Title: Quantum Information Review - Lecture 12

Date: Mar 01, 2011 10:10 AM

URL: http://pirsa.org/11030001

Abstract:

Fault Tolerance

- Fault tolerant circuits are designed to avoid error propagation.

Transversal gates

Fault Tolerance

- Fault tolerant circuits are designed to avoid error propagation.

Transversal gates

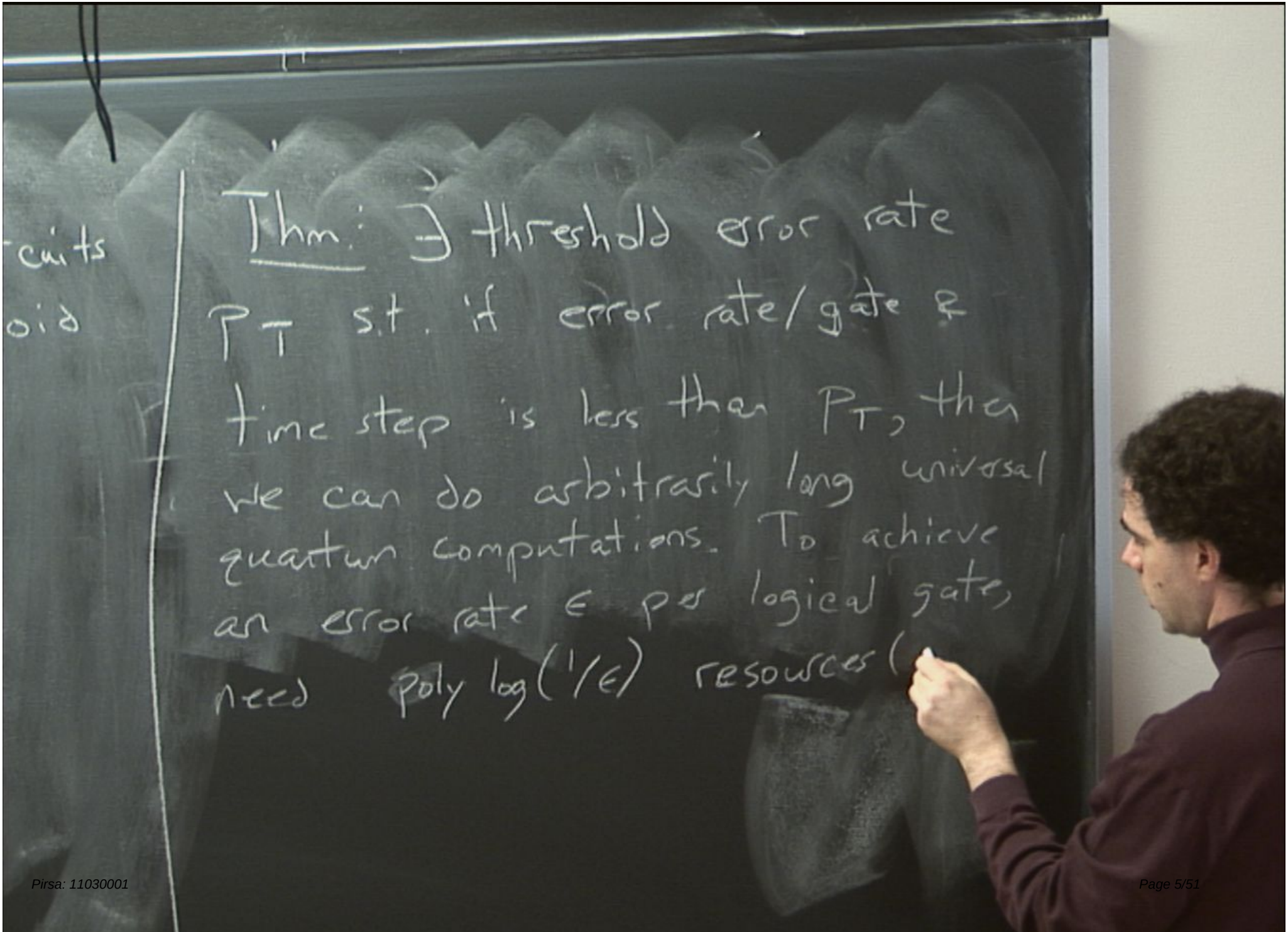
7-qubit code is particularly good.

Fault Tolerance

- Fault tolerant circuits are designed to avoid error propagation.

Transversal gates

7-qubit code is particularly good.


cuits
oid
Thm: ∃ threshold error rate
p_T s.t. if error rate/gate &
time step is less than p_T, then
we can do arbitrarily long universal
quantum computations. To achieve
an error rate ε per logical gate,
need poly log(1/ε) resources (

cuits

oid

Thm: $\exists$ threshold error rate p_T s.t. if error rate/gate & time step is less than p_T, then we can do arbitrarily long universal quantum computations. To achieve an error rate ϵ per logical gate, need $\text{poly}\log(1/\epsilon)$ resources (multiplicative factor of extra qubits, time steps,)

cuits

oid

Thm: $\exists$ threshold error rate p_T s.t. if error rate/gate & time step is less than p_T, then we can do arbitrarily long universal quantum computations. To achieve an error rate ϵ per logical gate, need poly log$(1/\epsilon)$ resources (multiplicative factor of extra qubits, time steps, ...)

Threshold ~ 5%

2D ~ 0.7%

cuits

oid

Thm: $\exists$ threshold error rate P_T s.t. if error rate/gate & time step is less than P_T, then we can do arbitrarily long universal quantum computations. To achieve an error rate ϵ per logical gate, need Poly log$(1/\epsilon)$ resources (multiplicative factor of extra qubits, time steps, ...)

