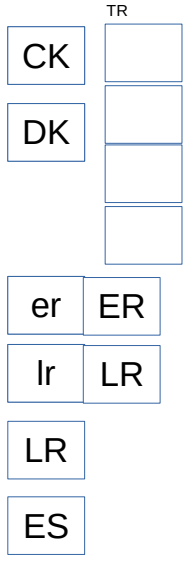


Sender

MITM

Receiver





Sender

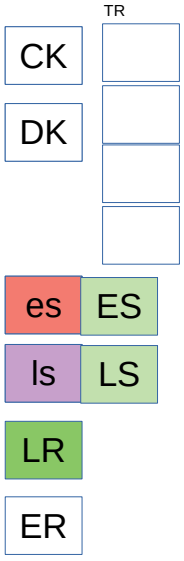
MITM

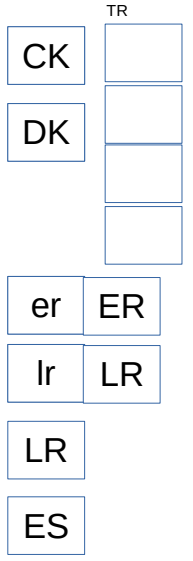
Receiver

Sender

MITM

Receiver

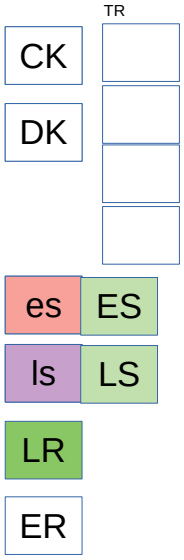




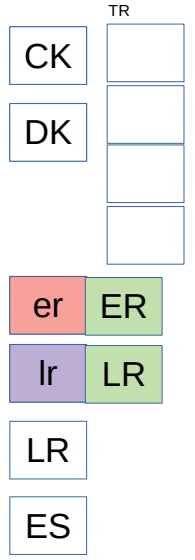
Sender

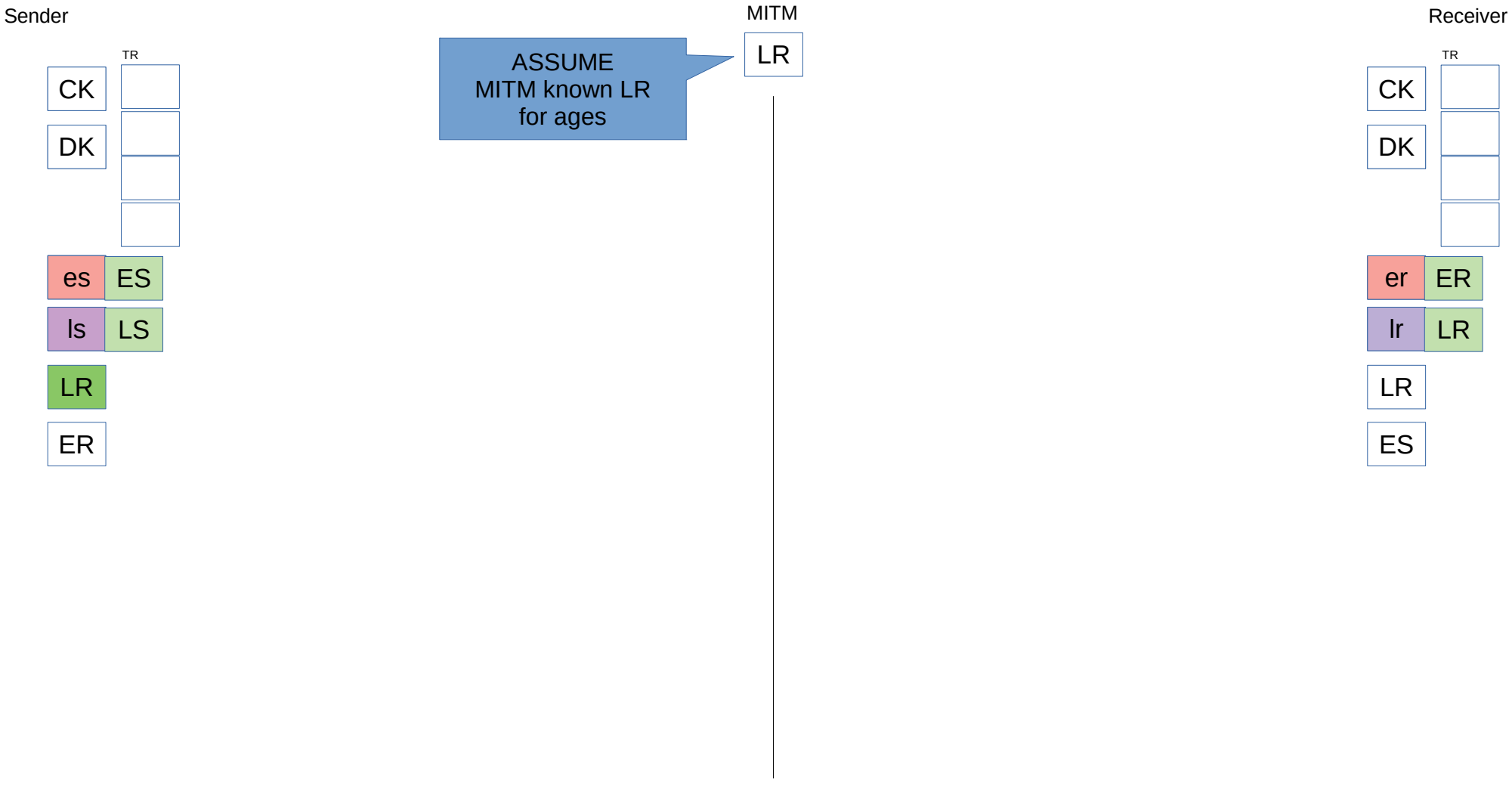
MITM

