

# Efficient verification of observational equivalences in finite process calculi

Vincent Cheval, Steve Kremer, Itsaka Rakotonirina

Inria Nancy Grand-Est

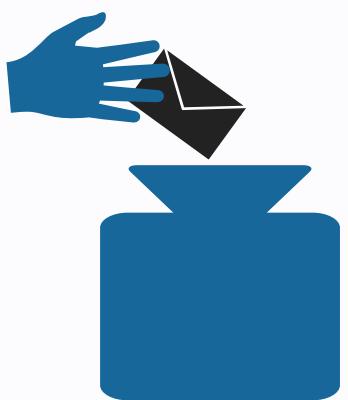
# Security protocols



TLS



Wifi



E-voting



E-passport

# Security protocols



TLS



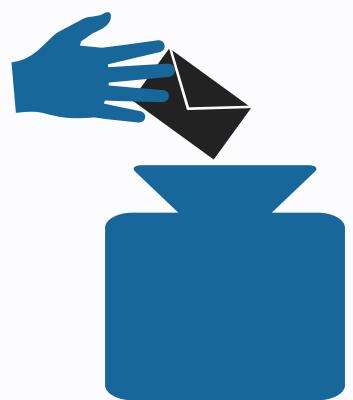
2016 (early ver. 1.3)



Wifi



2017 (WPA2)



E-voting



2010 (Helios)

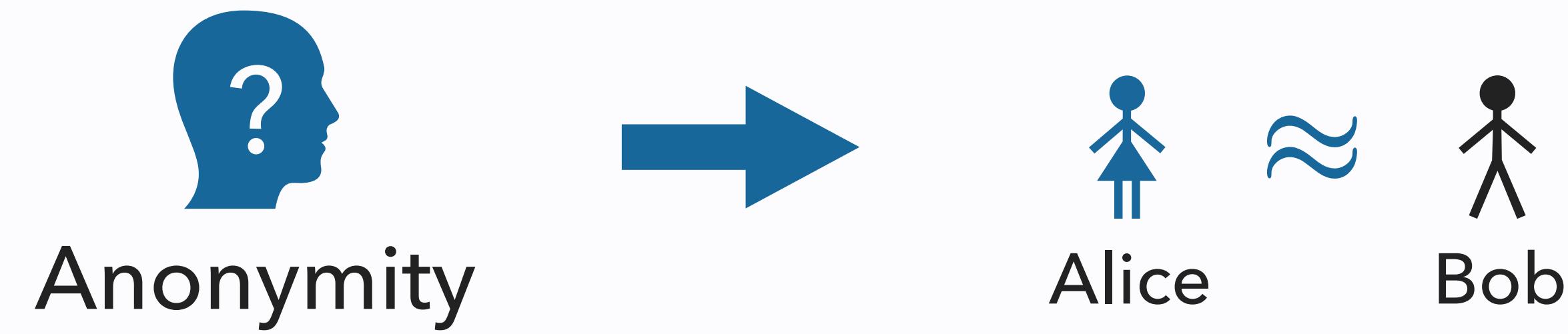


E-passport

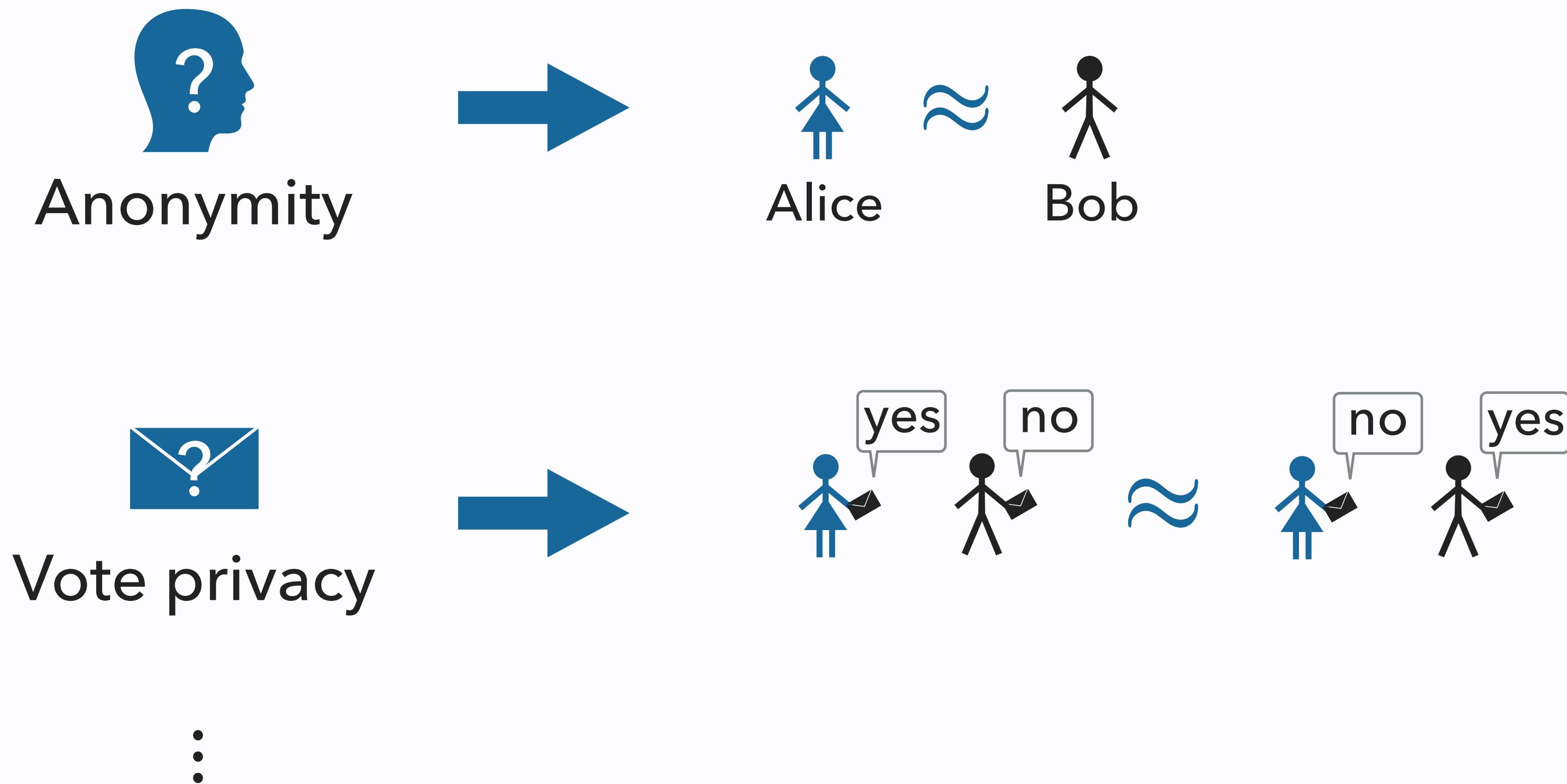


2013 (BAC)