Threshold ~ 5% Proof ~ 0.1%

2D ~ 0.7%

cuits

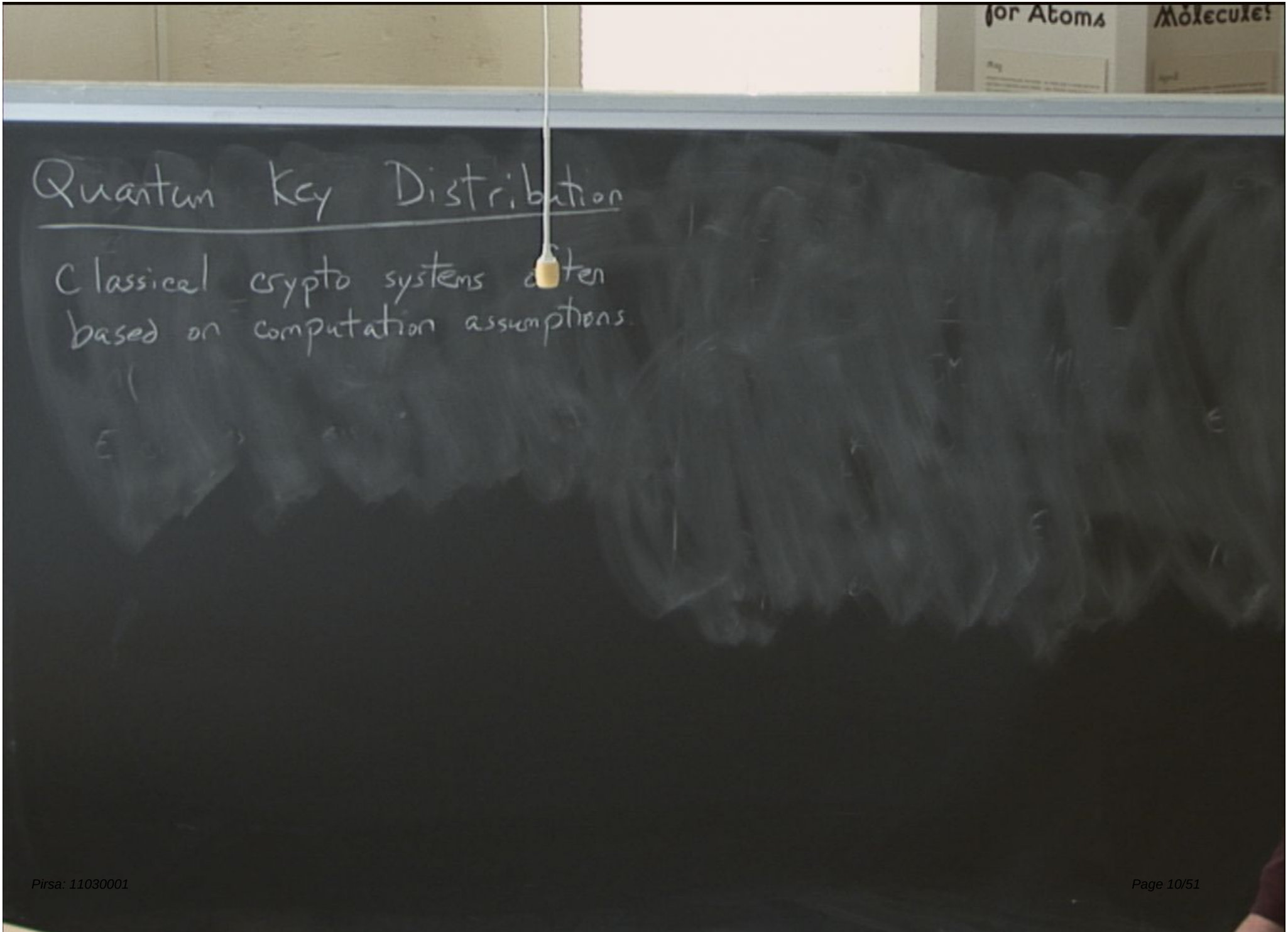
oid

Thm: $\exists$ threshold error rate p_T s.t. if error rate/gate & time step is less than p_T, then we can do arbitrarily long universal quantum computations. To achieve an error rate ϵ per logical gate, need poly log$(1/\epsilon)$ resources (multiplicative factor of extra qubits, time steps, ...)

Threshold ~ 5%

2D ~ 0.7%

Proof ~ 0.1%


for Atoms
Molecules
Quantum Key Distribution
Classical crypto systems often
based on computation assumptions.

Quantum Key Distribution

Classical crypto systems often based on computational assumptions.

Quantum Key Distribution

Classical crypto systems often based on computational assumptions.

One-time pad: Alice & Bob share key k - n random bits

Quantum Key Distribution

Classical crypto systems often based on computational assumptions.

One-time pad: Alice & Bob share
key k: n random bits
Encrypt m (n bits long): $m \oplus k$
(bitwise XOR)

Quantum Key Distribution

Classical crypto systems often based on computational assumptions.

One-time pad: Alice & Bob share
key k: n random bits
Encrypt m (n bits long): $m \oplus k$ ← Completely random to Eve
(bitwise XOR)

Quantum Key Distribution

Classical crypto systems often based on computational assumptions.

One-time pad: Alice & Bob share
key k: n random bits
Encrypt m (n bits long): $m \oplus k = e$ ← Completely random to Eve
(bitwise XOR)
Decrypt: $e \oplus k = m \oplus k \oplus k = m$

Key Distribution

crypto systems often computational assumptions.

e pad: Alice & Bob share
k: n random bits
pt m (n bits long): m⊕k=e
(bitwise XOR)
ypt: e⊕k = m⊕k⊕k = m

Completely random to Eve

(m⊕k)⊕(m'⊕k)
= m⊕m'