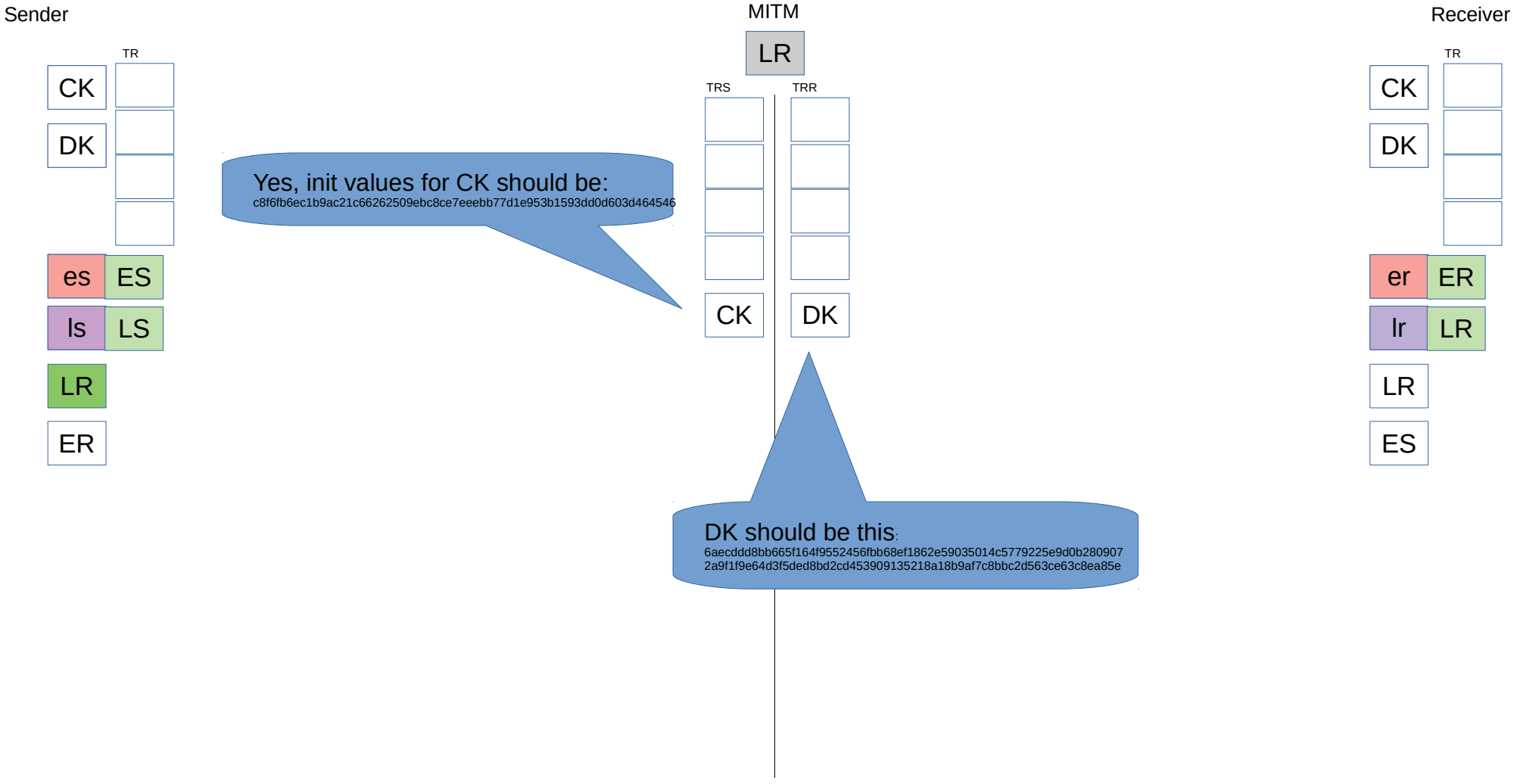
Receiver

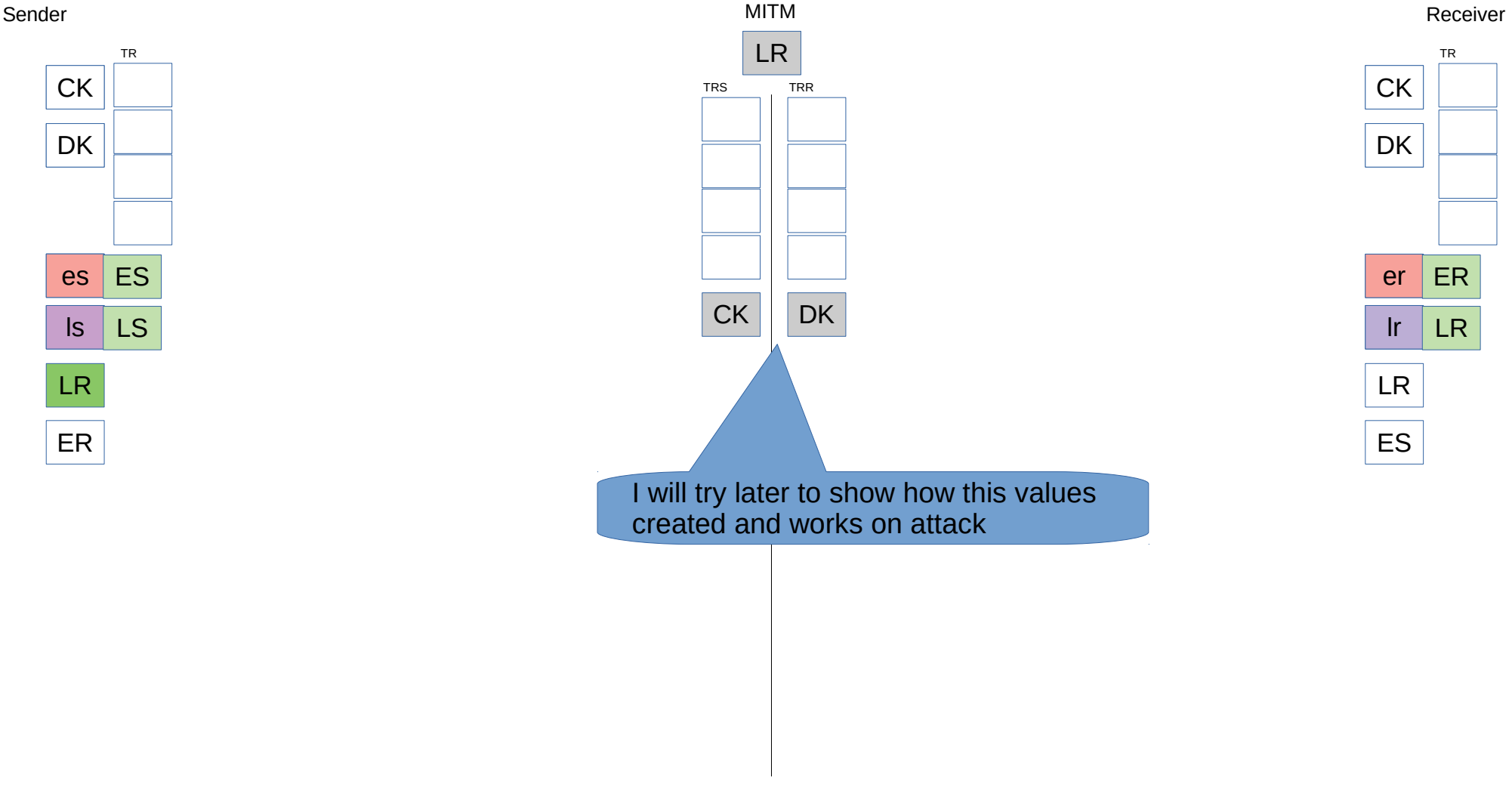


Receiver init pairs
and ready for exchange

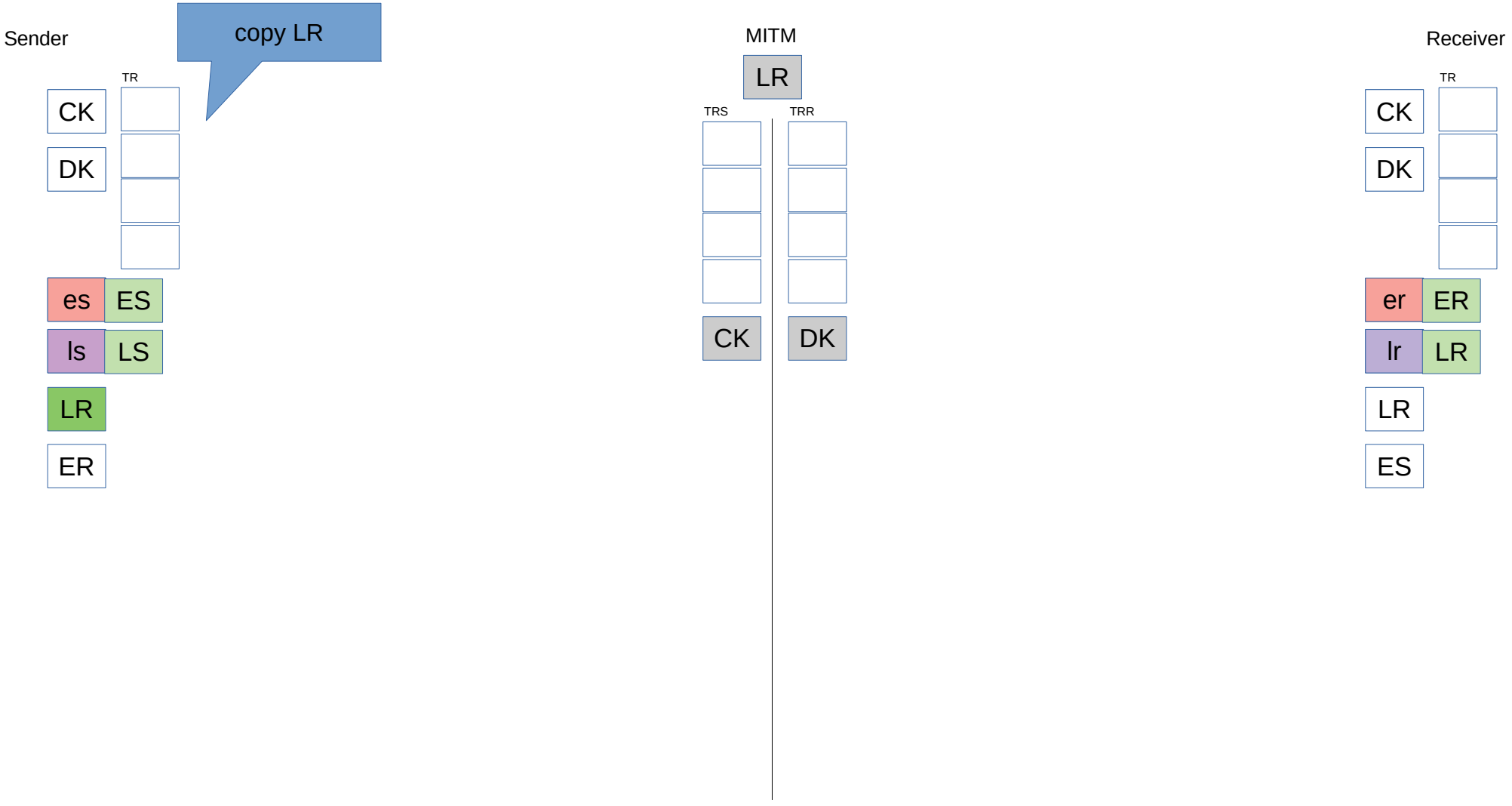


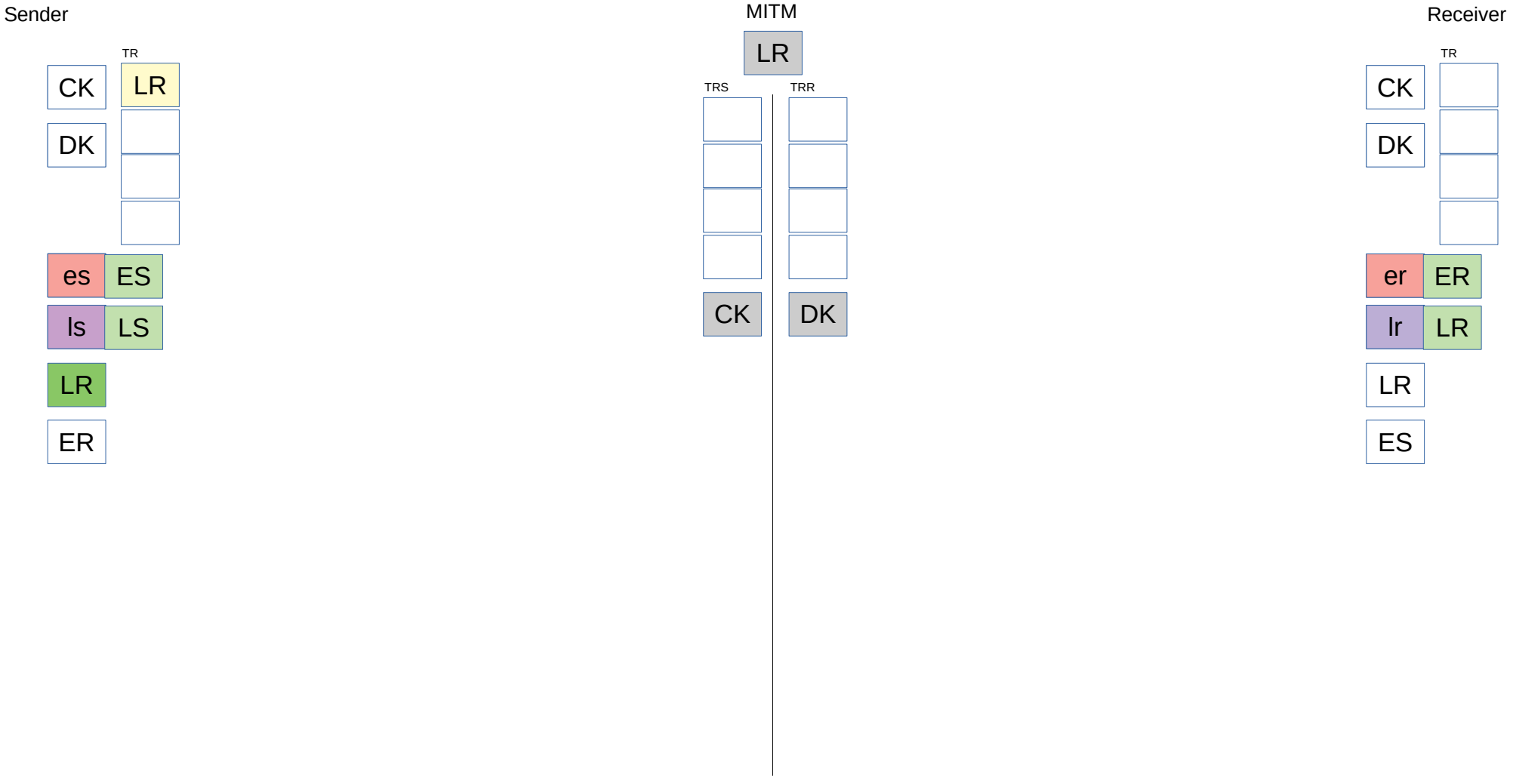

Sender
TR
CK
DK
es
ES
ls
LS
LR
ER
MITM
LR
ASSUME
MITM known LR
for ages
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