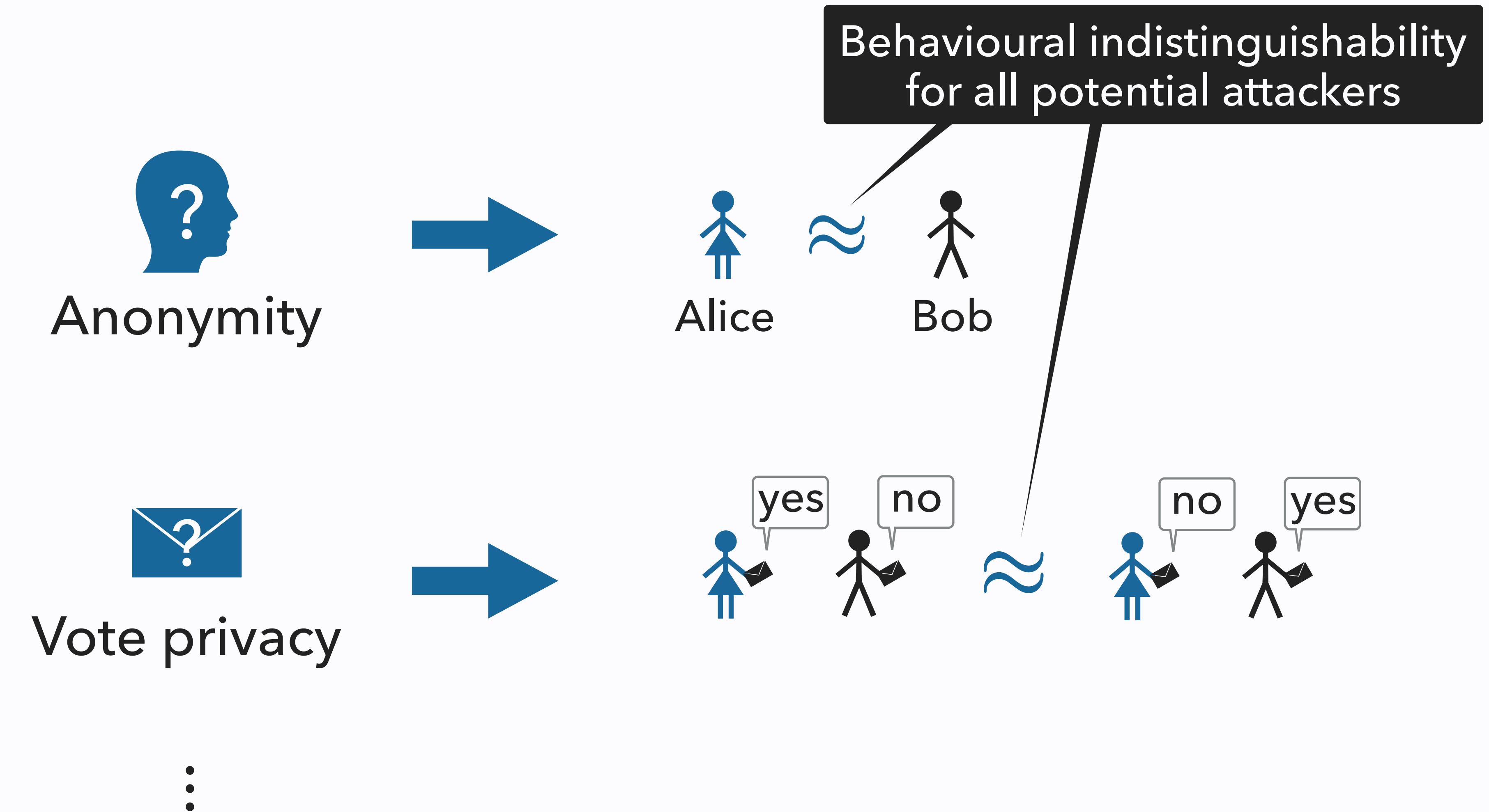
# Privacy as indistinguishability



# Privacy as indistinguishability



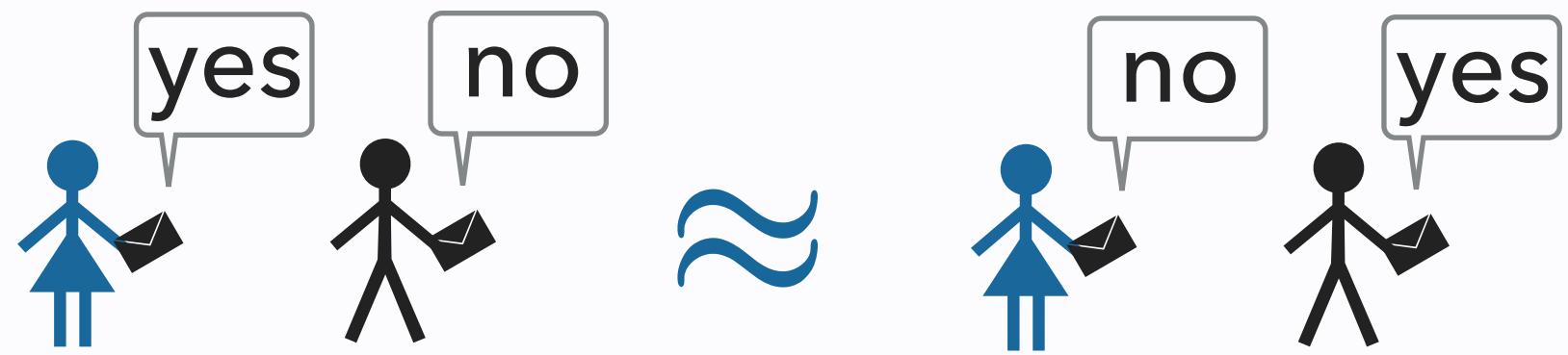
# Privacy as indistinguishability



# Privacy as indistinguishability



# Privacy as indistinguishability

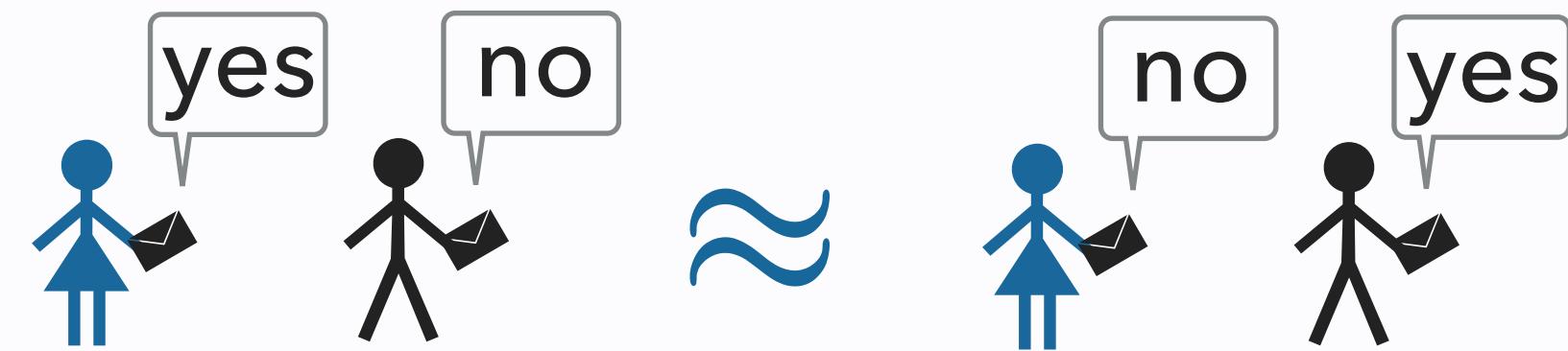


Equivalence coNEXP-complete  
for a fixed number of participants

[S&P18] S. Kremer, V. Cheval, I. Rakotonirina.

*DEEPSEC: Deciding equivalence properties  
in security protocols – theory and practice*

# Privacy as indistinguishability



Equivalence coNEXP-complete  
for a fixed number of participants

[S&P18] S. Kremer, V. Cheval, I. Rakotonirina.

*DEEPSEC: Deciding equivalence properties  
in security protocols – theory and practice*

Observation 🤔

Each time, the two processes  
share a common structure

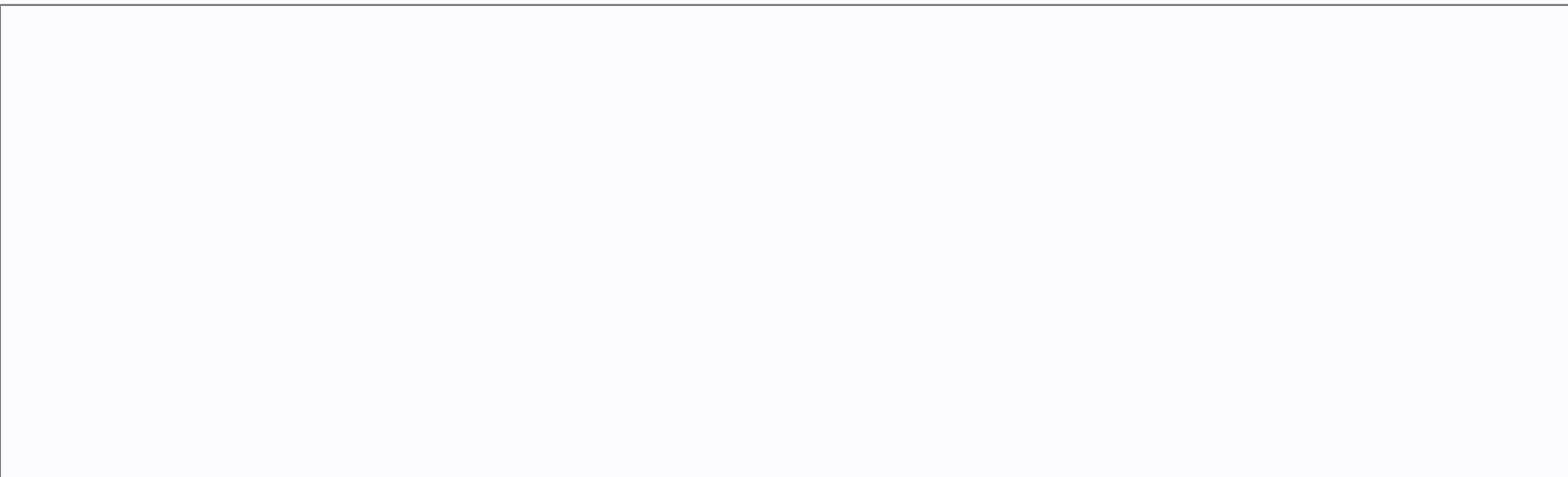
# Contributions

- + A refinement of trace equivalence  
for processes with structural similarities
- + Partial-order reductions  
for any process for this new equivalence
- + Integration into the DeepSec prover