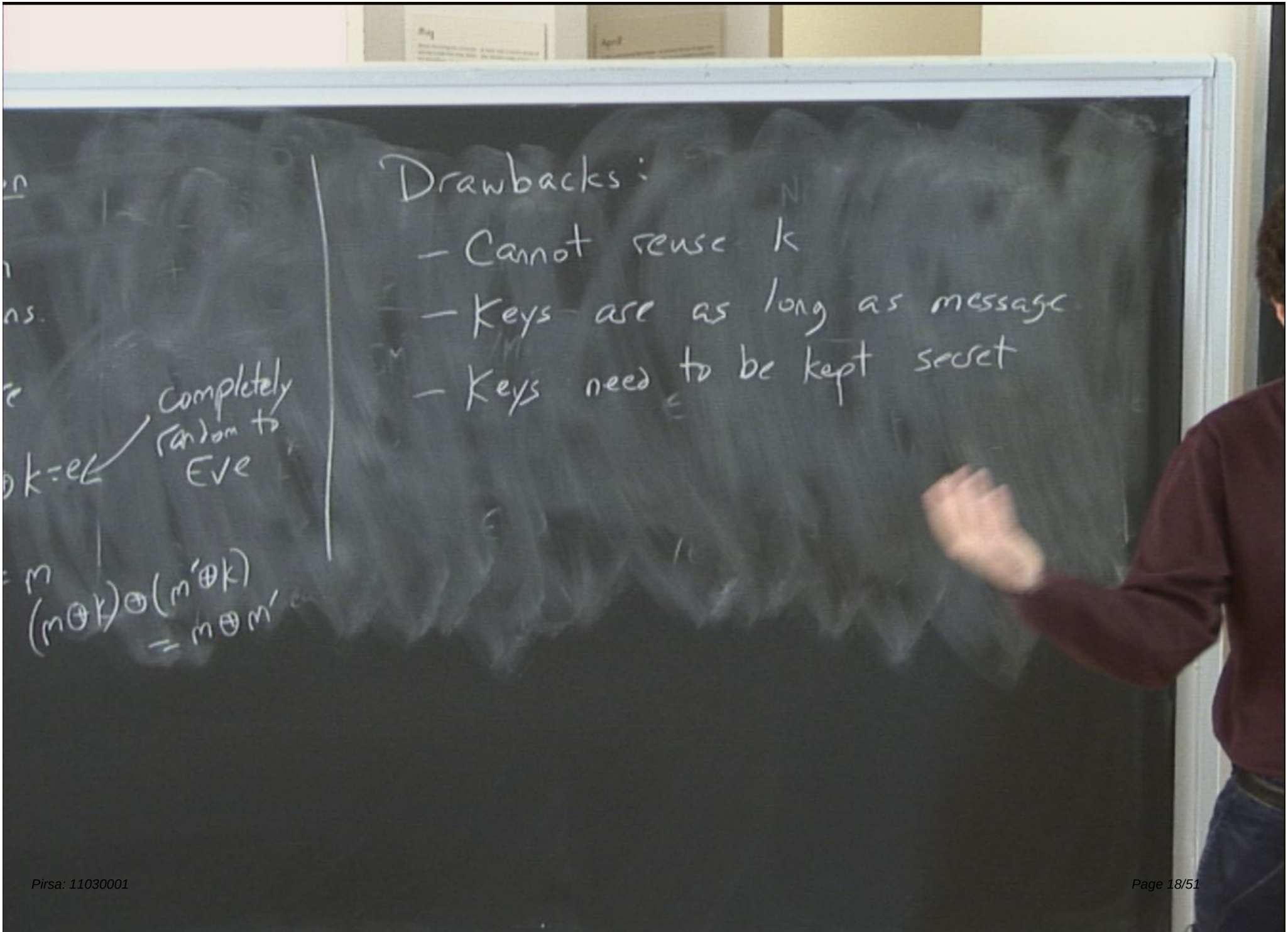
Drawbacks:
- Cannot reuse k

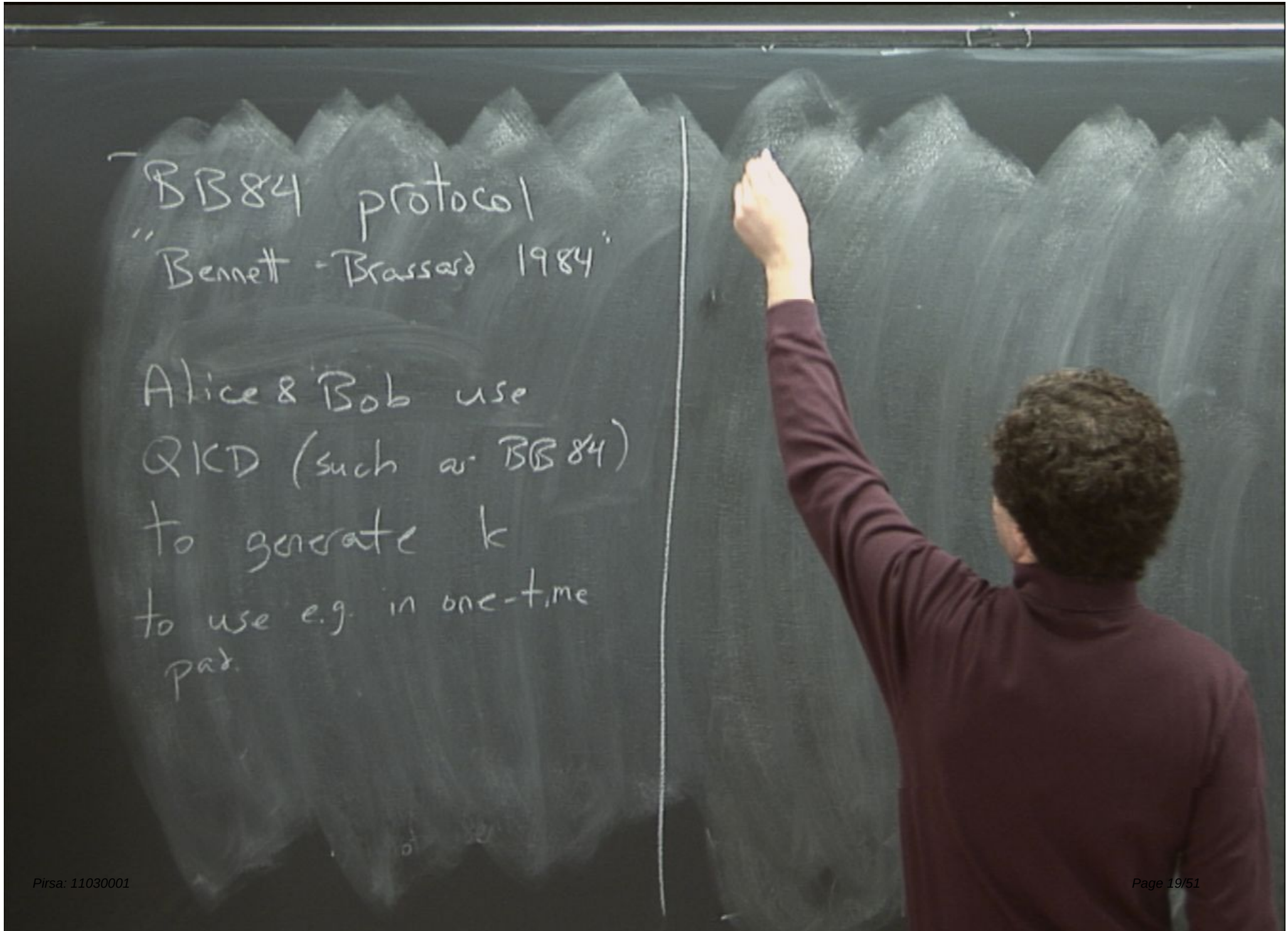
Drawbacks:

- Cannot reuse k
- Keys are as long as message

Completely random to Eve

$(m \oplus k) \oplus (m' \oplus k) = m \oplus m'$


Drawbacks:
- Cannot reuse k
- Keys are as long as message
- Keys need to be kept secret
Completely random to Eve
(m⊕k)⊕(m'⊕k)
= m⊕m'


BB84 protocol
"Bennett - Brassard 1984"
Alice & Bob use
QKD (such as BB84)
to generate k
to use e.g. in one-time
pad.

1. Alice generate random bit pairs (q_i, r_i).

q_i	r_i
0	0
0	1
1	0
1	1

2. q_i specifies basis
   r_i specifies state

1984

e

BB 84)

-time

① Alice generate random bit pairs (q_i, r_i).

② q_i specifies basis
r_i specifies state

q_i	r_i	
0	0	$\|0\rangle$
0	1	$\|1\rangle$
1	0	$\|+\rangle$
1	1	$\|-\rangle$

Fo

- F

ase

erro

Tr

$|1\rangle$

~~$|0\rangle$~~

$|0\rangle$ —

7-q

part

1984

e

BB84)