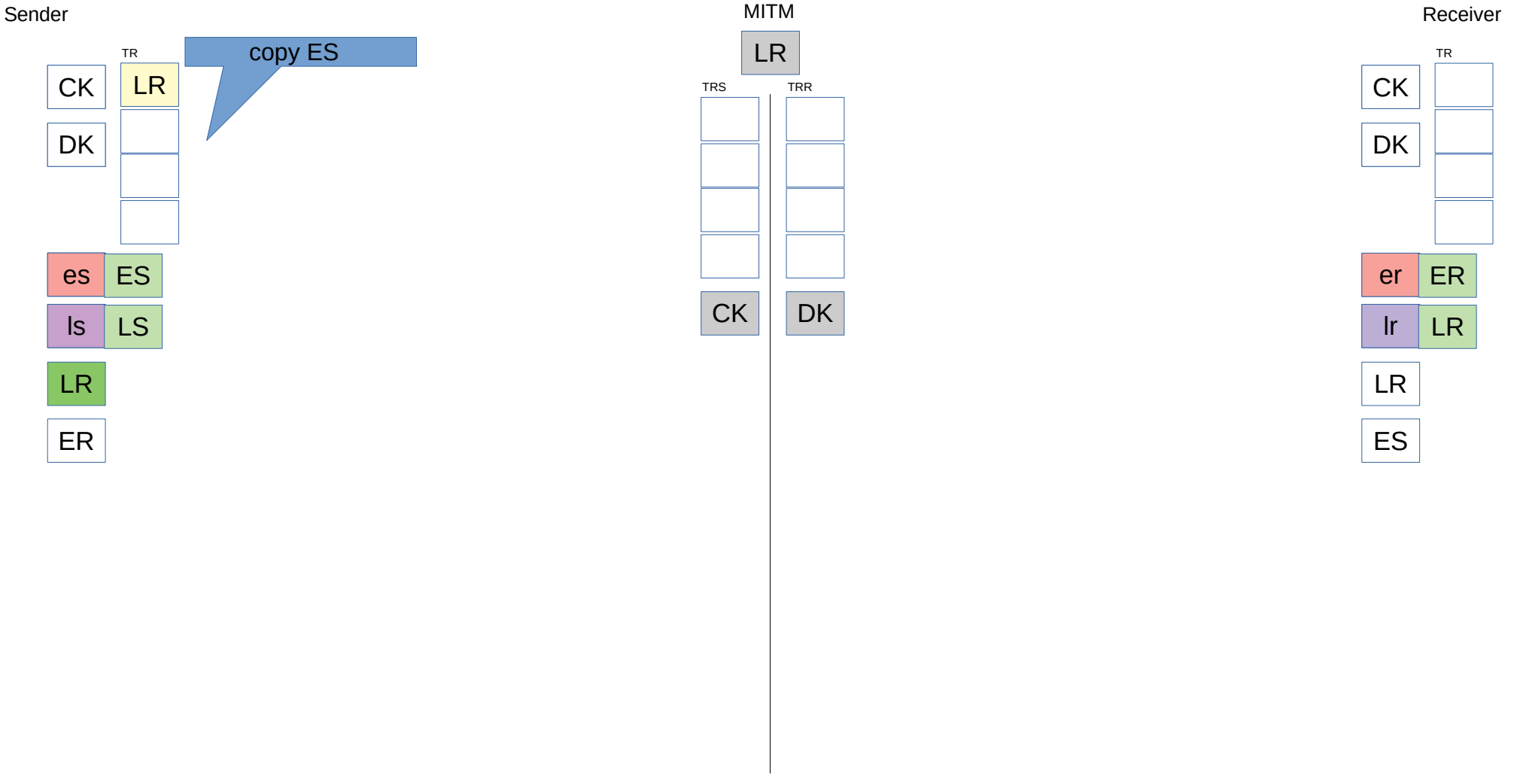

Sender
MITM
Receiver
TR
CK
DK
es
ES
ls
LS
LR
ER
LR
TRS
TRR
CK
DK
Yes, init values for CK should be:
c8f6fb6ec1b9ac21c66262509ebc8ce7eeebb77d1e953b1593dd0d603d464546
DK should be this:
6aecddd8bb665f164f9552456fbb68ef1862e59035014c5779225e9d0b280907
2a9f1f9e64d3f5ded8bd2cd453909135218a18b9af7c8bbc2d563ce63c8ea85e
TR
CK
DK
er
ER
lr
LR
LR
ES

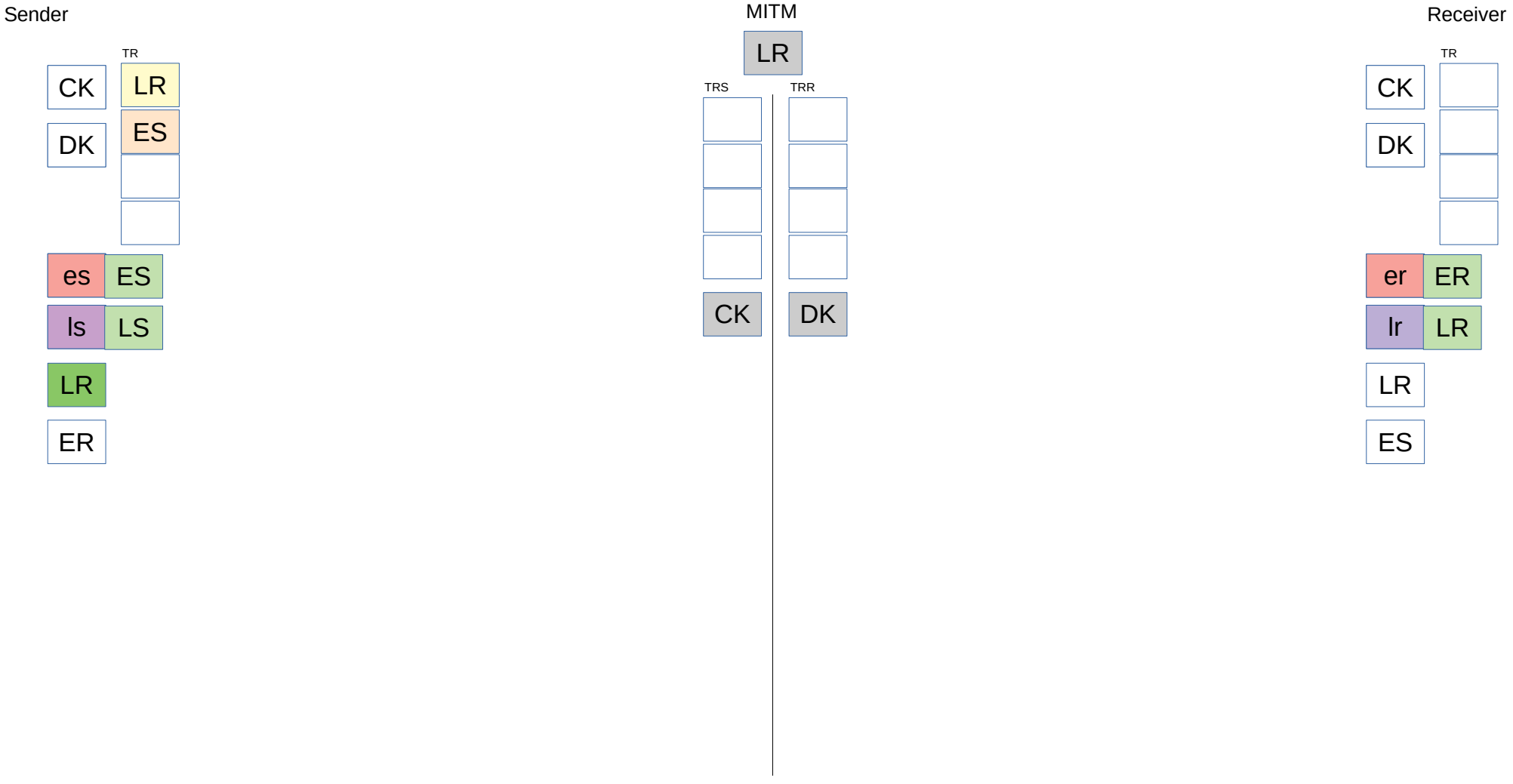

Sender
TR
CK
DK
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
CK
DK
I will try later to show how this values created and works on attack
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