# Trace equivalence

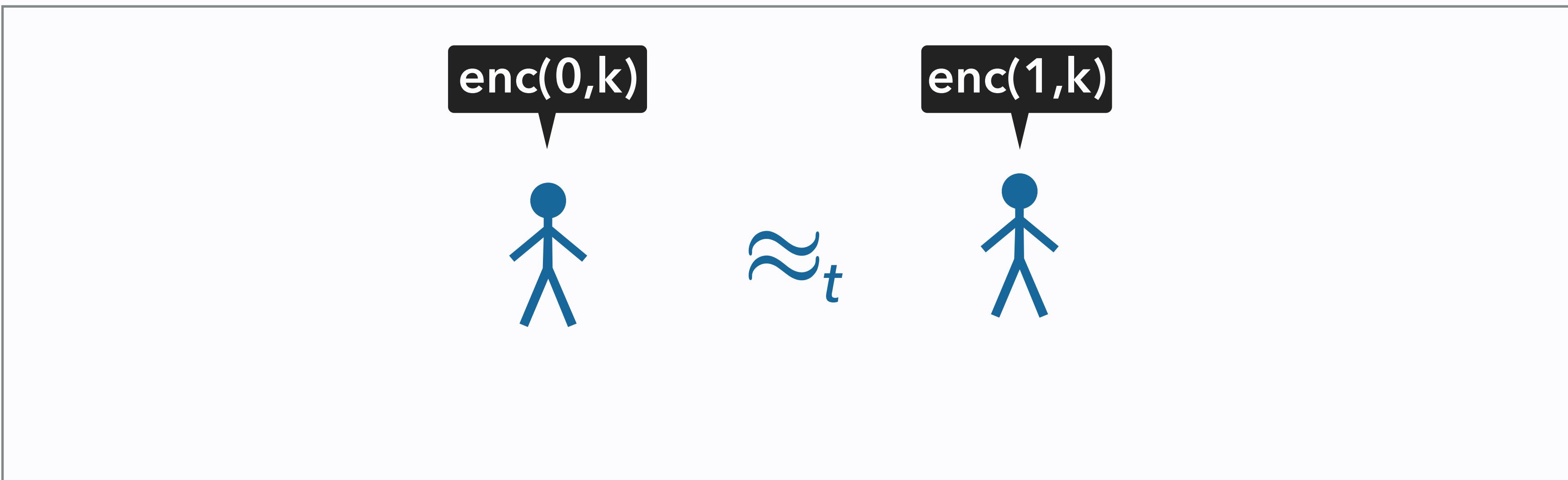
# Modelling indistinguishability

A simple example



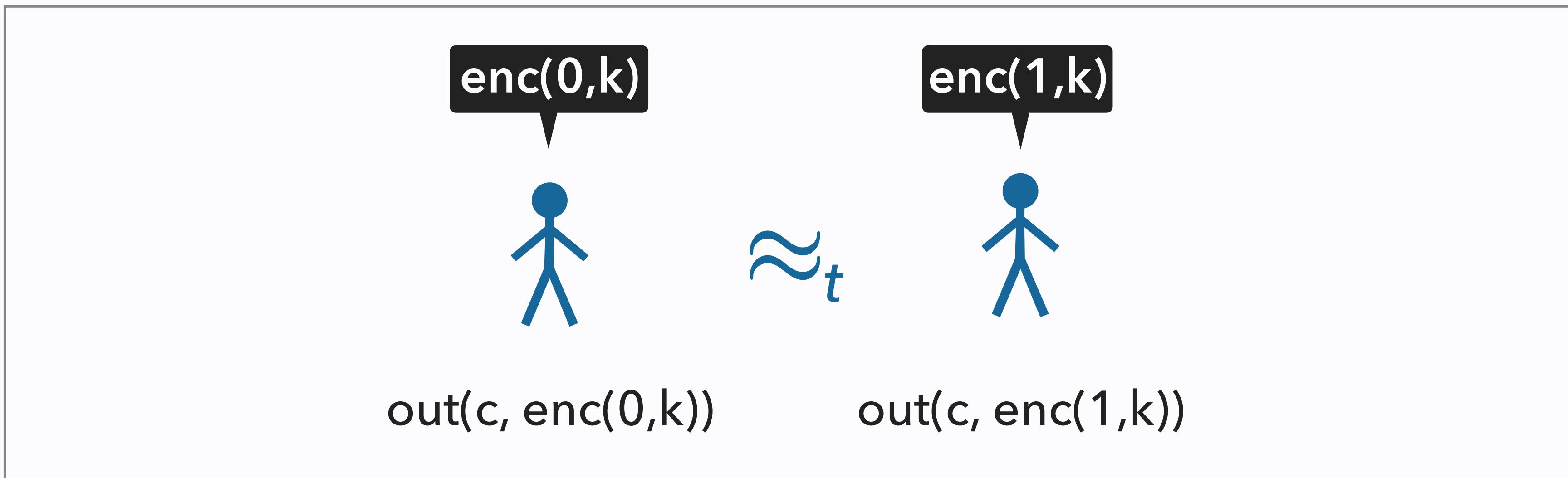
# Modelling indistinguishability

A simple example



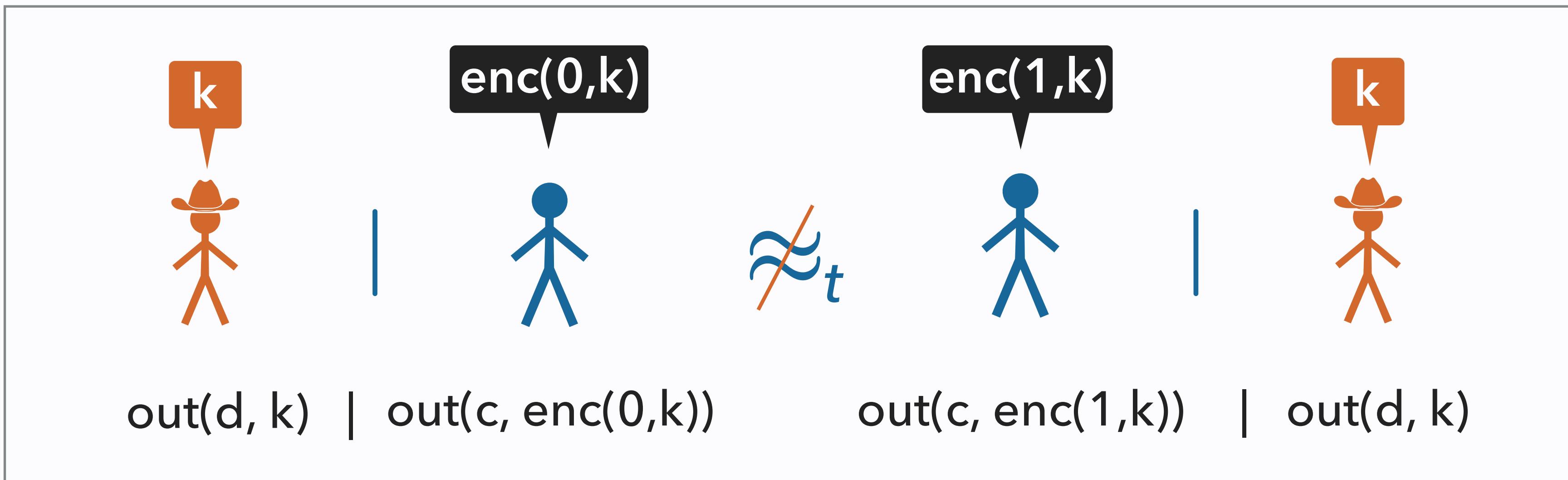
# Modelling indistinguishability

A simple example



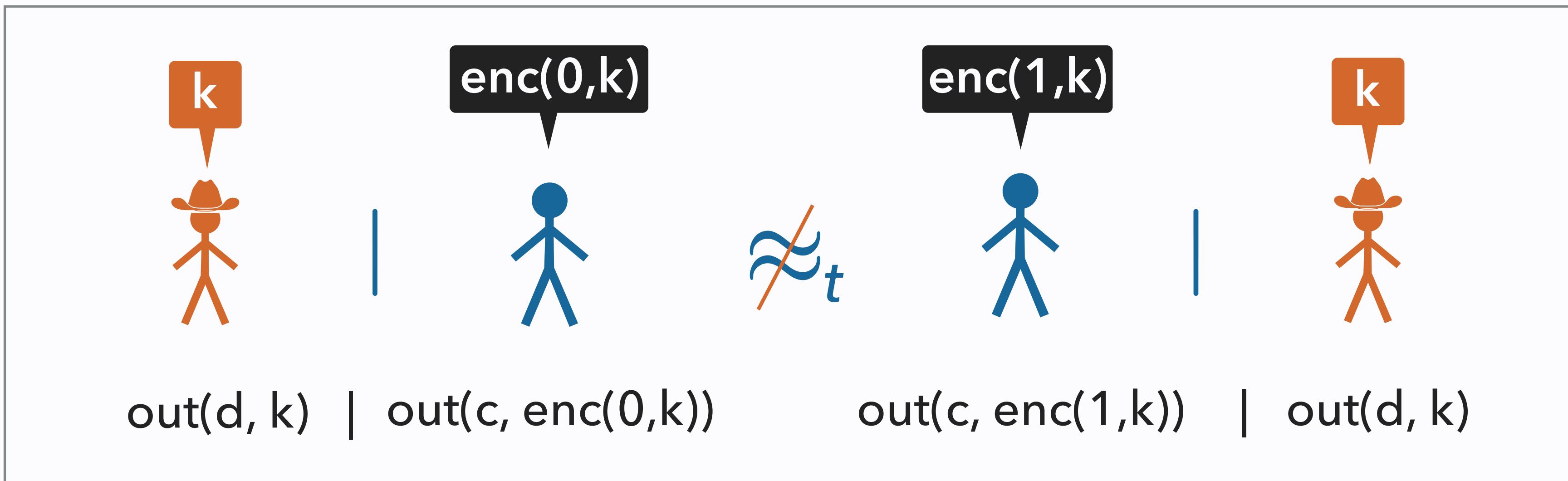
# Modelling indistinguishability

A simple example

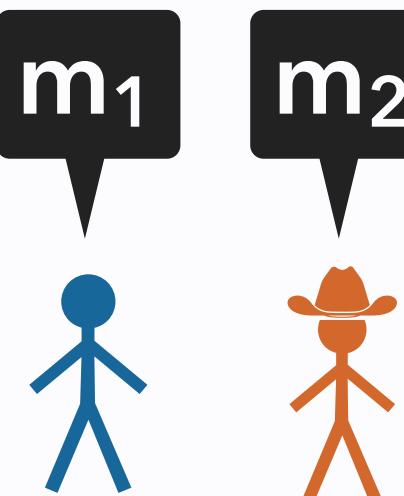


# Modelling indistinguishability

A simple example

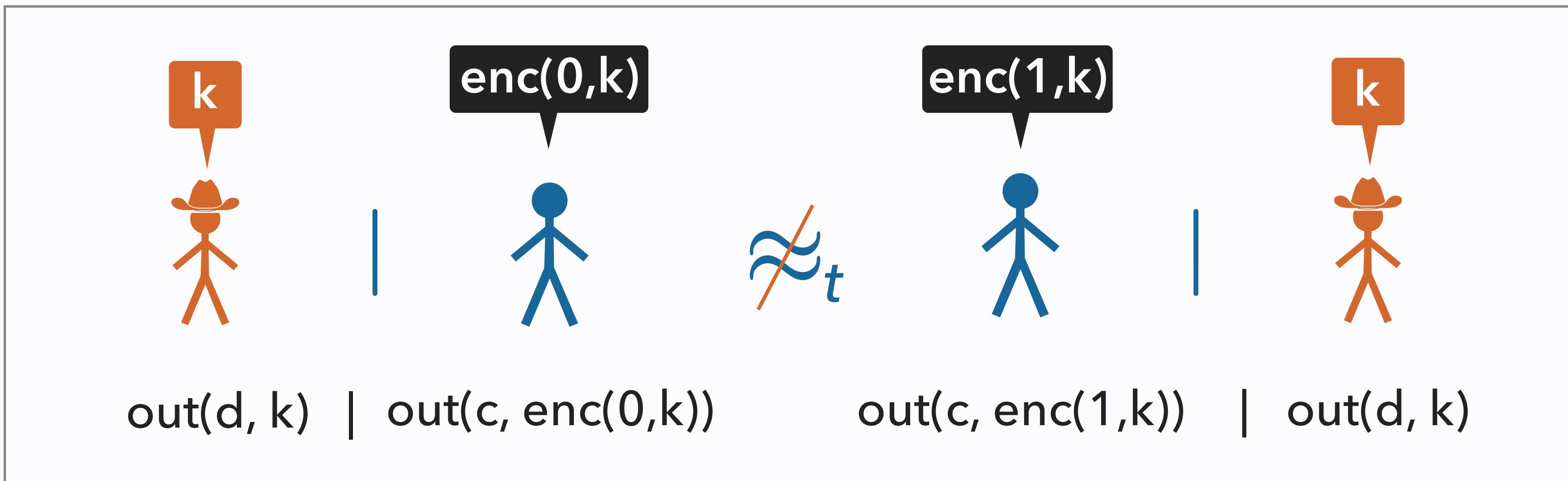


Distinguishing execution:



# Modelling indistinguishability

A simple example



Distinguishing execution:

$m_1$   $m_2$

$+ \text{test } dec(m_1, m_2) = 0$  ?