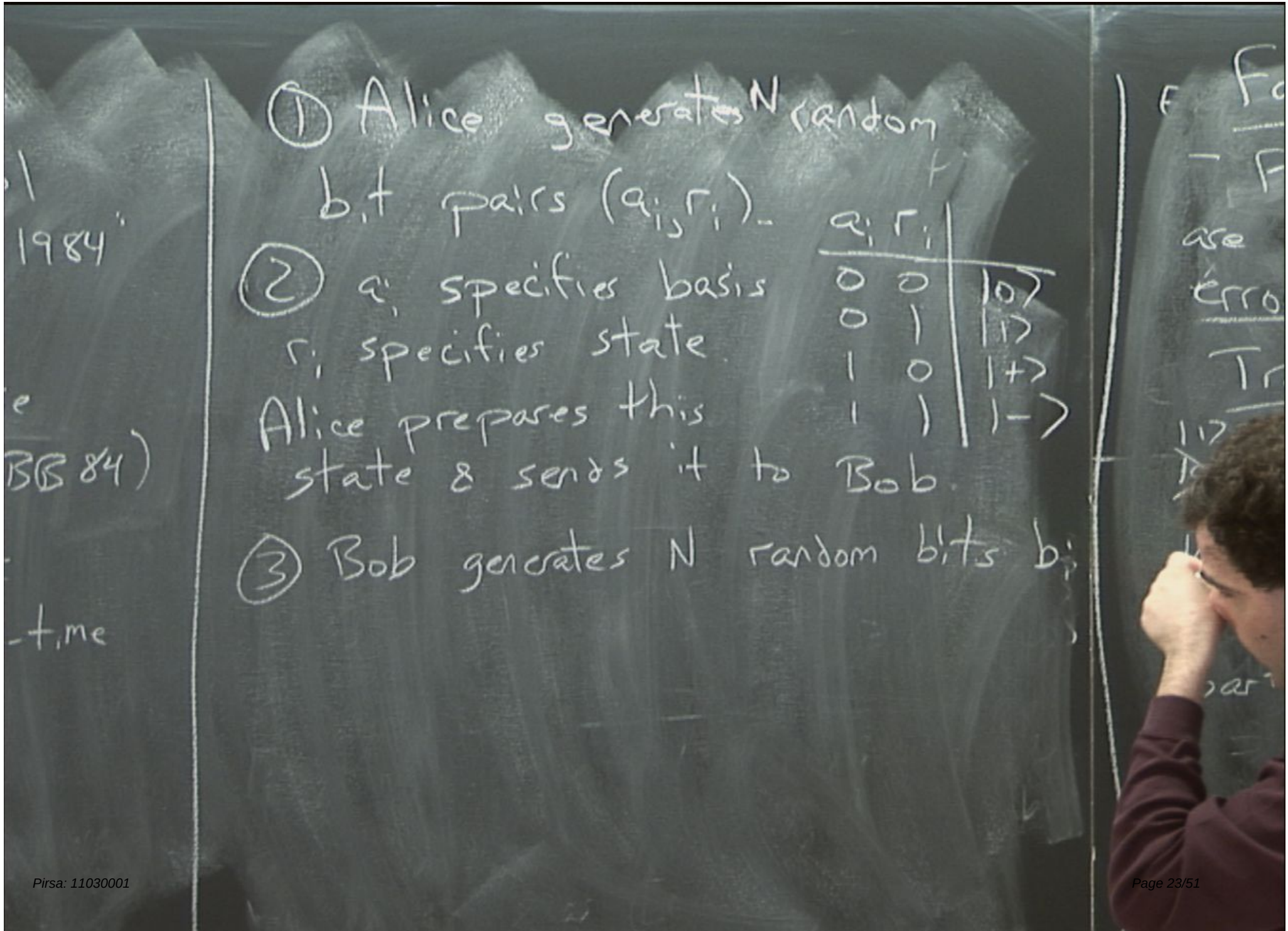
-time

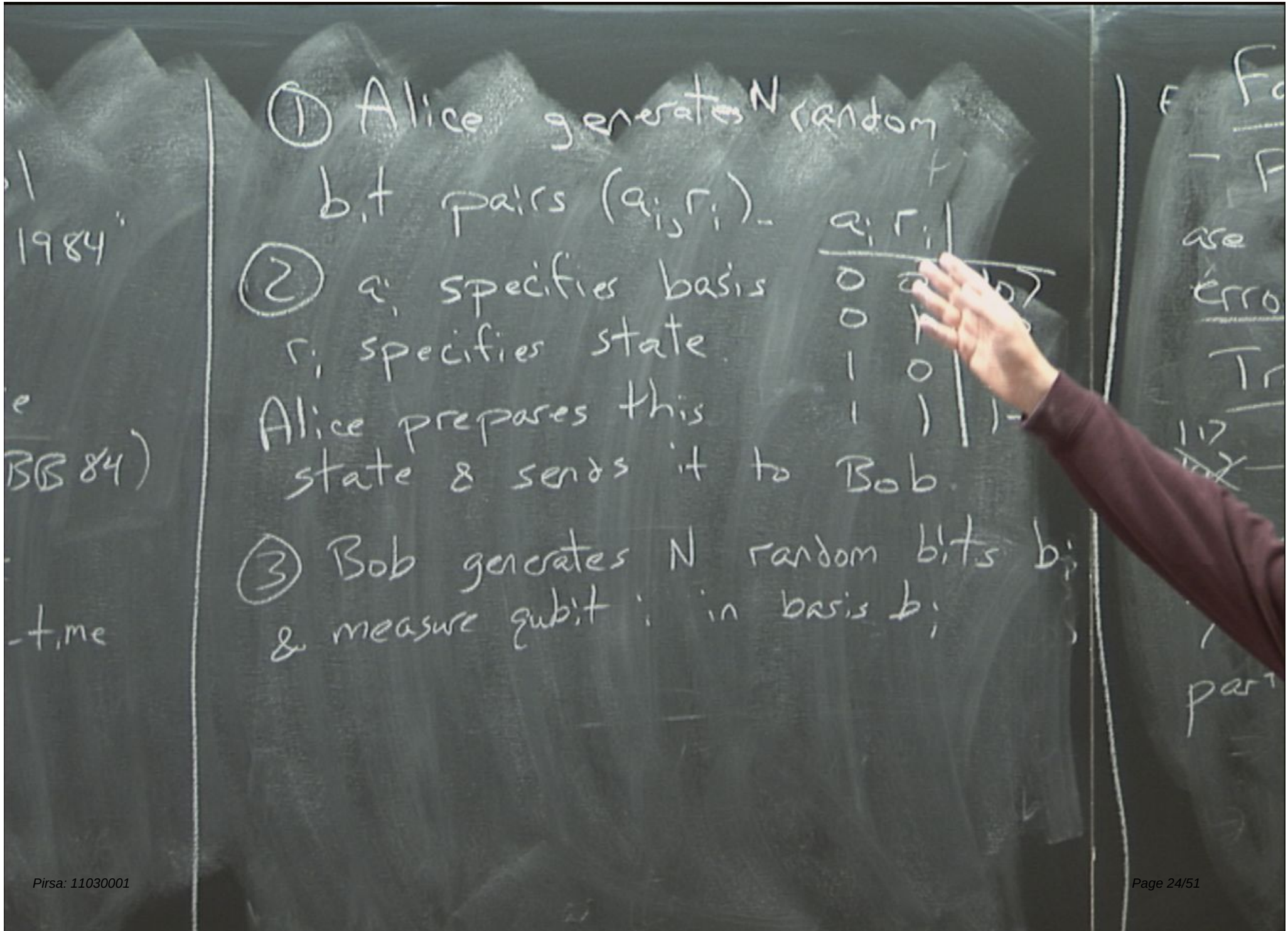
① Alice generate random bit pairs (q_i, r_i).

② q_i specifies basis
r_i specifies state.
Alice prepares this state & sends it to Bob.

q_i	r_i	
0	0	$\|0\rangle$
0	1	$\|1\rangle$
1	0	$\|+\rangle$
1	1	$\|-\rangle$

Fo
- F
ase
erro
Tr
11⟩
10⟩
10⟩
7-q
part


1984
BB 84)
-time
① Alice generates N random bit pairs (a_i, r_i).
② a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to Bob.
③ Bob generates N random bits b
a_i r_i
0 0 |0>
0 1 |1>
1 0 |+>
1 1 |->
Fo
- F
ase
erro
Tr
117


1984
BB84)
-time
1. Alice generates N random bit pairs (a_i, r_i).
2. a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to Bob
3. Bob generates N random bits b_i & measure qubit i in basis b_i
a_i r_i
0 0
0 1
1 0
1 1
case
erro
par

1984

e

BB84)

-time

① Alice generates N random bit pairs (a_i, r_i).

② a_i specifies basis
r_i specifies state.
Alice prepares this state & sends it to Bob.

a_i	r_i	
0	0	$\|0\rangle$
0	1	$\|1\rangle$
1	0	$\|+\rangle$
1	1	$\|-\rangle$

③ Bob generates N random bits b_i & measure qubit i in basis b_i

Fo

- F

ase

érro

Tr

117

10⟩

7-q

part

1984

e

BB84)

-time

① Alice generates N random bit pairs (a_i, r_i).

② a_i specifies basis
r_i specifies state.
Alice prepares this state & sends it to Bob.