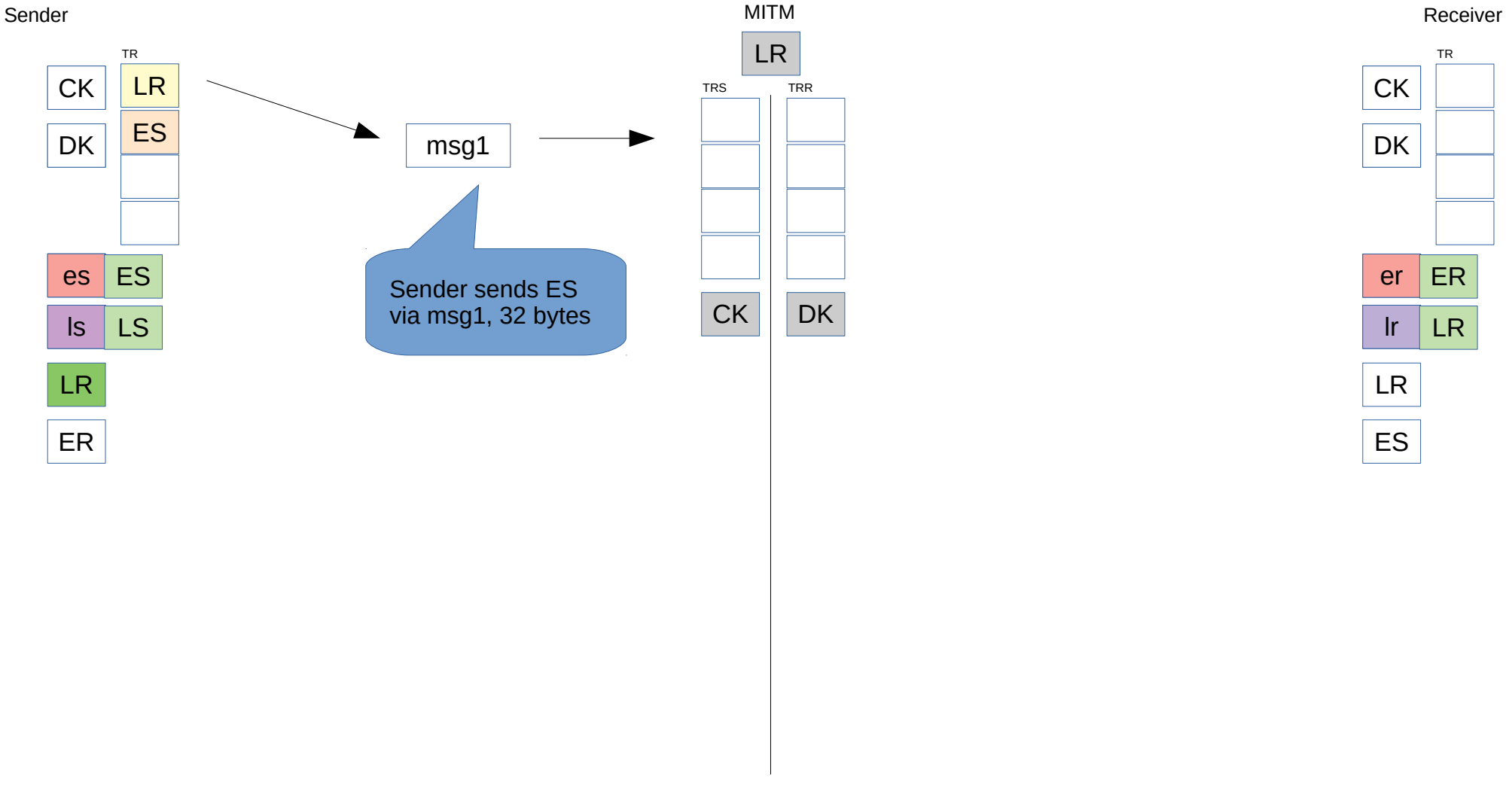
REQUEST

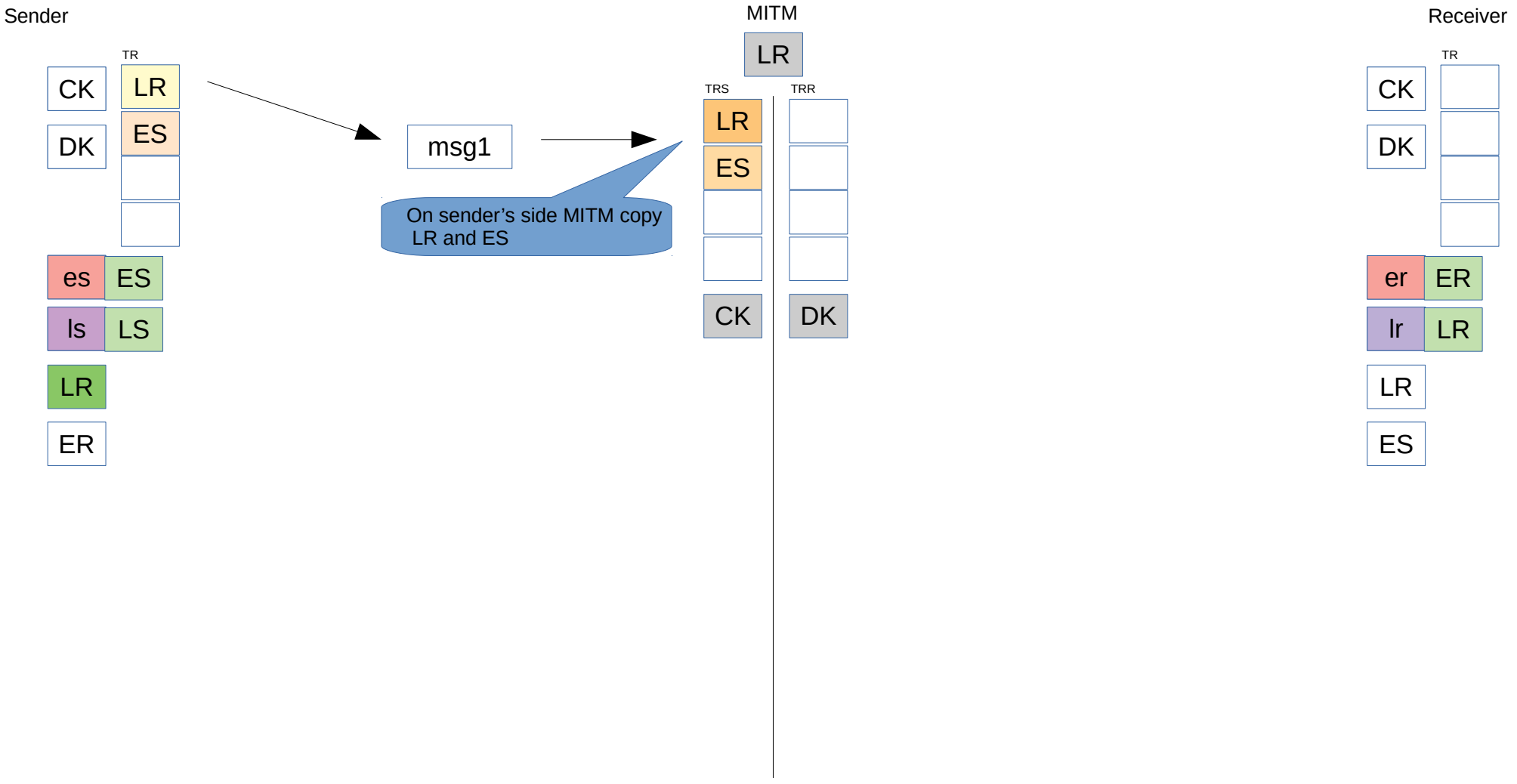

Sender
copy LR
TR
CK
DK
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
CK
DK
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

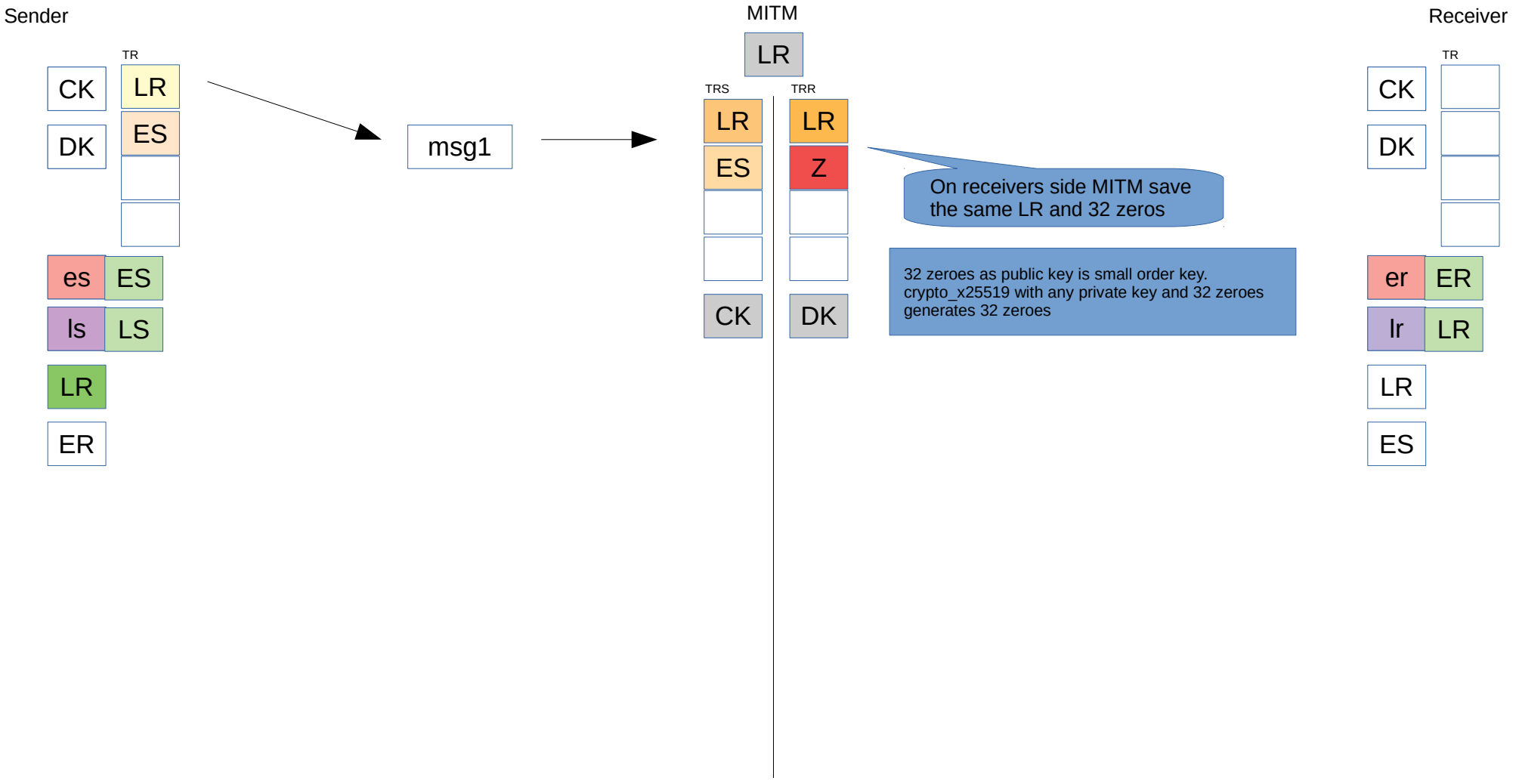

Sender
MITM
Receiver
TR
CK
LR
DK
es
ES
ls
LS
LR
ER
LR
TRS
TRR
CK
DK
TR
CK
DK
er
ER
lr
LR
LR
ES

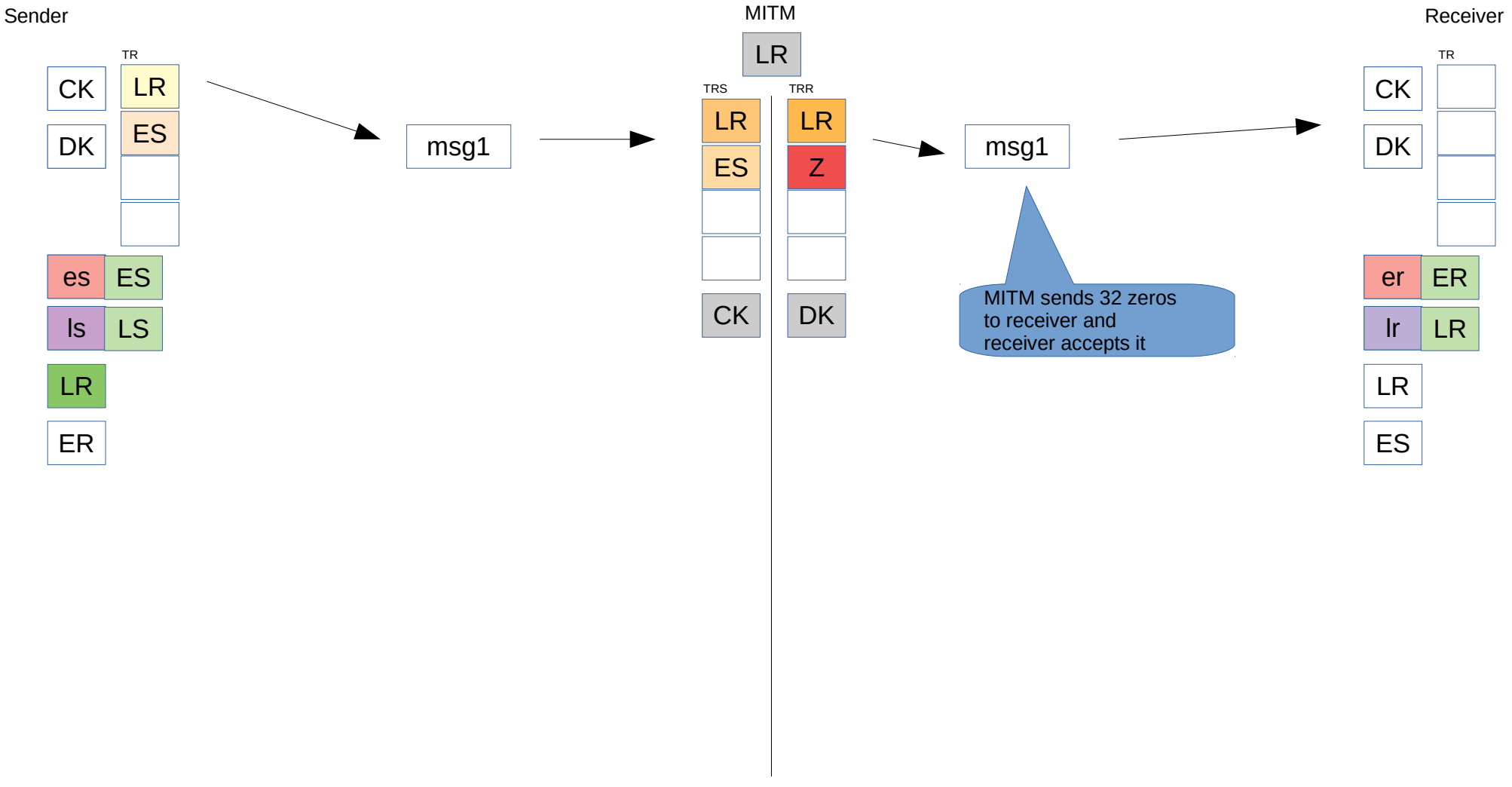

Sender
MITM
Receiver
TR
copy ES
CK
LR
DK
es
ES
ls
LS
LR
ER
LR
TRS
TRR
CK
DK
TR
CK
DK
er
ER
lr
LR
LR
ES


Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
CK
DK
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