# Modelling indistinguishability

## Formalism

$$P_0 \approx_t P_1 \quad \text{iff} \quad \forall t \in \text{Traces}(P_i), \exists t' \in \text{Traces}(P_{1-i}), t \sim t'$$

# Modelling indistinguishability

## Formalism

algebra of **finite concurrent processes**

$$P_0 \approx_t P_1 \quad \text{iff} \quad \forall t \in \text{Traces}(P_i), \exists t' \in \text{Traces}(P_{1-i}), t \sim t'$$

# Modelling indistinguishability

## Formalism

algebra of **finite concurrent processes**

$$P_0 \approx_t P_1 \quad \text{iff} \quad \forall t \in \text{Traces}(P_i), \exists t' \in \text{Traces}(P_{1-i}), t \sim t'$$

**sequences of inputs/outputs** in  
an active adversarial environment

# Modelling indistinguishability

## Formalism

algebra of **finite concurrent processes**

$$P_0 \approx_t P_1 \quad \text{iff} \quad \forall t \in \text{Traces}(P_i), \exists t' \in \text{Traces}(P_{1-i}), t \sim t'$$

**sequences of inputs/outputs** in  
an active adversarial environment

Output  $\Rightarrow$  increases the attacker's knowledge

Input  $\Rightarrow$  receives a term from the attacker

# Modelling indistinguishability

## Formalism

algebra of **finite concurrent processes**

$$P_0 \approx_t P_1 \quad \text{iff} \quad \forall t \in \text{Traces}(P_i), \exists t' \in \text{Traces}(P_{1-i}), t \sim t'$$

**sequences of inputs/outputs** in  
an active adversarial environment

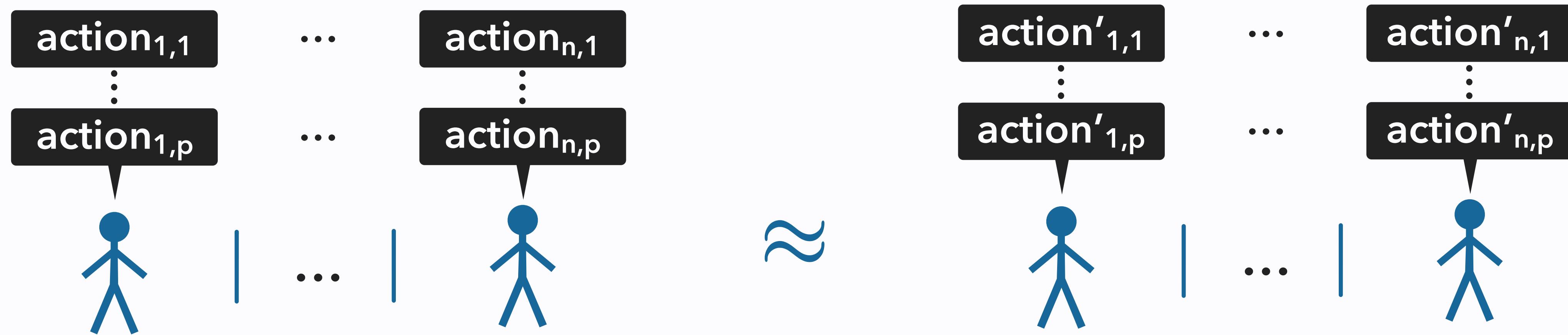
**static indistinguishability** of  
sequences of inputs/outputs

Output  $\Rightarrow$  increases the attacker's knowledge

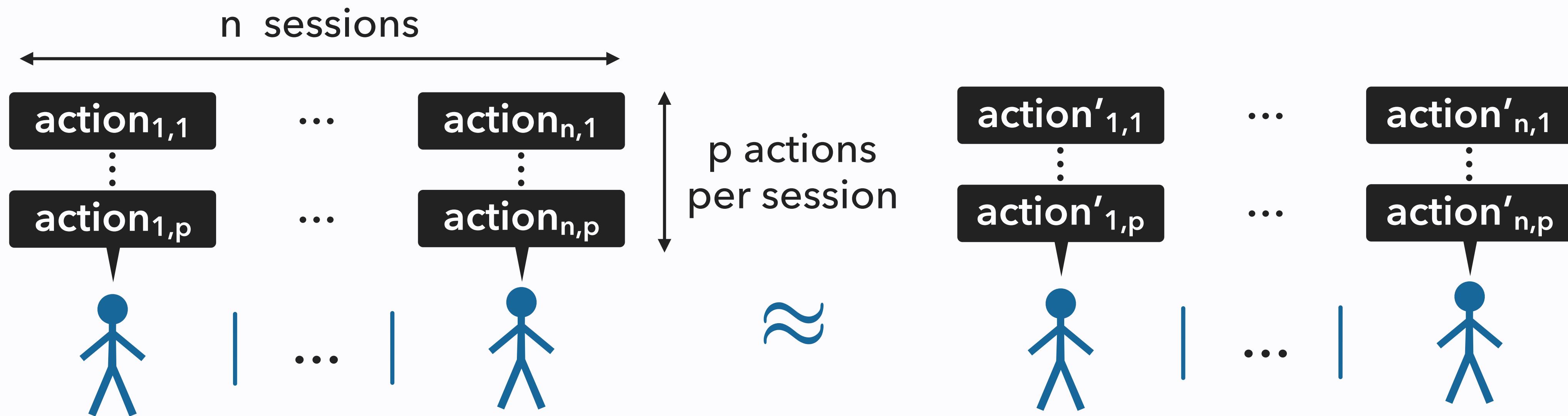
Input  $\Rightarrow$  receives a term from the attacker

# Trace equivalence... in practice

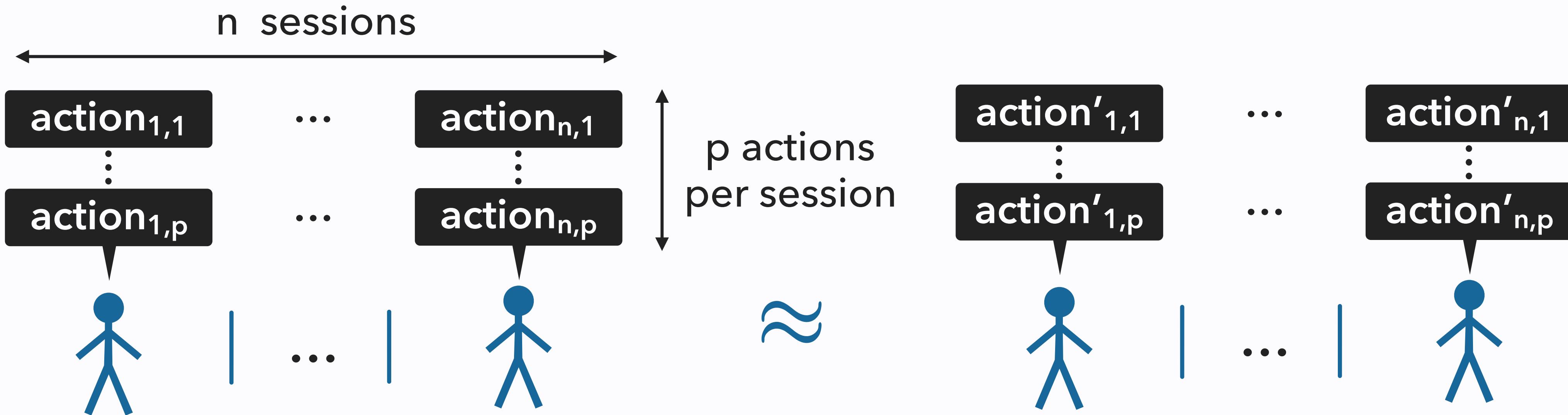
# A combinatorial fact



# A combinatorial fact

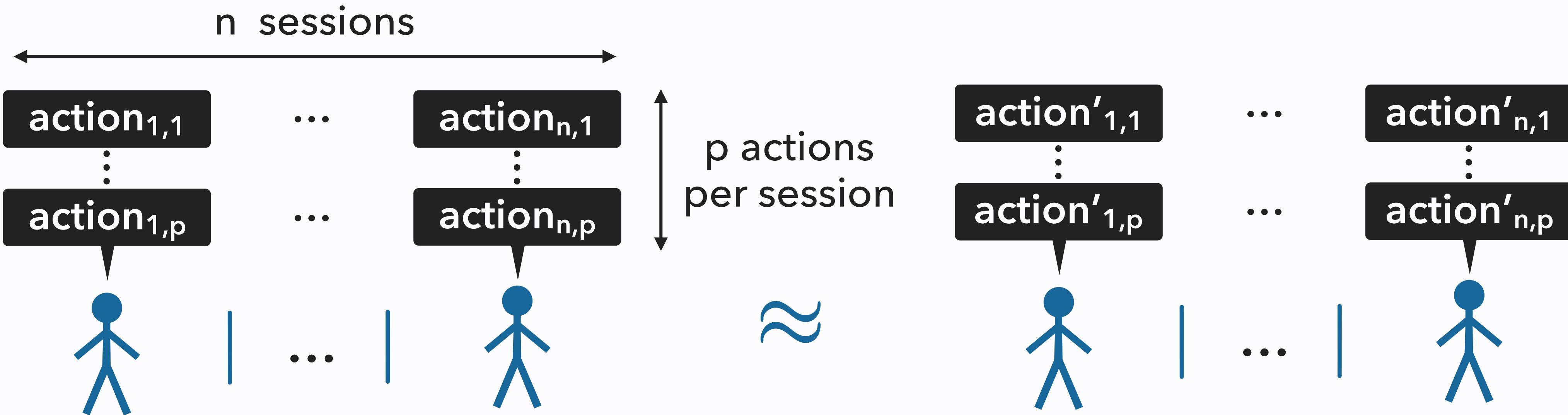


# A combinatorial fact



Goal  $\forall$  sequences  $[a_1 \ a_2 \ \dots \ a_{np}]$ ,  $\exists$  equivalent sequence  $[a'_1 \ a'_2 \ \dots \ a'_{np}]$

# A combinatorial fact

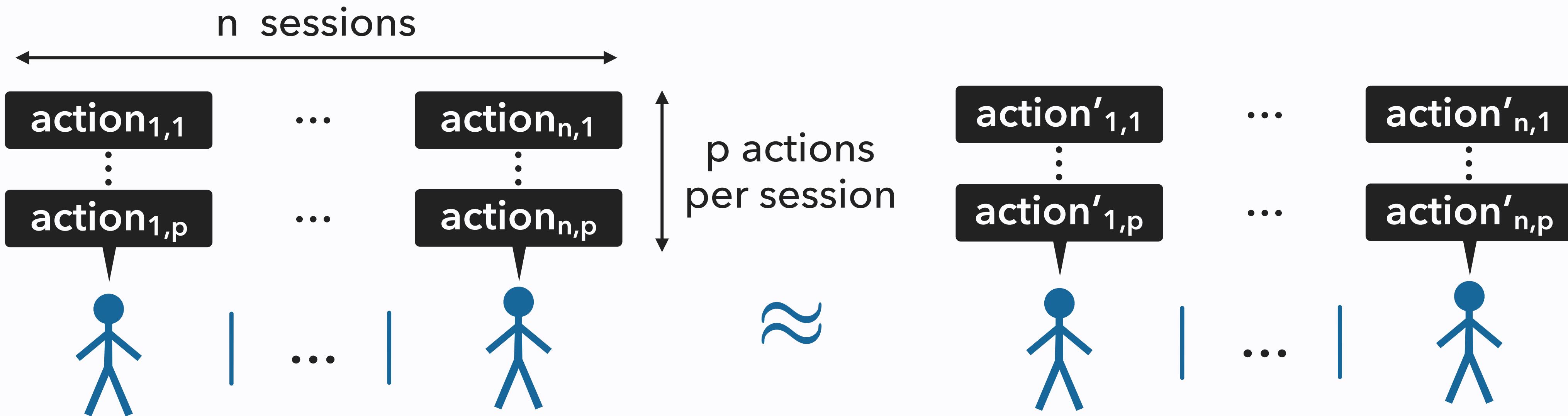


Goal  $\forall$  sequences  $[a_1 \ a_2 \ \dots \ a_{np}]$ ,  $\exists$  equivalent sequence  $[a'_1 \ a'_2 \ \dots \ a'_{np}]$