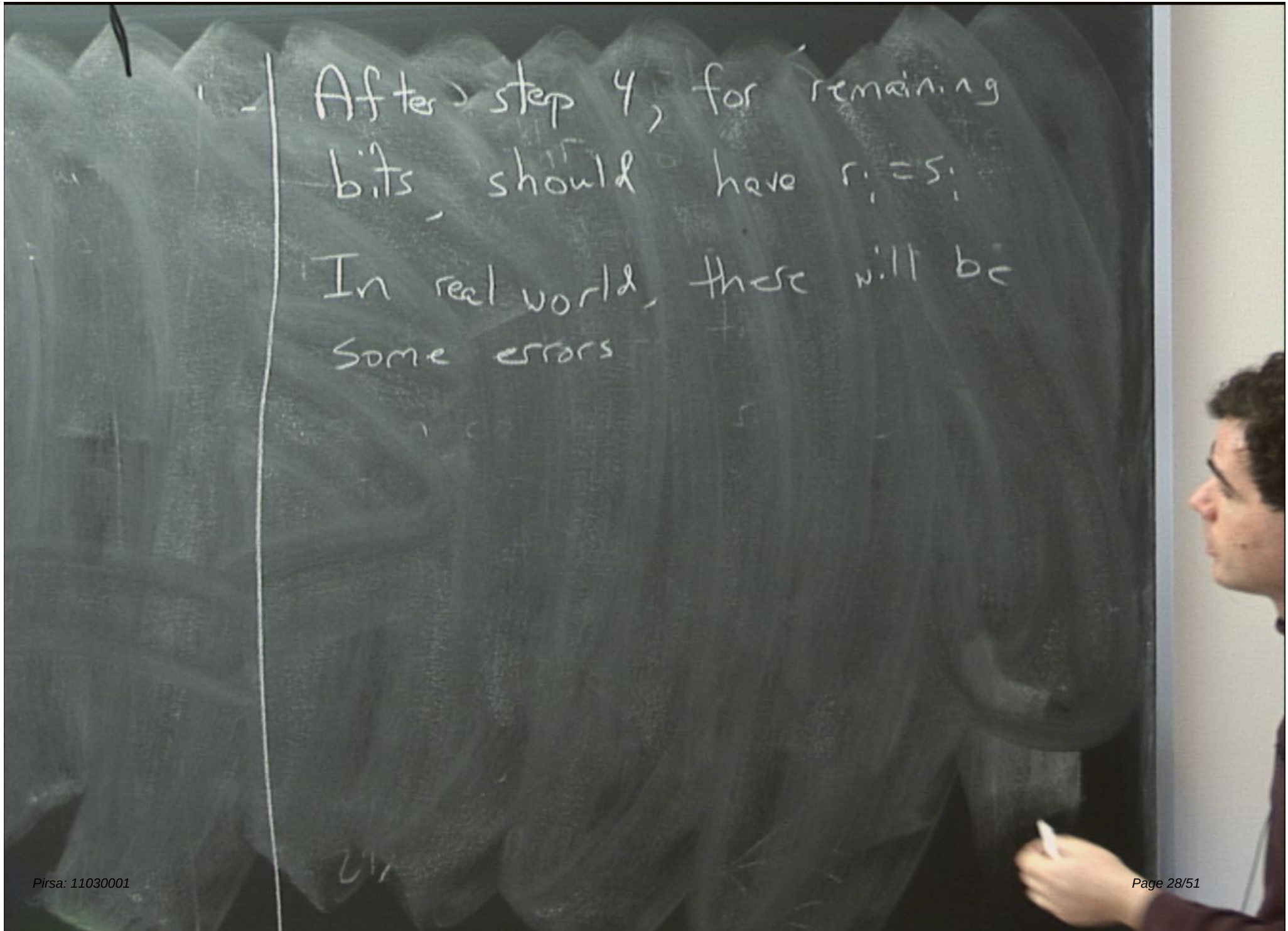
a_i	r_i	
0	0	$\|0\rangle$
0	1	$\|1\rangle$
1	0	$\|+\rangle$
1	1	$\|-\rangle$

③ Bob generates N random bits b_i & measure qubit i in basis b_i, getting result s_i

④ Alice & Bob announce publicly a_i & b_i, $\forall i$
If $a_i = b_i$, keep bit i
If $a_i \neq b_i$, discard bit i

Fo
- F
ase
erro
Tr
|1⟩
|0⟩
|0⟩ —
7-q
part

After step 4, for remaining bits should have $r_i = s_i$


After step 4, for remaining bits, should have $r_i = s_i$
In real world, there will be some errors

- After step 4, for remaining bits, should have $r_i = 5_i$

In real world, there will be some errors.

Eavesdropper (Eve) can cause errors.

- In fact, Eve must cause errors

After step 4, for remaining bits, should have $r_i = s_i$

In real world, there will be some errors.

Eavesdropper (Eve) can cause errors.

- In fact, Eve must cause errors
- e.g. Alice sends $|0\rangle$, Eve doesn't know a_i

After step 4, for remaining bits, should have $r_i = s_i$

In real world, there will be some errors.

Eavesdropper Eve can cause errors.

- In fact, Eve must cause errors
- e.g. Alice sends $|0\rangle$, Eve doesn't know a_i - suppose she guesses $|+\rangle/|-\rangle$ basis & measures; she might get $|+\rangle$

After step 4, for remaining bits, should have $r_i = s_i$

In real world, there will be some errors.

Eavesdropper Eve can cause errors.

- In fact, Eve must cause errors

- e.g. Alice sends $|0\rangle$, Eve doesn't know a_i - suppose she guesses $|+\rangle/|-\rangle$ basis & measures; she might get $|+\rangle$, and sends that to Bob.

After step 4, for remaining bits, should have $r_i = s_i$

In real world, there will be some errors.

Eavesdropper Eve can cause errors.

- In fact, Eve must cause errors
- e.g. Alice sends $|0\rangle$, Eve doesn't know a_i - suppose she guesses $|+\rangle/|-\rangle$ basis & measures; she might get $|+\rangle$, and sends that to Bob. When B&B keep the bit, Bob measures $|0\rangle/|1\rangle$ basis, gets $|1\rangle$ half the time.

After step 4, for remaining bits, should have $r_i = s_i$

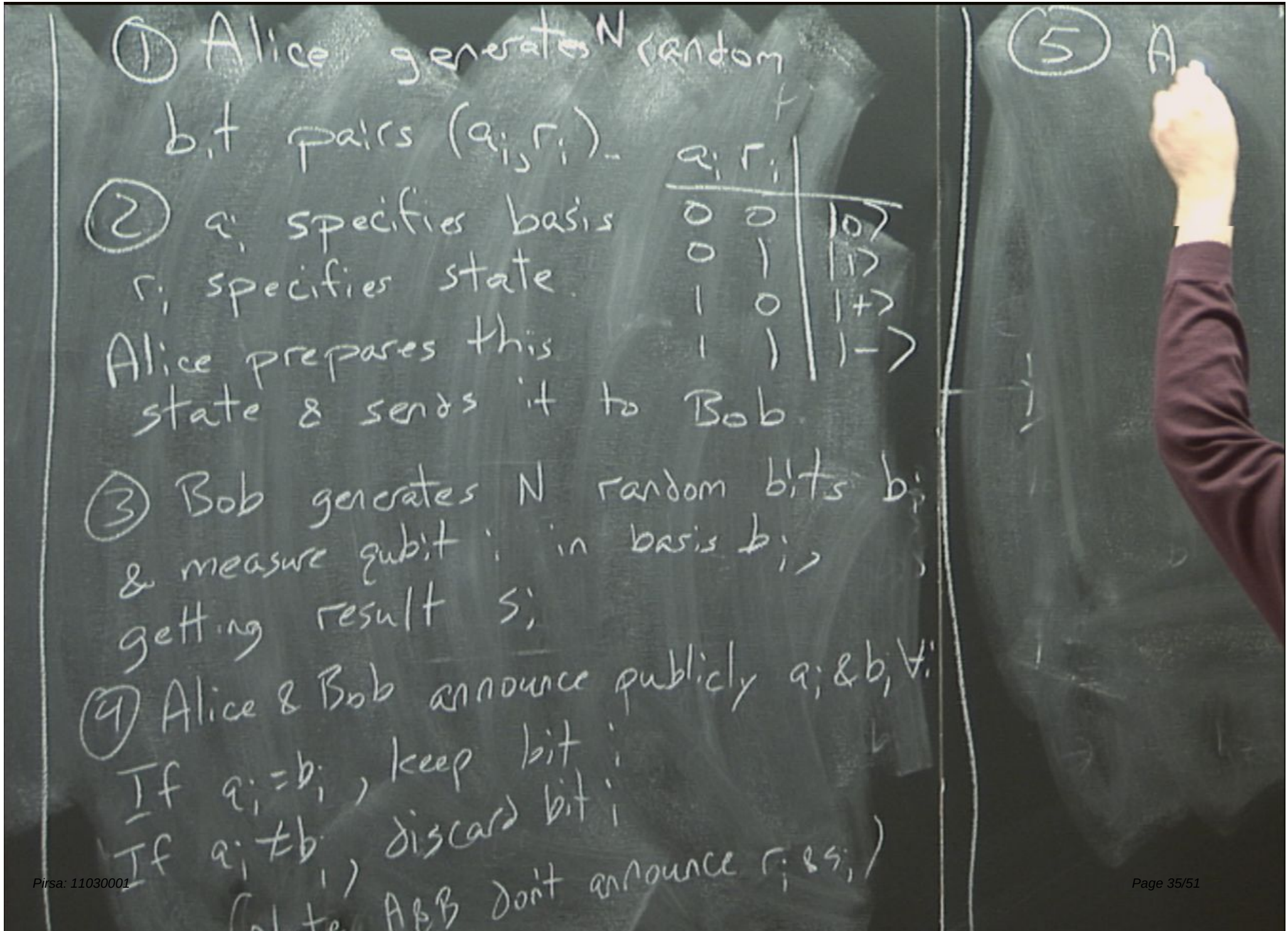
In real world, there will be some errors.