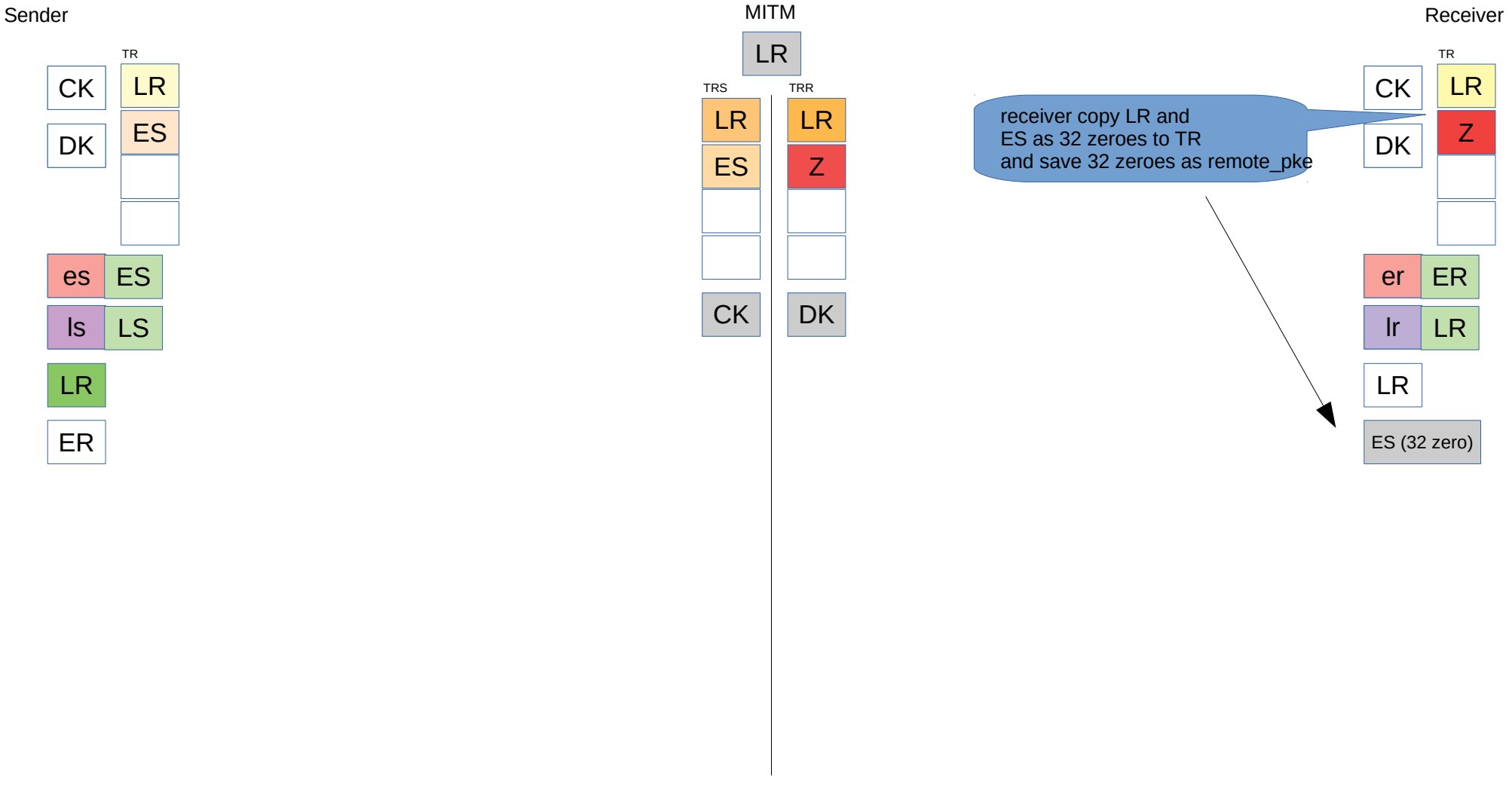

Sender
MITM
Receiver
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg1
Sender sends ES
via msg1, 32 bytes
LR
TRS
TRR
CK
DK
TR
CK
DK
er
ER
lr
LR
LR
ES

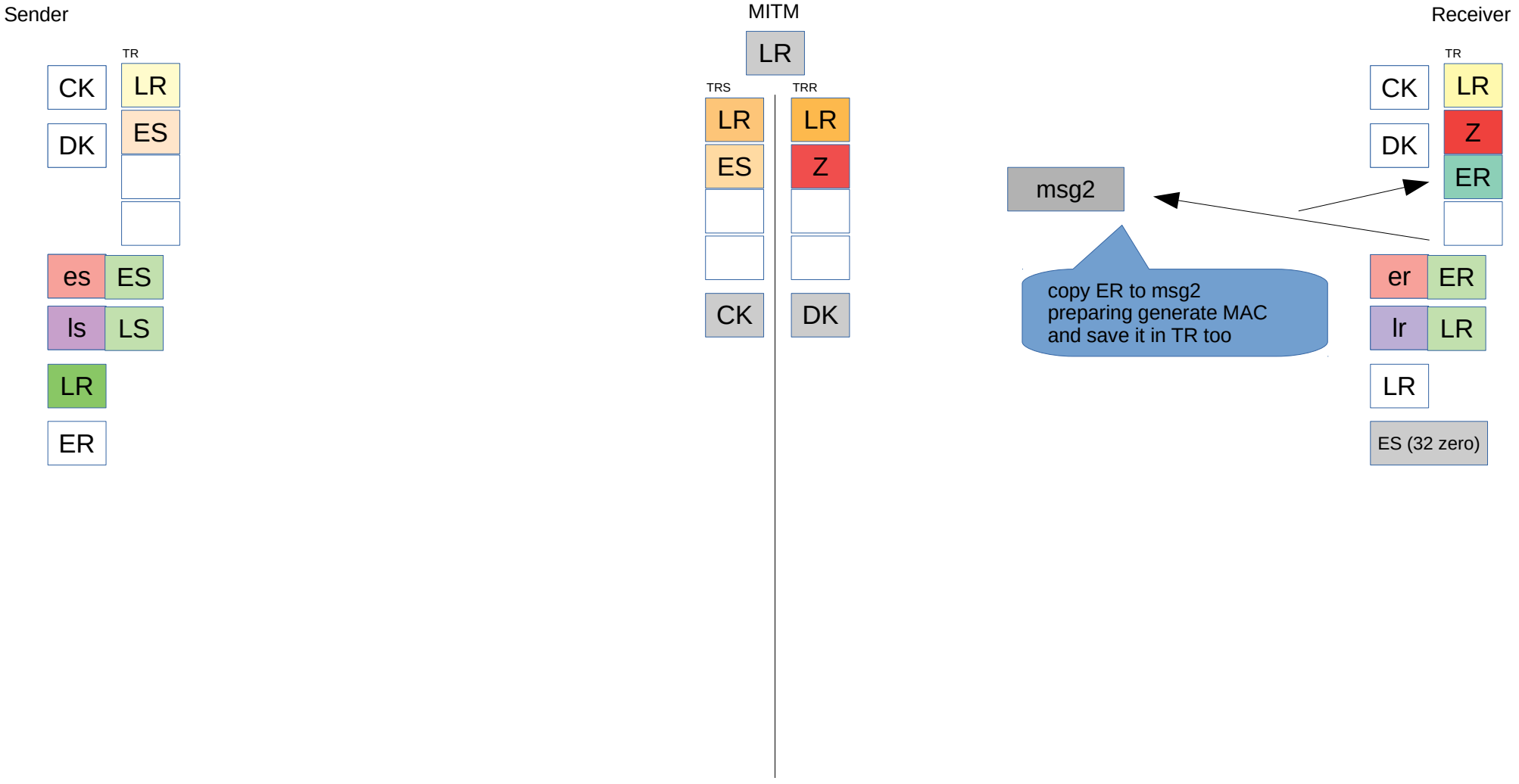

Sender
MITM
Receiver
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg1
On sender's side MITM copy LR and ES
LR
TRS
TRR
LR
ES
CK
DK
TR
CK
DK
er
ER
lr
LR
LR
ES

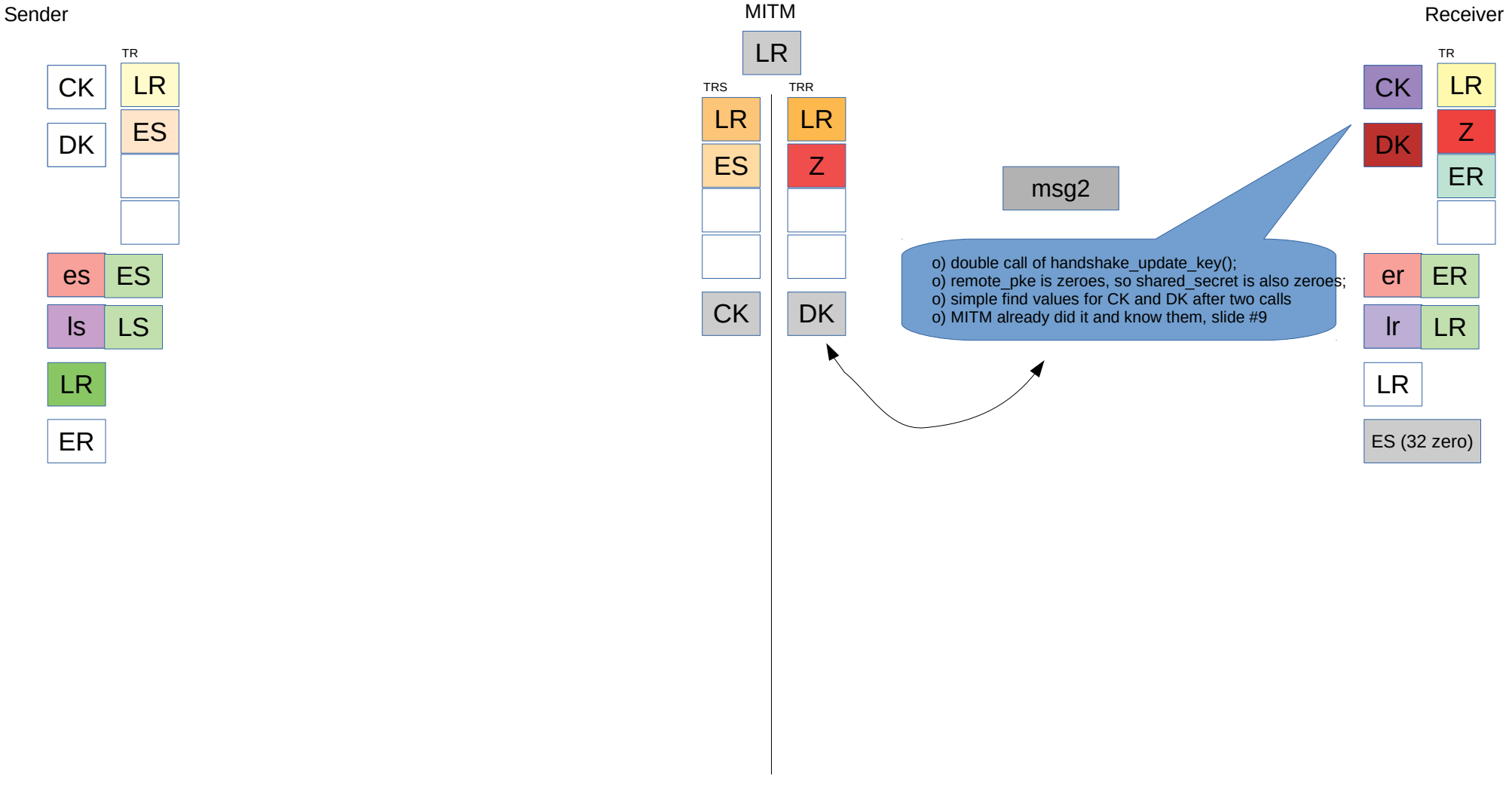

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg1
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
On receivers side MITM save the same LR and 32 zeros
32 zeroes as public key is small order key. crypto_x25519 with any private key and 32 zeroes generates 32 zeroes
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