$\sim(np)!$  matchings

Actions

# A combinatorial fact



Goal  $\forall$  sequences  $[a_1 \ a_2 \ \dots \ a_{np}]$ ,  $\exists$  equivalent sequence  $[a'_1 \ a'_2 \ \dots \ a'_{np}]$

$\sim(np)!$  matchings

Actions

$\sim n!$  matchings

Sessions

# Formally: process pairing

# Formally: process pairing

(MATCH)  $(P_1 | \dots | P_n, Q_1 | \dots | Q_n)$   $\longrightarrow (P_1, Q_{\sigma(1)}), \dots, (P_n, Q_{\sigma(n)})$   
 $\sigma$  permutation of  $\{1, \dots, n\}$

# Formally: process pairing

(MATCH)  $(P_1 | \dots | P_n, Q_1 | \dots | Q_n) \longrightarrow (P_1, Q_{\sigma(1)}), \dots, (P_n, Q_{\sigma(n)})$   
 $\sigma$  permutation of  $\{1, \dots, n\}$

(EXEC)  $(P, Q) \xrightarrow{\alpha} (P', Q')$  if  $P \xrightarrow{\alpha} P'$  and  $Q \xrightarrow{\alpha} Q'$  (in the single-process semantics)

# Formally: process pairing

(MATCH)  $(P_1 | \dots | P_n, Q_1 | \dots | Q_n) \longrightarrow (P_1, Q_{\sigma(1)}), \dots, (P_n, Q_{\sigma(n)})$   
 $\sigma$  permutation of  $\{1, \dots, n\}$

(EXEC)  $(P, Q) \xrightarrow{\alpha} (P', Q')$  if  $P \xrightarrow{\alpha} P'$  and  $Q \xrightarrow{\alpha} Q'$  (in the single-process semantics)

Trace Equiv.  $\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t' \sim t$

# Formally: process pairing

(MATCH)  $(P_1 | \dots | P_n, Q_1 | \dots | Q_n) \longrightarrow (P_1, Q_{\sigma(1)}), \dots, (P_n, Q_{\sigma(n)})$   
 $\sigma$  permutation of  $\{1, \dots, n\}$

(EXEC)  $(P, Q) \xrightarrow{\alpha} (P', Q')$  if  $P \xrightarrow{\alpha} P'$  and  $Q \xrightarrow{\alpha} Q'$  (in the single-process semantics)

~~Trace Equiv.~~

Equiv. by session

$\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t' \sim t$

+  $\exists t^2 \in \text{Traces}(P, Q), \text{fst}(t^2) = t$  and  $\text{snd}(t^2) = t'$

# Optimisations

# For trace equivalence



[CONCUR15] D. Baelde, S. Delaune, L. Hirschi.  
*Partial-order reductions for security protocols*

# For trace equivalence



[CONCUR15] D. Baelde, S. Delaune, L. Hirschi.  
*Partial-order reductions for security protocols*

$$\forall t \in \text{Traces}(P), \quad \exists t' \in \text{Traces}(Q), \quad t \sim t'$$

# For trace equivalence



[CONCUR15] D. Baelde, S. Delaune, L. Hirschi.  
*Partial-order reductions for security protocols*

$$\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t \sim t'$$

💡 reduce the number  
of traces to check?



# For trace equivalence



[CONCUR15] D. Baelde, S. Delaune, L. Hirschi.  
*Partial-order reductions for security protocols*

$$\forall t \in \text{Traces}(P), \quad \exists t' \in \text{Traces}(Q), \quad t \sim t'$$

reduce the number  
of traces to check?

## Main theorem

If  $P, Q$  are determinate, it is sufficient to consider traces up to permutation of adjacent independent actions.

# For trace equivalence



[CONCUR15] D. Baelde, S. Delaune, L. Hirschi.

*Partial-order reductions for security protocols*

$$\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t \sim t'$$

reduce the number  
of traces to check?

## Main theorem

If  $P, Q$  are determinate, it is sufficient to consider traces up to permutation of adjacent **independent** actions.

Concurrent actions  
with no data flow

# For equivalence by session

$$\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t \sim t'$$

-💡 reduce the number  
of traces to check?

$$+ \exists t^2 \in \text{Traces}(P,Q), \text{fst}(t^2) = t \text{ and } \text{snd}(t^2) = t'$$

## Main theorem

If  $P, Q$  are determinate, it is sufficient to consider traces up to permutation of adjacent **independent** actions.

Concurrent actions  
with no data flow

# For equivalence by session

$$\forall t \in \text{Traces}(P), \exists t' \in \text{Traces}(Q), t \sim t'$$

-💡 reduce the number  
of traces to check?

+  $\exists t^2 \in \text{Traces}(P,Q), \text{fst}(t^2) = t \text{ and } \text{snd}(t^2) = t'$

## Main theorem

~~If  $P, Q$  are determinate~~, it is sufficient to consider traces up to permutation of adjacent **independent** actions.

Concurrent actions  
with no data flow

# Experiments

# Approaches to prove trace equivalence



# Approaches to prove trace equivalence



Baseline

P and Q trace equivalent?

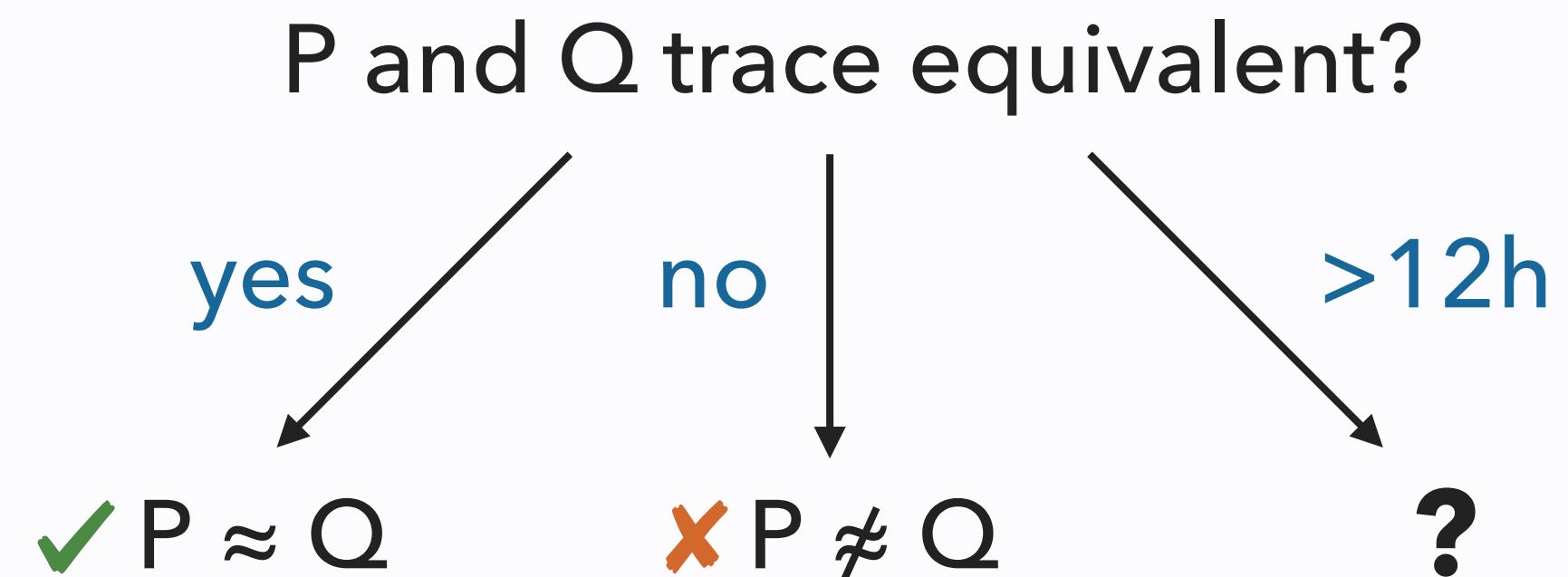
Structure-guided

P and Q equivalent by session?

# Approaches to prove trace equivalence



Baseline



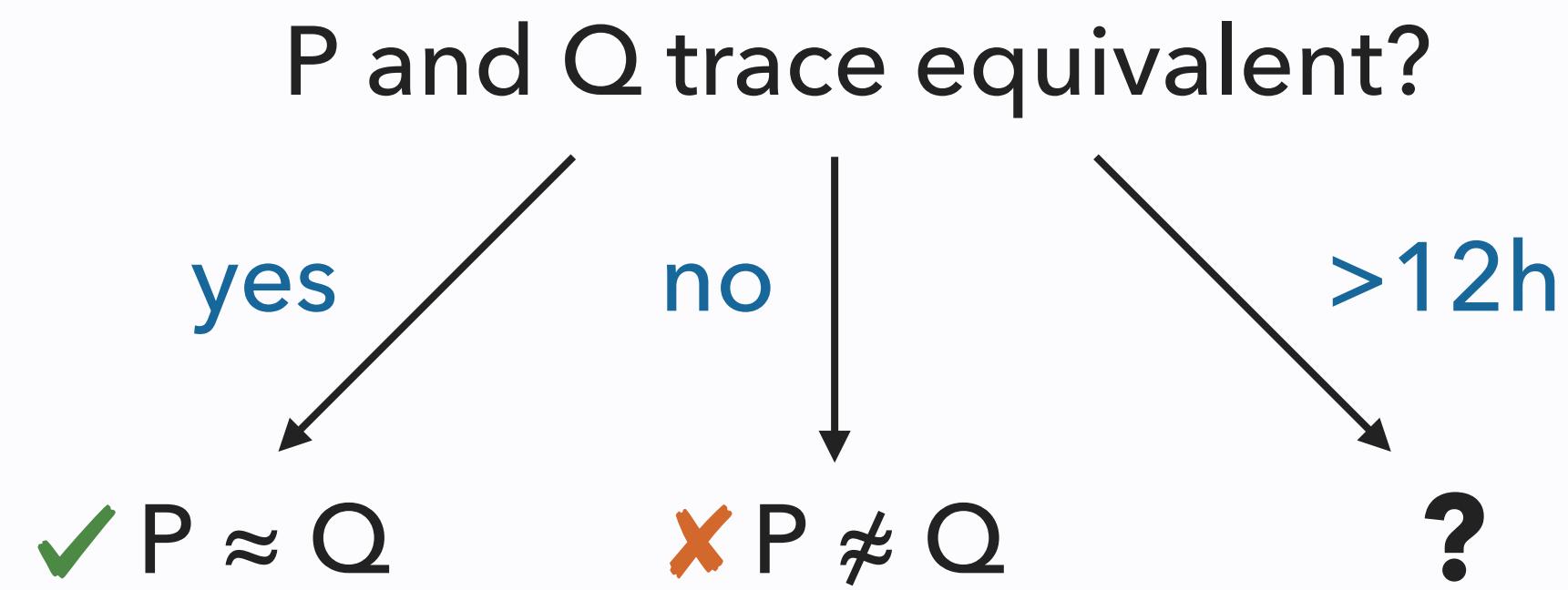
Structure-guided

P and Q equivalent by session?