Eavesdropper Eve can cause errors.

- In fact, Eve must cause errors

- e.g. Alice sends $|0\rangle$, Eve doesn't know a_i - suppose she guesses $|+\rangle/|-\rangle$ basis & measures; she might get $|+\rangle$, and sends that to Bob. When BBB keep the bit, Bob measures $|0\rangle/|1\rangle$ basis, gets $|1\rangle$ half the time.

$\Rightarrow$ 25% bit error rate


① Alice generates N random bit pairs (a_i, r_i).
② a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to Bob
a_i r_i
0 0 |0>
0 1 |1>
1 0 |+>
1 1 |->
③ Bob generates N random bits b_i & measure qubit i in basis b_i, getting result s_i
④ Alice & Bob announce publicly a_i & b_i, $\forall i$
If $a_i = b_i$, keep bit i
If $a_i \neq b_i$, discard bit i
Alice & Bob don't announce r_i & s_i)
⑤ A

(5) A&B choose some subset of remaining bits & announce #'s for those bits.

After > step
bits, shou

In real wor
Some error
Eavesdropper
- In fact, E
- e.g. Alice s
know a; - s
basis & measure
and sends that
the bit, Bob
½ half the

(5) A&B choose some subset of remaining bits & announce bases for those bits. Calculate error rate on these test bits. If error rate is too high ($\gtrsim 20\%$), they abort (discard key).

After step
bits, shou

In real wor
Some error

Eavesdropper

- In fact,

- e.g. Alice s
know a; - s
basis & measure
and sends that
the bit, Bob
½ half the

~ 25%

⑤ A&B choose some subset of remaining bits & announce bits for those bits. Calculate error rate on these test bits.
If error rate is too high ($\gtrsim 20\%$), they abort (discard key).

⑥ Do error correction by revealing some parities of key bits (must discard some key bits).

After step
bits, shou

In real wor
Some error

Eavesdropper

- In fact,

- e.g. Alice s
know a; - s
basis & measure
and sends that
the bit, Bob
½ half the

~ 25%

|0>
|1>
|+>
|->

s b

; & b, ∀

(5) A&B choose some subset of remaining bits & announce bits for those bits. Calculate error rate on these test bits. If error rate is too high (≳ 20%), they abort (discard key).

(6) Do error correction by revealing some parities of key bits (must discard some key bits).

(7) Privacy amplification:

After step
bits, shou

In real wor
Some error

Eavesdropper

- In fact,

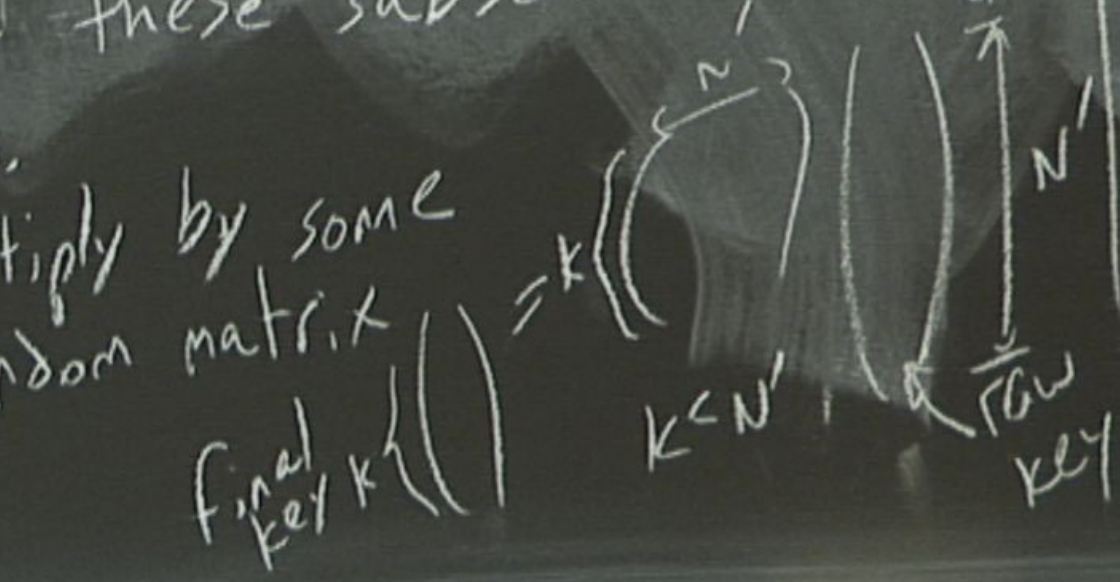
- e.g. Alice s
know a; - s
basis & measure
and sends that
Bob
half

25%

⑥ Do error correction by revealing some parities of key bits (must discard some key bits).

⑦ Privacy amplification: A & B choose random subsets & use the parities of these subsets as final key bits.

i.e. multiply by some random matrix



dom

r_i

0	0	$\|0\rangle$
0	1	$\|1\rangle$
	0	$\|+\rangle$
1	1	$\|-\rangle$

Bob.

m bits b_i

is $b_i\rangle$

icly a_i & b_i $\forall i$

nce r_i & s_i)

(5) A&B choose some subset of remaining bits & announce r_i & s_i for those bits. Calculate error rate on these test bits.

If error rate is too high ($\gtrsim 20\%$), they abort (discard key).

(6) Do error correction by revealing some parities of key bits (must discard some key bits).

(7) Privacy amplification: A&B choose random subsets & use the parities of these subsets as final key bits.

i.e. multiply by some random matrix

$$\text{final key } k\left\{\left(\;\right)\right. = k\left\{\left(\;\overset{N'}{\;}\;\right)\right.\left(\;\right)\uparrow N' \quad \text{raw key}$$

$k < N'$

After step
bits shoul

In real worl
Some errors
Eavesdropper
– In fact, E
– e.g. Alice se
know a_i – su
basis & measures
and sends that
the bit, Bob m
$|1\rangle$ half the t
$\rightsquigarrow$ 25% bit

BB84 protocol

"Bennett - Brassard 1984"

Alice & Bob use QKD (such as BB84) to generate k to use e.g. in one-time pad.

Uses quantum channel

(1) Alice generates N rand bit pairs (a_i, r_i).