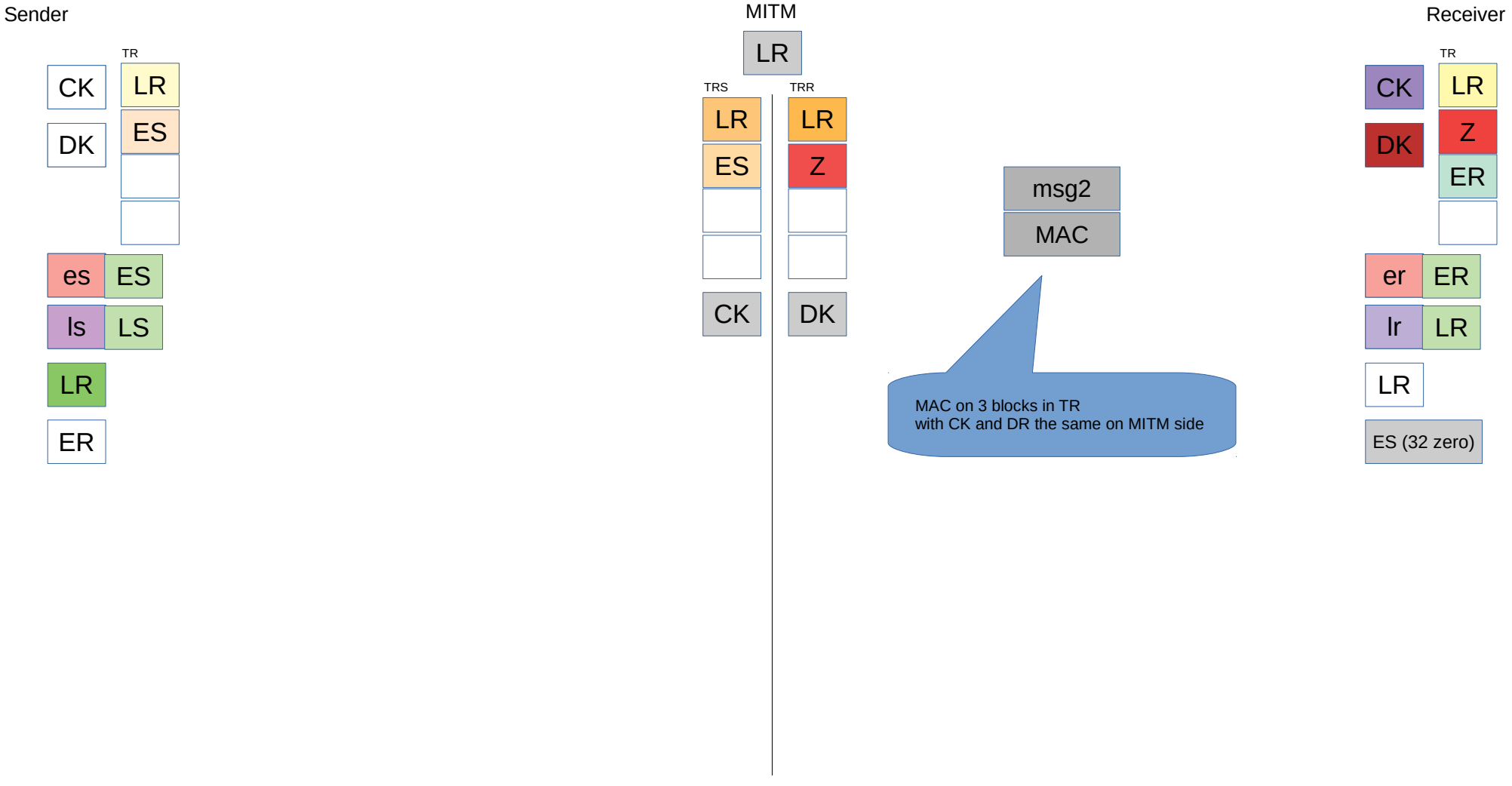

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg1
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
msg1
MITM sends 32 zeros
to receiver and
receiver accepts it
Receiver
TR
CK
DK
er
ER
lr
LR
LR
ES

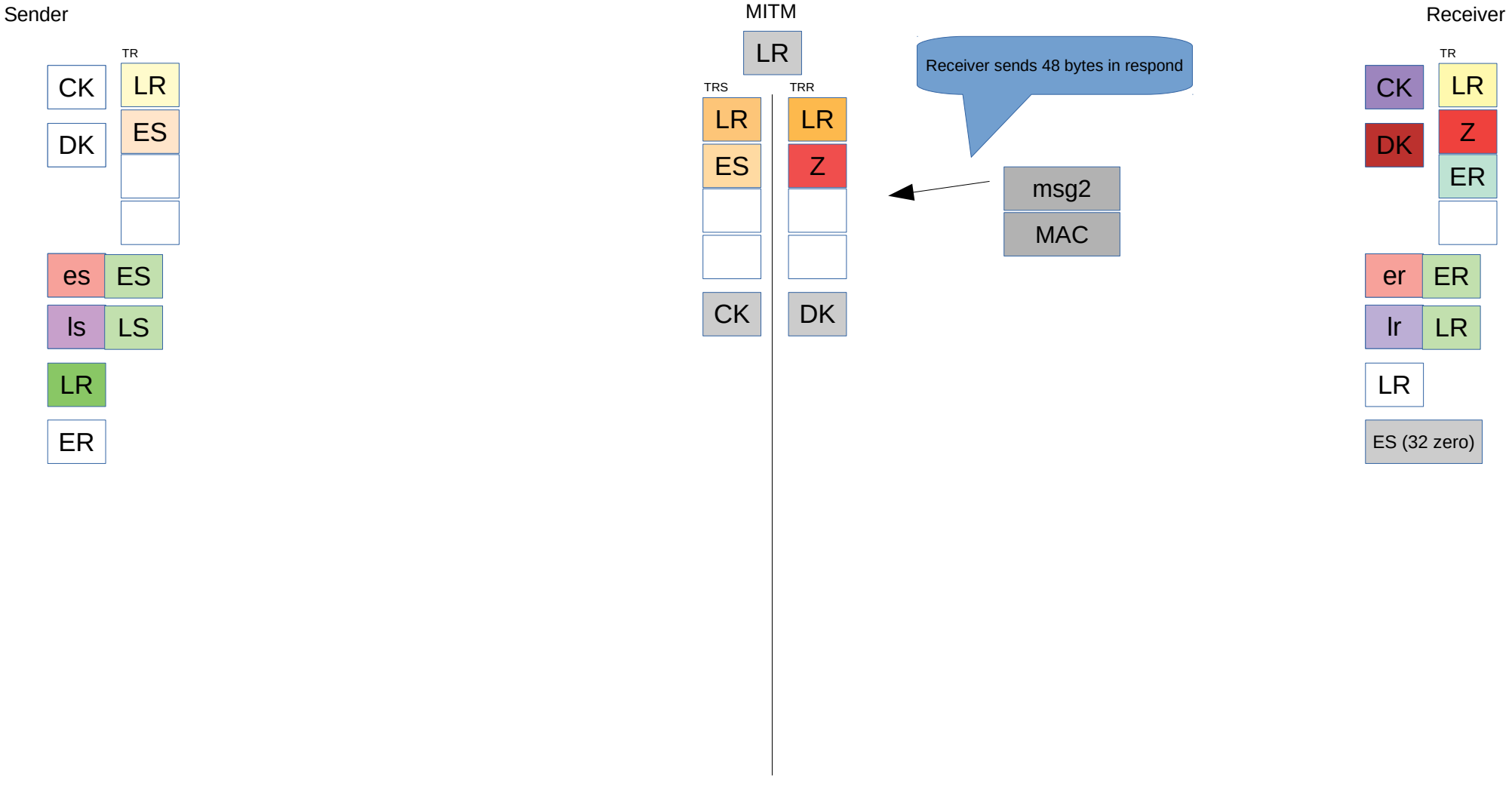
RESPOND

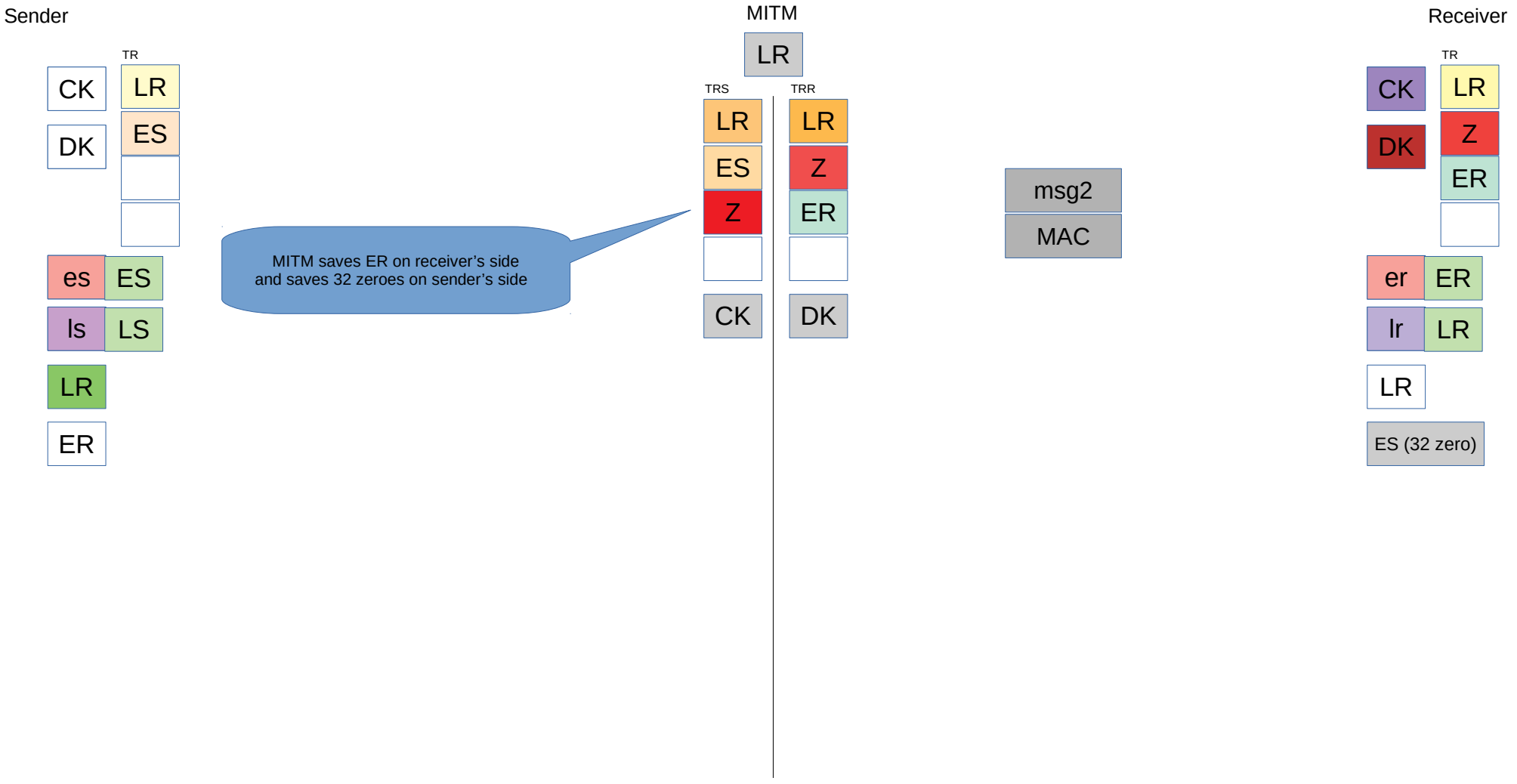

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
receiver copy LR and
ES as 32 zeroes to TR
and save 32 zeroes as remote_pke
Receiver
TR
CK
LR
DK
Z
er
ER
lr
LR
LR
ES (32 zero)