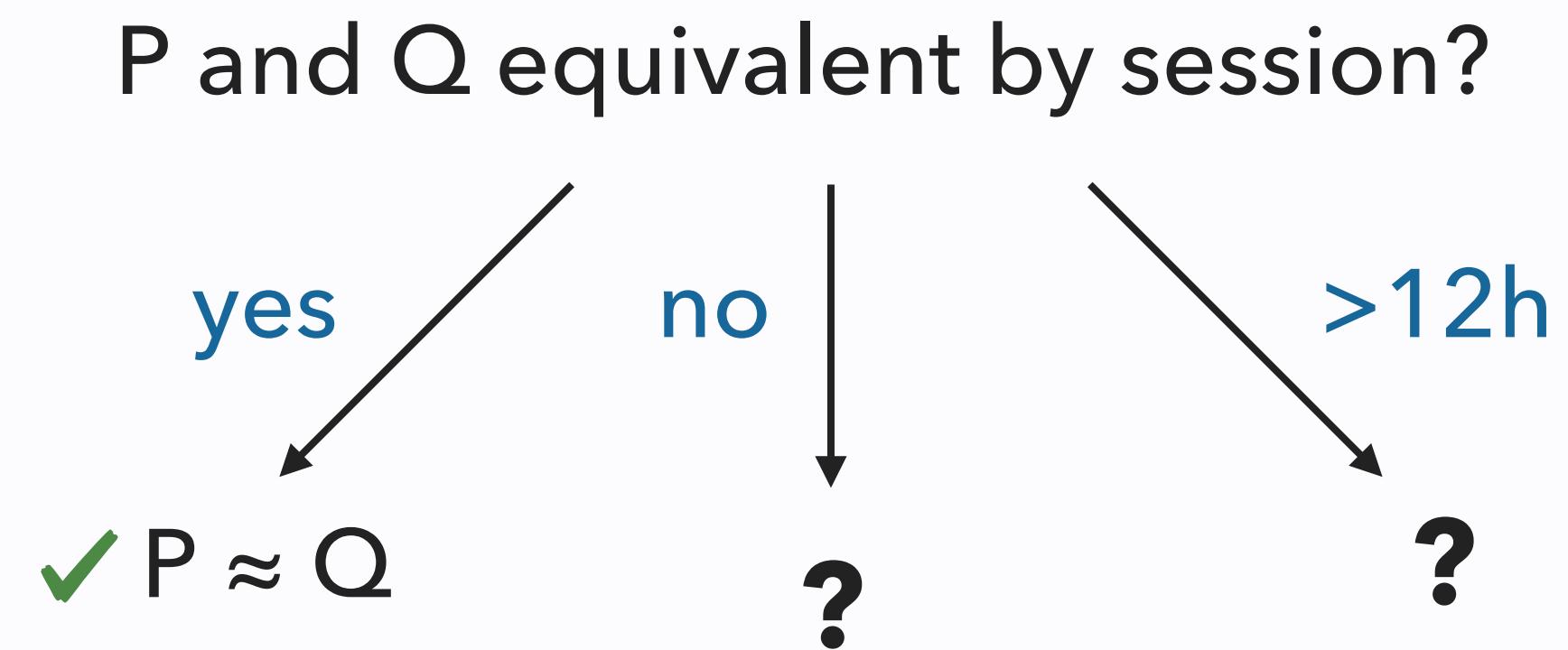
# Approaches to prove trace equivalence



Baseline



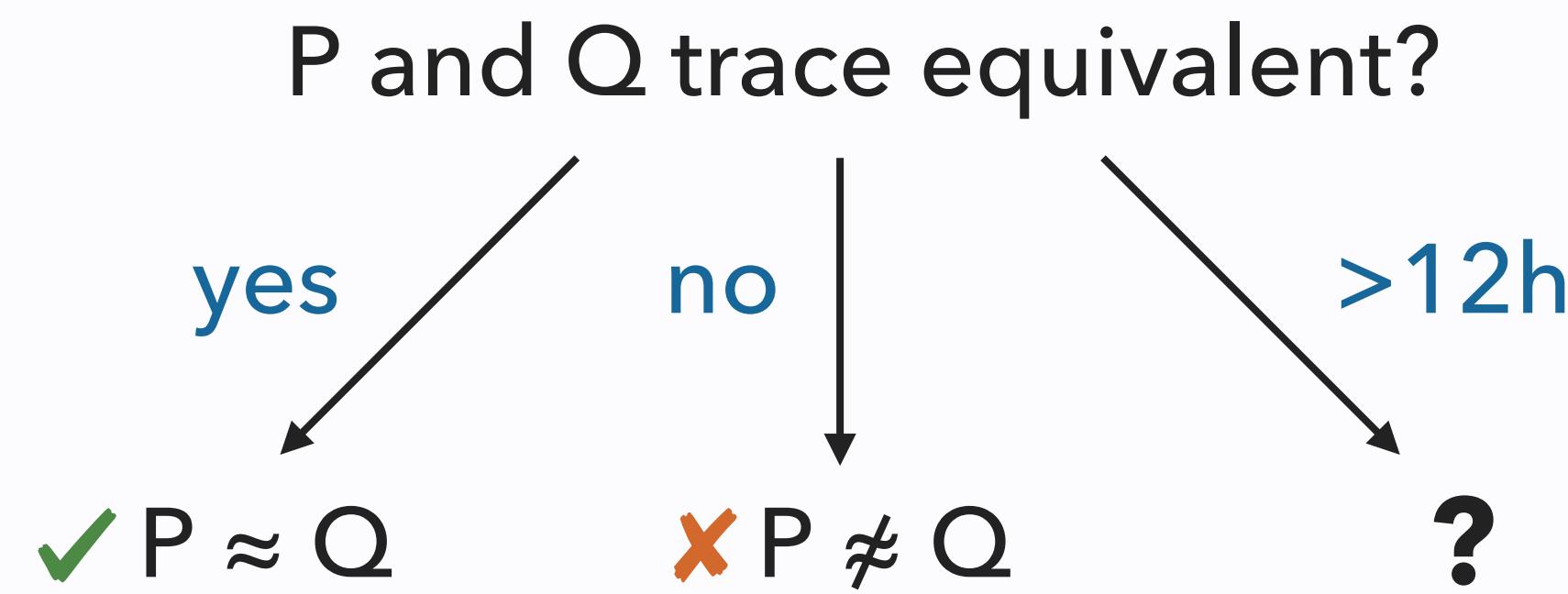
Structure-guided



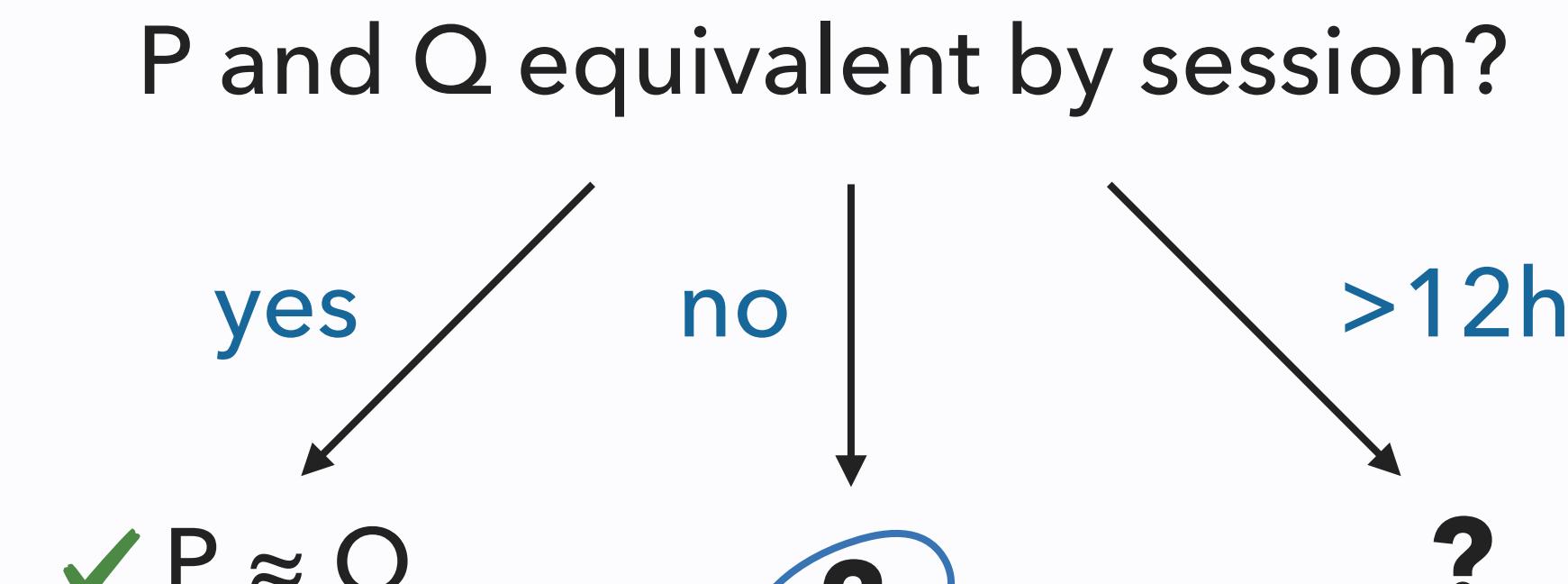
# Approaches to prove trace equivalence



Baseline



Structure-guided



Ongoing work:  
conclude in this case  
(currently: heuristic)