(2) a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to Bo

a_i
0
0
1
1

(3) Bob generates N random & measure qubit i in basis getting result s_i

(4) Alice & Bob announce publicly
If $a_i = b_i$, keep bit i
If $a_i \neq b_i$, discard bit i
(Note A&B don't announce

BB84 protocol

"Bennett - Brassard 1984"

Alice & Bob use QKD (such as BB84) to generate k to use e.g. in one-time pad.

- Uses quantum channel
- Uses classical channels

(1) Alice generates N ran... bit pairs (a_i, r_i)..

(2) a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to ...

(3) Bob generates N rando... & measure qubit i in basi... getting result s_i

(4) Alice & Bob announce publi...
If $a_i = b_i$, keep bit i
If $a_i \neq b_i$, discard bit i
- (Note A&B don't announc...

BB84 protocol

"Bennett - Brassard 1984"

Alice & Bob use QKD (such as BB84) to generate k to use e.g. in one-time pad.

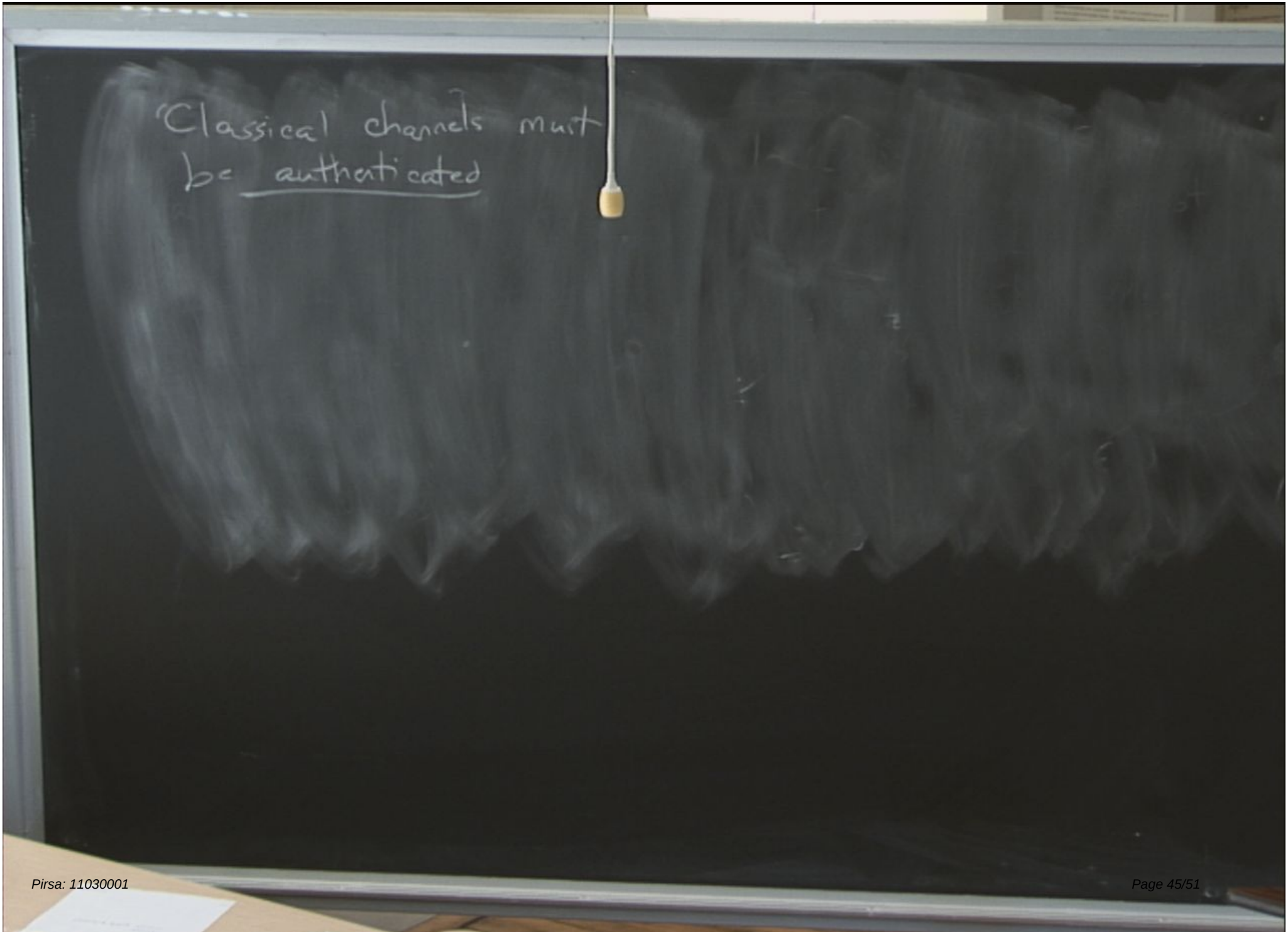
- Uses quantum channel
- Uses classical channels
  - Can be public

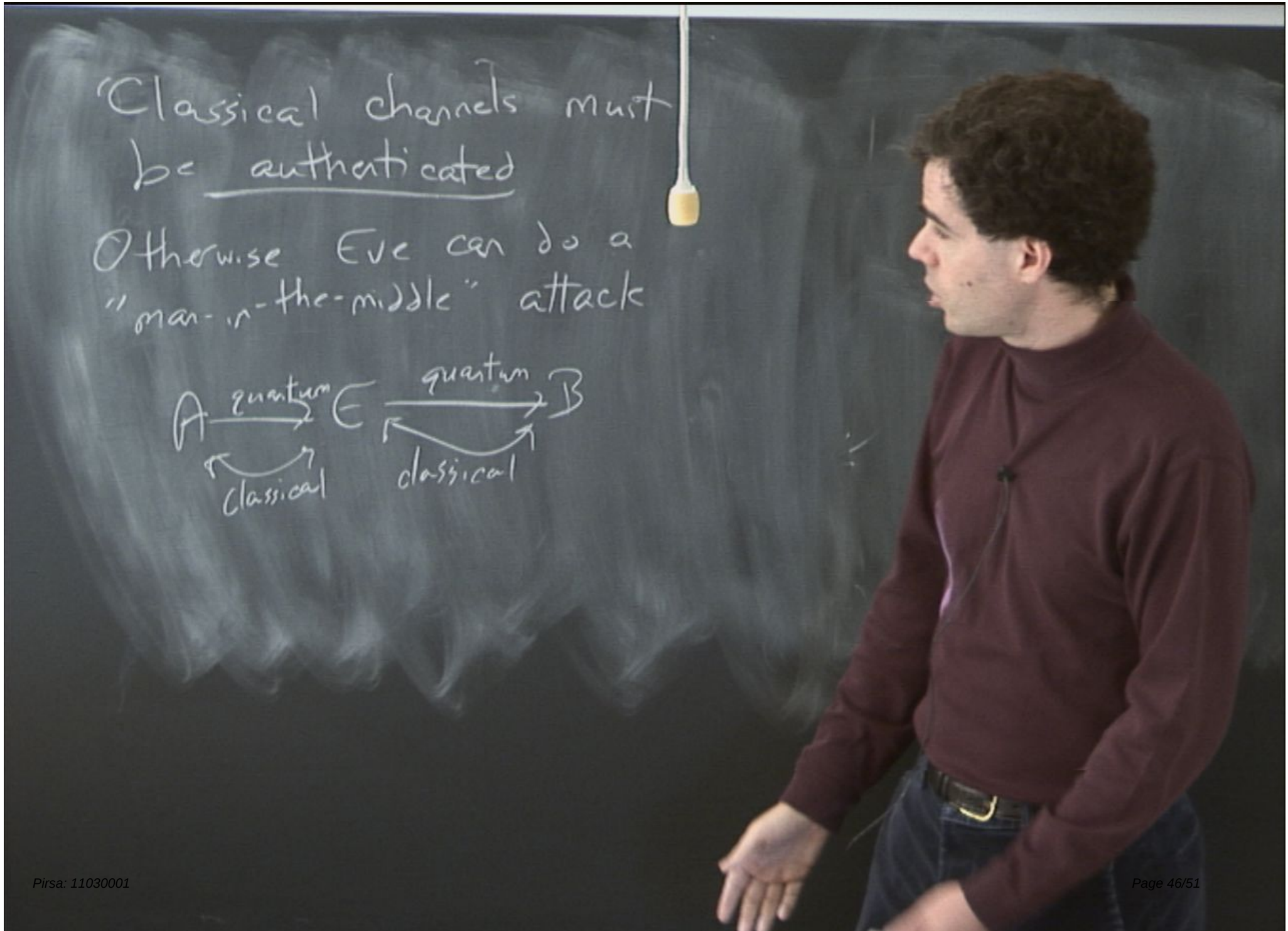
(1) Alice generates N ran bit pairs (a_i, r_i).