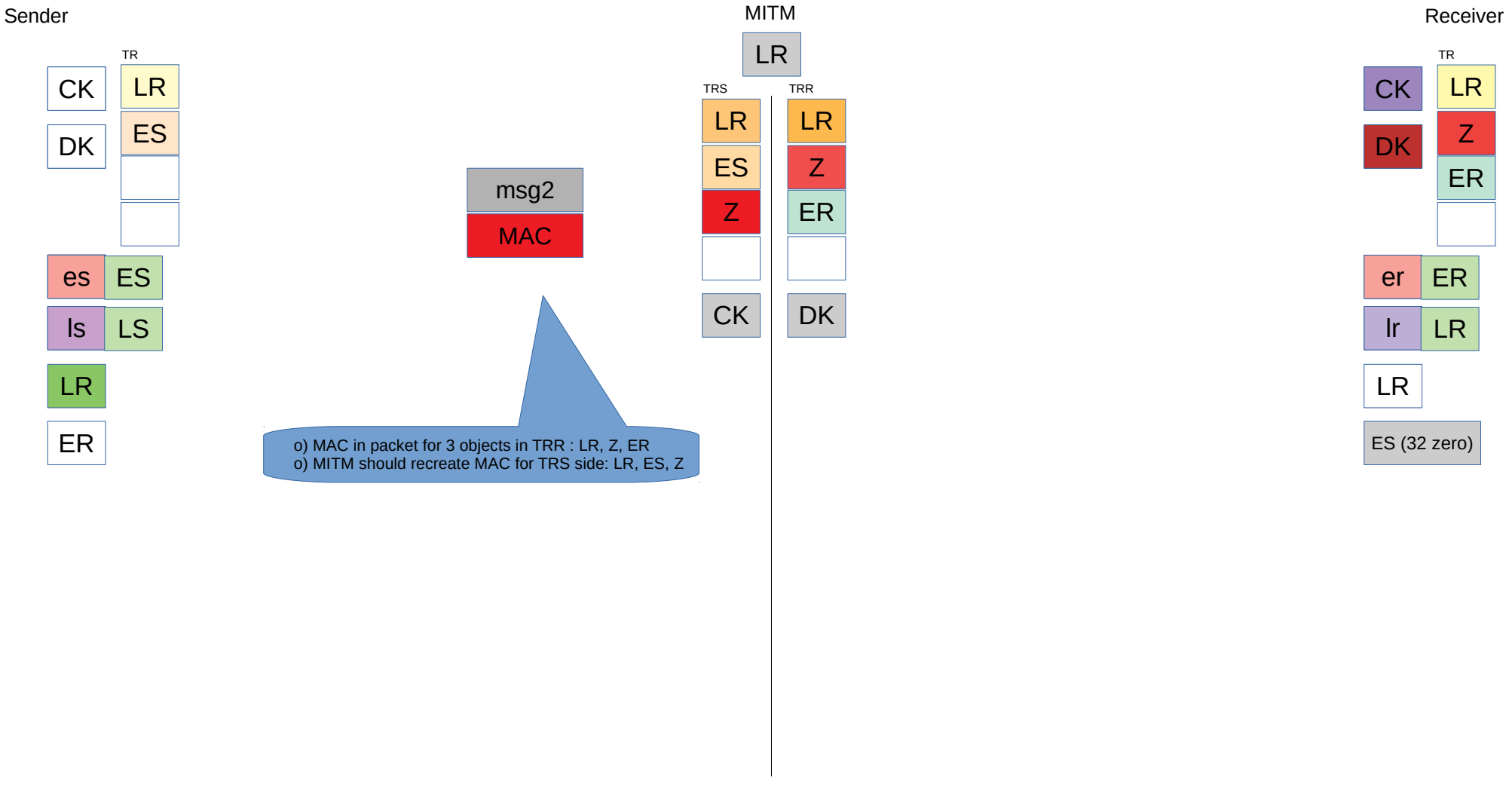

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
Receiver
TR
CK
LR
DK
Z
ER
msg2
copy ER to msg2
preparing generate MAC
and save it in TR too
er
ER
lr
LR
LR
ES (32 zero)

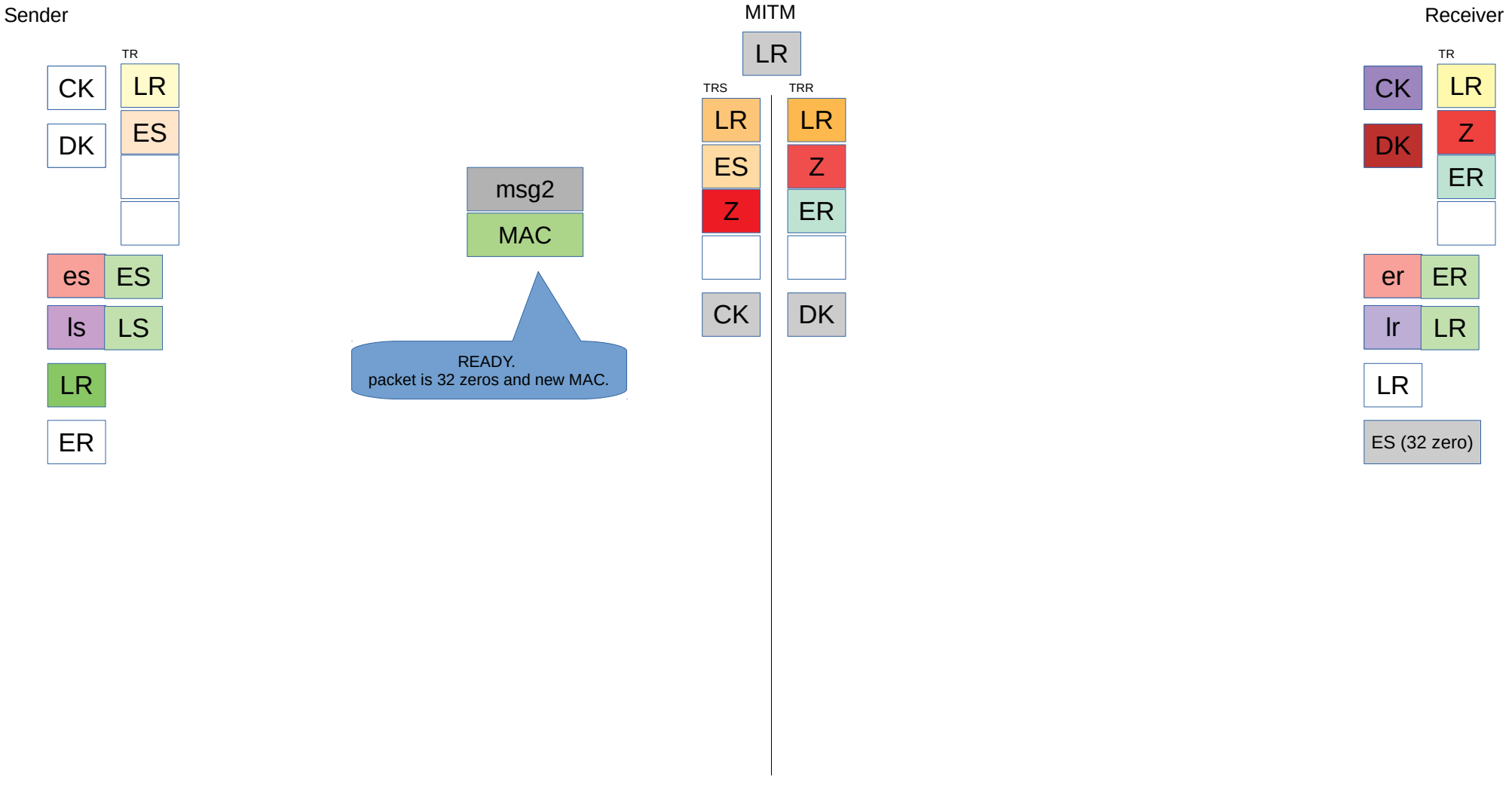

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
msg2
o) double call of handshake_update_key();
o) remote_pke is zeroes, so shared_secret is also zeroes;
o) simple find values for CK and DK after two calls
o) MITM already did it and know them, slide #9
Receiver
TR
CK
LR
DK
Z
ER
er
ER
lr
LR
LR
ES (32 zero)