# Experimental results (1)

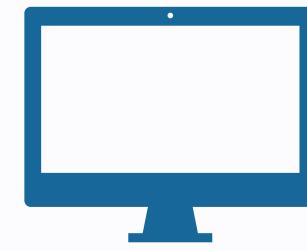
## Unlinkability in the Electronic passport

# Experimental results (1)

## Unlinkability in the Electronic passport



2 identical passports   readers



2 different passports   readers

# Experimental results (1)

## Unlinkability in the Electronic passport



2 identical passports



2 different passports



Scenario	baseline	structure-guided
2 identical	<1s ✗	<1s ✗
2 identical + 1 fresh	>12h	2s ✗
3 identical + 1 fresh	>12h	3s ✗
2 identical + 2 fresh	>12h	1min20 ✓
2 identical + 3 fresh	>12h	11h06 ✓

✗ non-equivalent  
✓ equivalent

# Experimental results (1)

## Unlinkability in the Electronic passport

?  
≈

- ✗ non-equivalent
- ✓ equivalent

# Experimental results (2)

Vote privacy in Helios (vote swap)



- ✗ non-equivalent
- ✓ equivalent

# Experimental results (2)

## Vote privacy in Helios (vote swap)



Scenario	baseline	structure-guided
no revote	<1s ✓	<1s ✓
A x 2 + B x 1	2h41 ✓	1min2 ✓
A x 3 + B x 2	>12h	7min40 ✓
A x 4 + B x 2	>12h	16min36 ✓
A x 7 + B x 3	>12h	3h53 ✓