(2) a_i specifies basis
r_i specifies state
Alice prepares this state & sends it to

(3) Bob generates N rando & measure qubit i in basi getting result s_i

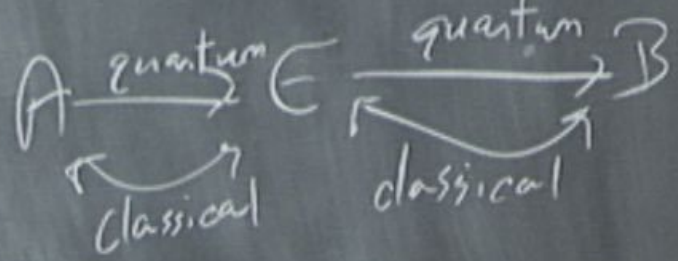
(4) Alice & Bob announce publi
If $a_i = b_i$, keep bit i
If $a_i \neq b_i$, discard bit i
(Note A&B don't announc


Classical channels must
be authenticated


Classical channels must be authenticated
Otherwise Eve can do a "man-in-the-middle" attack
A
quantum
E
quantum
B
classical
classical

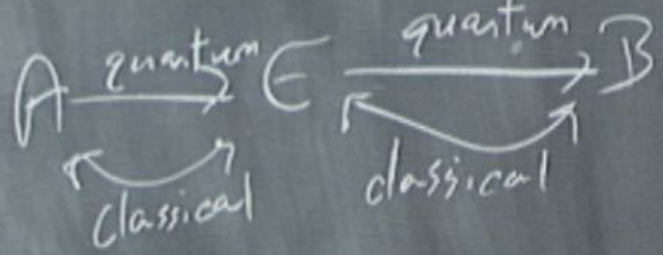
Classical channels must be authenticated

Otherwise Eve can do a "man-in-the-middle" attack



Classical channels must be authenticated → Can use secret key

Otherwise Eve can do a "man-in-the-middle" attack



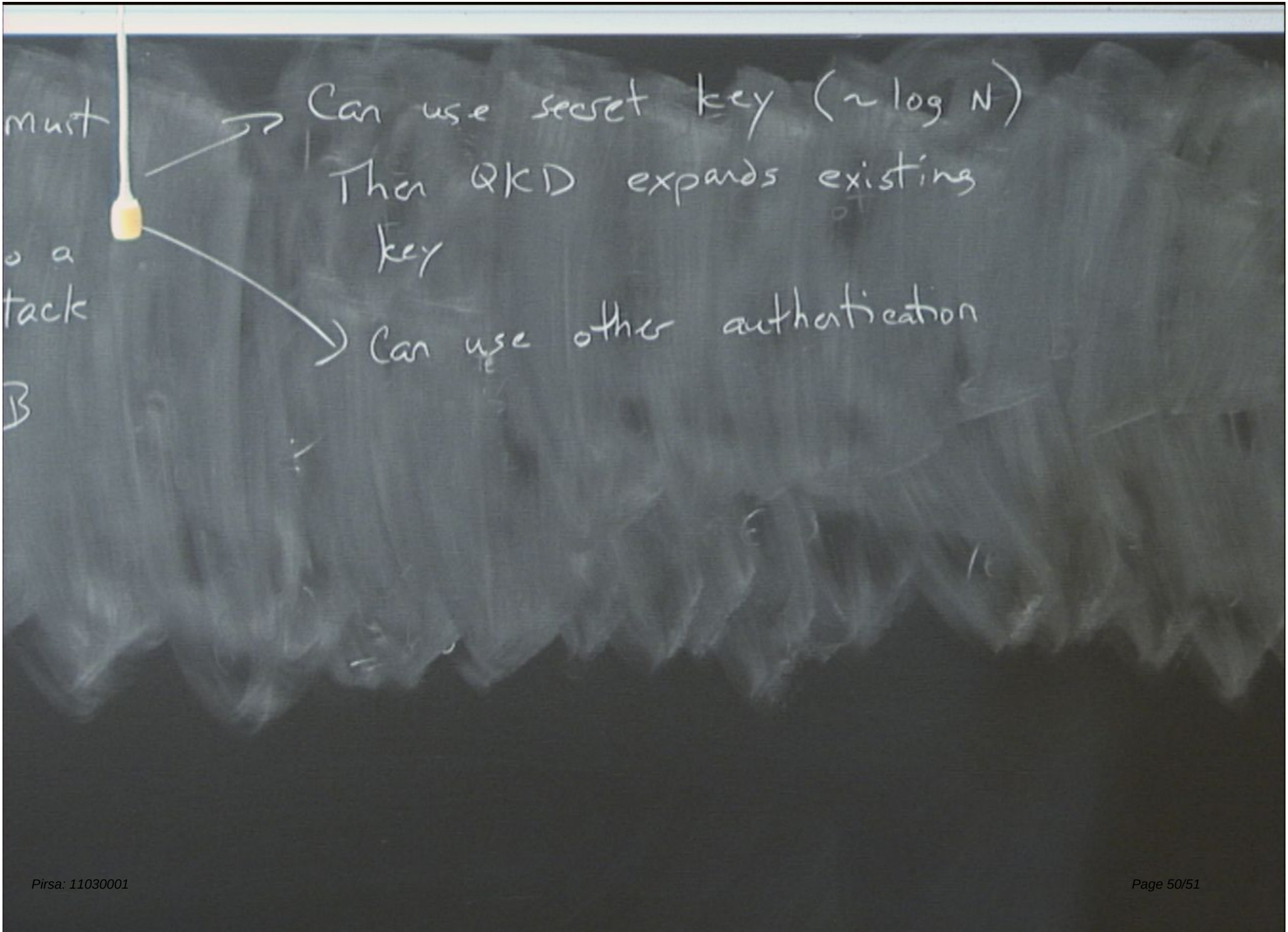
must

o a

tack

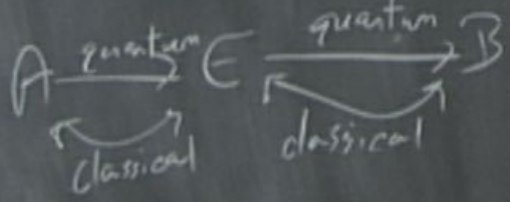
B

→ Can use secret key ($\sim \log N$)
Then QKD expands existing key


must
a
tack
B
Can use secret key (~ log N)
Then QKD expands existing
key
Can use other authentication

Classical channels must be authenticated

Otherwise Eve can do a "man-in-the-middle" attack



Can use secret key ($\sim \log N$)
Then QKD expands existing key

Can use other authentication