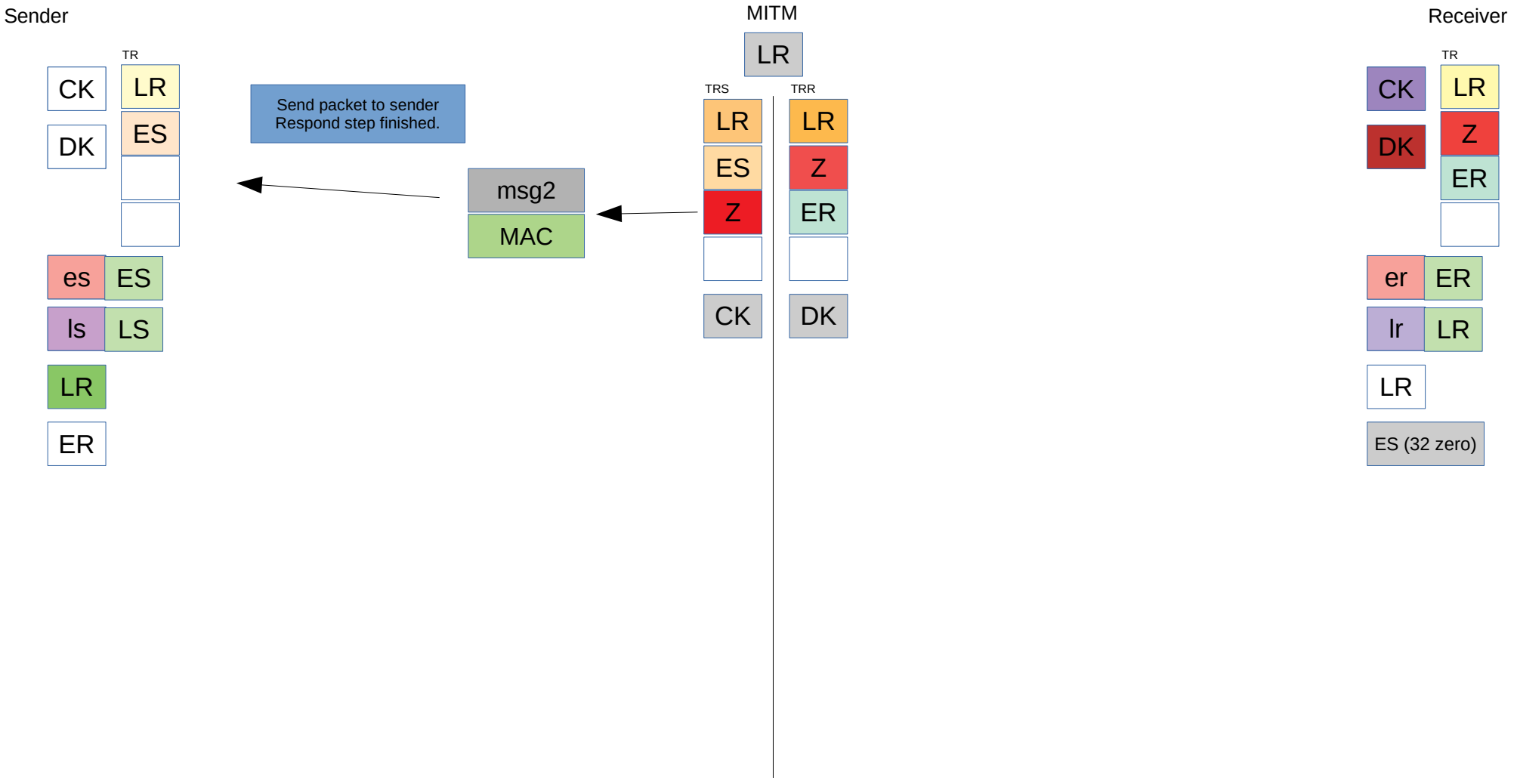

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
msg2
MAC
MAC on 3 blocks in TR
with CK and DR the same on MITM side
Receiver
TR
CK
LR
DK
Z
ER
er
ER
lr
LR
LR
ES (32 zero)


Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM
LR
TRS
TRR
LR
LR
ES
Z
CK
DK
Receiver sends 48 bytes in respond
msg2
MAC
Receiver
TR
CK
LR
DK
Z
ER
er
ER
lr
LS
LR
ES (32 zero)

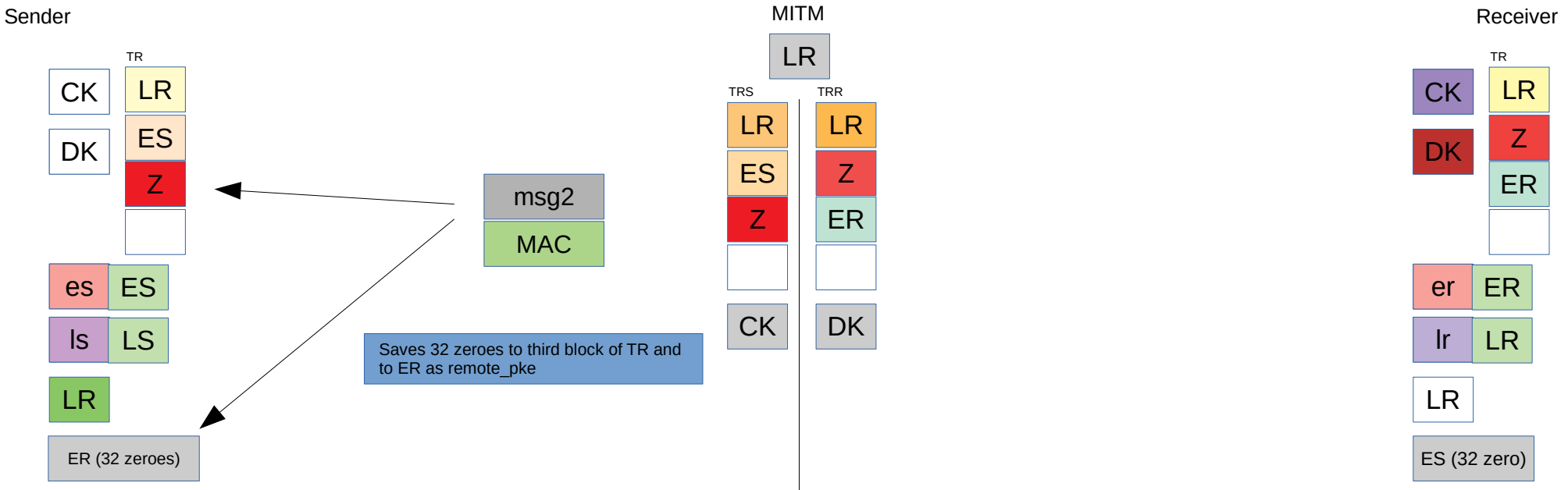

Sender
MITM
Receiver
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
MITM saves ER on receiver's side
and saves 32 zeroes on sender's side
LR
TRS
TRR
LR
LR
ES
Z
Z
ER
CK
DK
msg2
MAC
TR
CK
LR
DK
Z
ER
er
ER
lr
LS
LR
ES (32 zero)

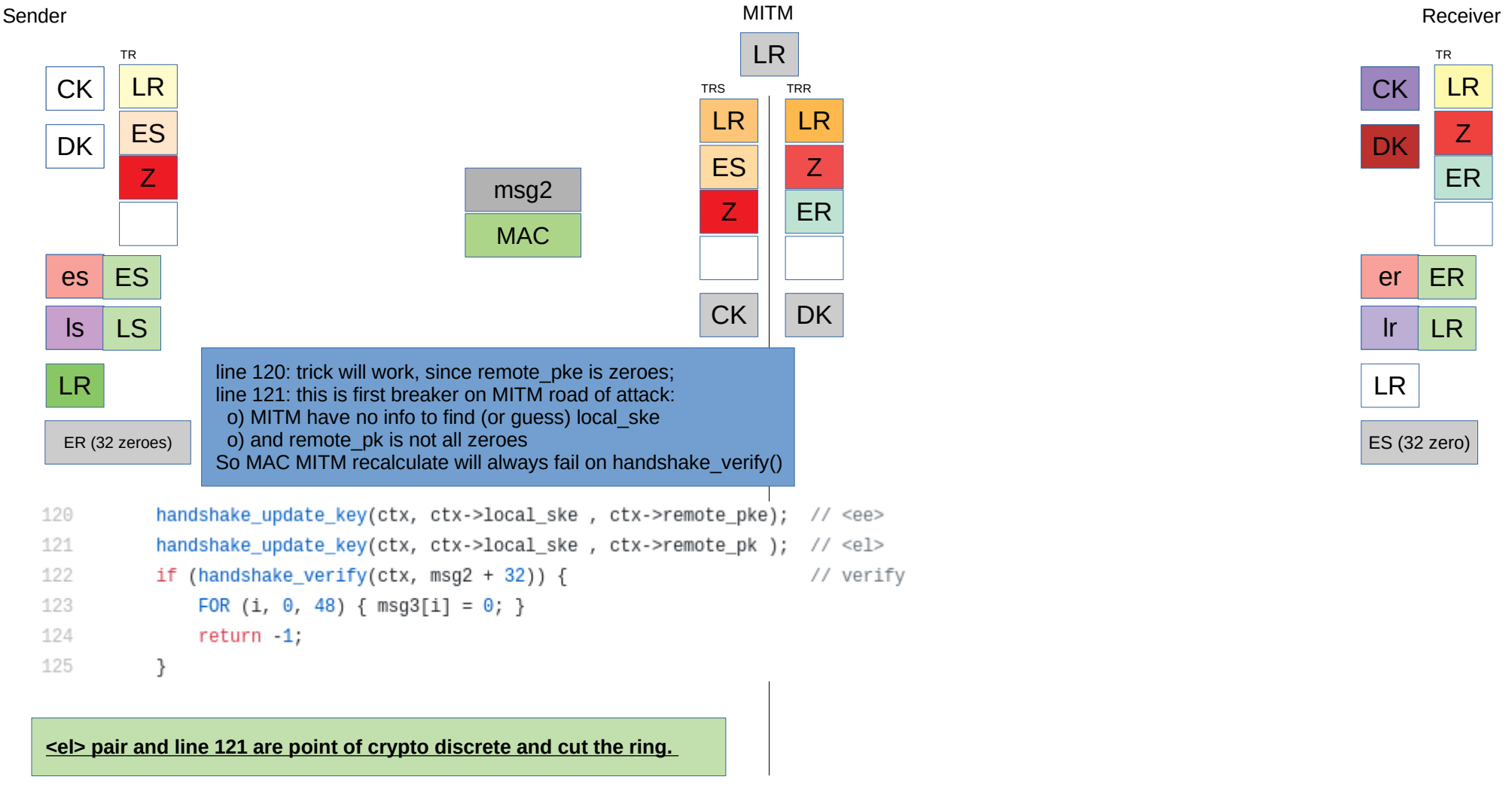

Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg2
MAC
o) MAC in packet for 3 objects in TRR : LR, Z, ER
o) MITM should recreate MAC for TRS side: LR, ES, Z
MITM
LR
TRS
TRR
LR
LR
ES
Z
Z
ER
CK
DK
Receiver
TR
CK
LR
DK
Z
ER
er
ER
lr
LS
LR
ES (32 zero)


Sender
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
msg2
MAC
READY.
packet is 32 zeros and new MAC.
MITM
LR
TRS
TRR
LR
LR
ES
Z
Z
ER
CK
DK
Receiver
TR
CK
LR
DK
Z
ER
er
ER
lr
LS
LR
ES (32 zero)


Sender
MITM
Receiver
TR
CK
LR
DK
ES
es
ES
ls
LS
LR
ER
Send packet to sender
Respond step finished.
msg2
MAC
LR
TRS
TRR
LR
LR
ES
Z
Z
ER
CK
DK
TR
CK
LR
DK
Z
ER
er
ER
lr
LR
LR
ES (32 zero)

CONFIRM


Sender
MITM
Receiver
TR
CK
LR
DK
ES
Z
es
ES
ls
LS
LR
ER (32 zeroes)
msg2
MAC
Saves 32 zeroes to third block of TR and to ER as remote_pke
LR
TRS
TRR
LR
LR
ES
Z
Z
ER
CK
DK
TR
CK
LR
DK
Z
ER
er
ER
lr
LR
LR
ES (32 zero)


Sender
MITM
Receiver
TR
CK
LR
DK
ES
Z
es
ES
ls
LS
LR
ER (32 zeroes)
msg2
MAC
TRS
TRR
LR
ES
Z
CK
LR
Z
ER
DK
TR
CK
LR
DK
Z
ER
er
ER
lr
LR
LR
ES (32 zero)
line 120: trick will work, since remote_pke is zeroes;
line 121: this is first breaker on MITM road of attack:
o) MITM have no info to find (or guess) local_ske
o) and remote_pk is not all zeroes
So MAC MITM recalculate will always fail on handshake_verify()
120 handshake_update_key(ctx, ctx->local_ske , ctx->remote_pke); // <ee>
121 handshake_update_key(ctx, ctx->local_ske , ctx->remote_pk); // <el>
122 if (handshake_verify(ctx, msg2 + 32)) { // verify
123 FOR (i, 0, 48) { msg3[i] = 0; }
124 return -1;
125 }
<el> pair and line 121 are point of crypto discrete and cut the ring.

The end.

Thank you for your attention.