✗ non-equivalent  
✓ equivalent



v1.0.2



v1.0.2

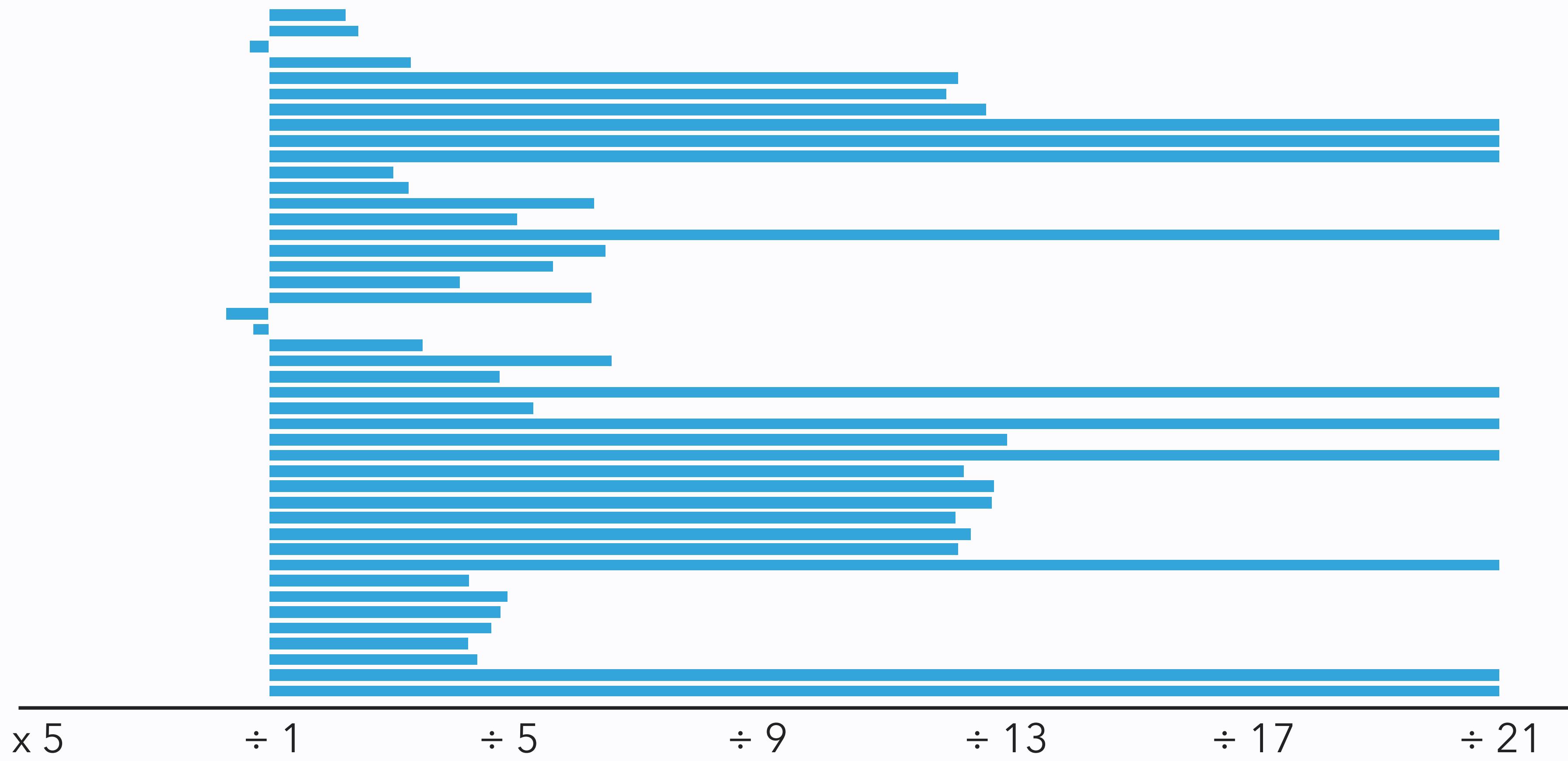
v2.0.0



v1.0.2

v2.0.0

## Evolution of verification time v1.0.2 → v2.0.0

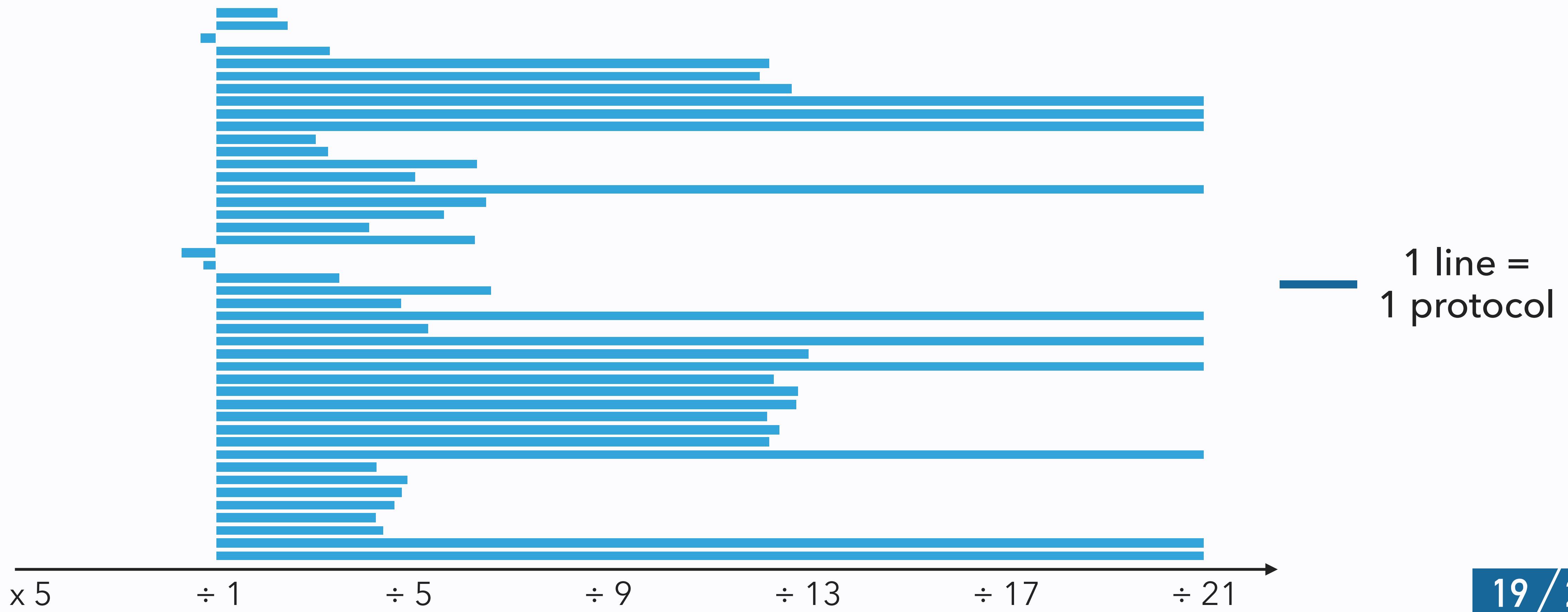




v1.0.2

v2.0.0

## Evolution of verification time v1.0.2 → v2.0.0

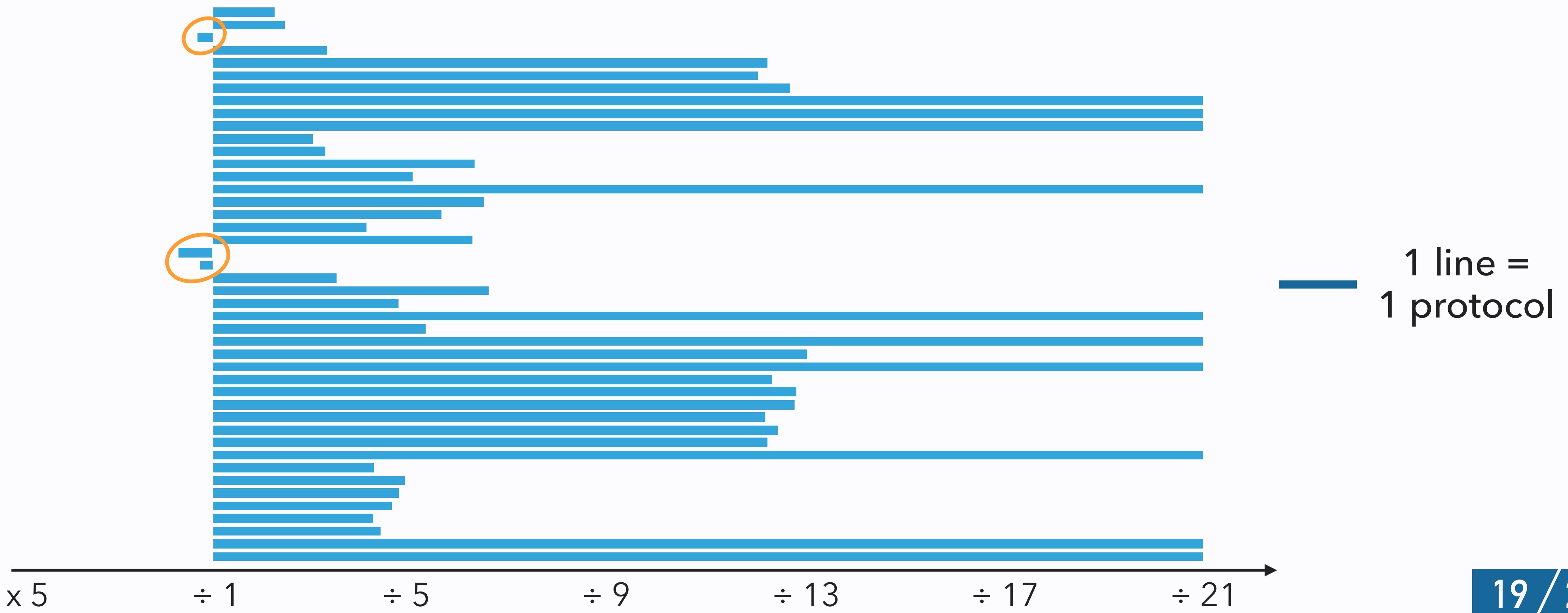




v1.0.2

v2.0.0

## Evolution of verification time v1.0.2 → v2.0.0

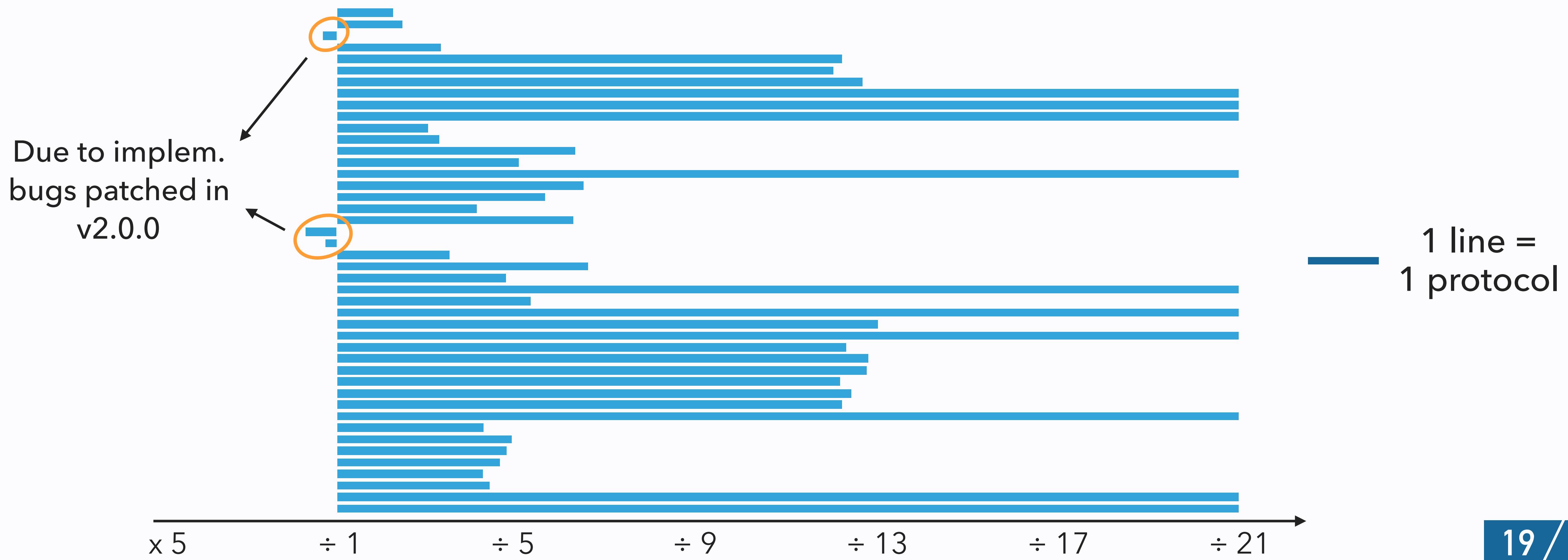




v1.0.2

v2.0.0

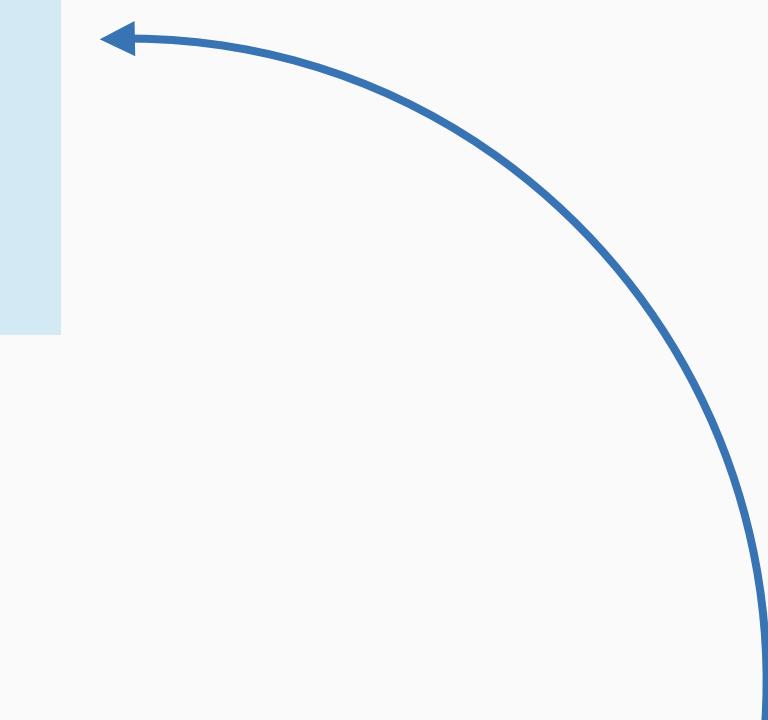
## Evolution of verification time v1.0.2 → v2.0.0



```
11  free yes.  
12  free no.  
13  
14  (* Randomized asymmetric encryption *)  
15  
16  fun aenc/3.  
17  fun pk/1.  
18  
19  reduc adec(sk, aenc(pk(sk), sr, xm)) -> xm.  
20  
21  (* Signature *)  
22  
23  fun sign/2.  
24  fun vk/1.  
25  reduc checksign(vk(sk), sign(sk,m)) -> m.  
26
```



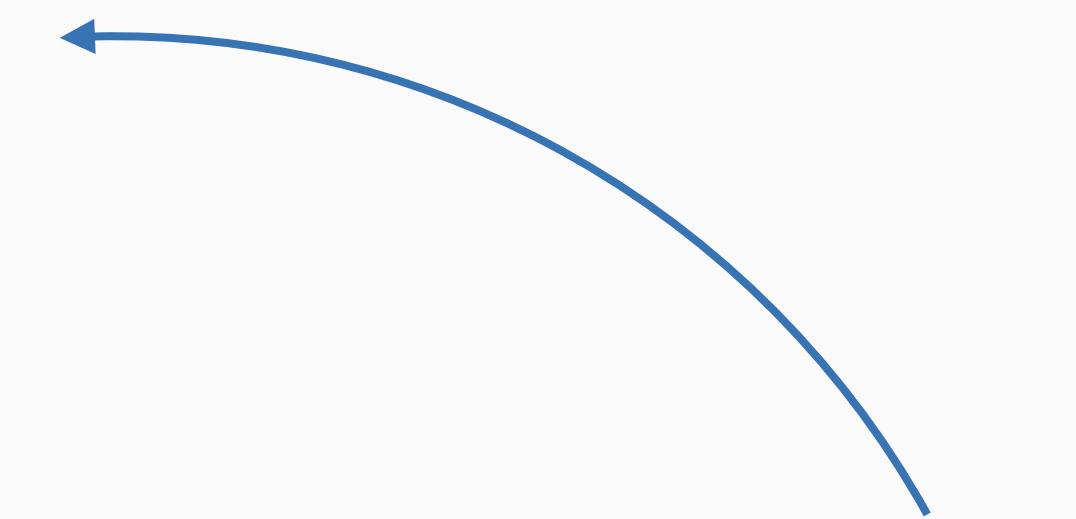
```
11   free yes.  
12   free no.  
13  
14 (* Randomized asymmetric encryption *)  
15  
16 fun aenc/3.  
17 fun pk/1.  
18  
19 reduc adec(sk, aenc(pk(sk), sr, xm)) -> xm.  
20  
21 (* Signature *)  
22  
23 fun sign/2.  
24 fun vk/1.  
25 reduc checksign(vk(sk), sign(sk,m)) -> m.
```



Definition of crypto primitives  
(here, asymmetric encryption)

```
36 fun s/1.  
37  
38 (* The voting process *)  
39  
40 let Voter(sk,id,v,pkE) =  
41     new r;  
42     let ballot = aenc(pkE, r, v) in  
43     let zk = zkp(r, id, v, ballot) in  
44     out(c, (id, sign(sk, (ballot, zk)))).  
45  
46 (* The Tally *)  
47  
48 let Outcome(prv_ch,skE) =  
49     in(prv_ch,z);  
50     let (vote1,vote2,vote3,nb_vote) = z in
```

```
36 fun s/1.  
37  
38 (* The voting process *)  
39  
40 let Voter(sk,id,v,pkE) =  
41     new r;  
42     let ballot = aenc(pkE, r, v) in  
43     let zk = zkp(r, id, v, ballot) in  
44     out(c, (id, sign(sk, (ballot, zk)))).  
45  
46 (* The Tally *)  
47  
48 let Outcome(prv_ch,skE) =  
49     in(prv_ch,z);  
50     let (vote1,vote2,vote3,nb_vote) = z in
```



Definition of the protocol  
(here, voting process)

```
new skE;
93   out(c,pk(skE));
94   new sk1;
95   new sk2;
96   new sk3;
97   out(c,sk3);
98   out(c,vk(sk1));
99   out(c,vk(sk2)); (
L00     !^2 Voter(sk1,id1,vote1,pk(skE)) |
L01     Voter(sk2,id2,vote2,pk(skE)) |
L02     Tally(skE,vk(sk1),vk(sk2),vk(sk3))
L03   ).  

L04
L05 (* Should not find an attack. *)
L06 query
• session_equiv(VotingSystem21(yes,no),VotingSystem21(no,yes)).  

L07
```



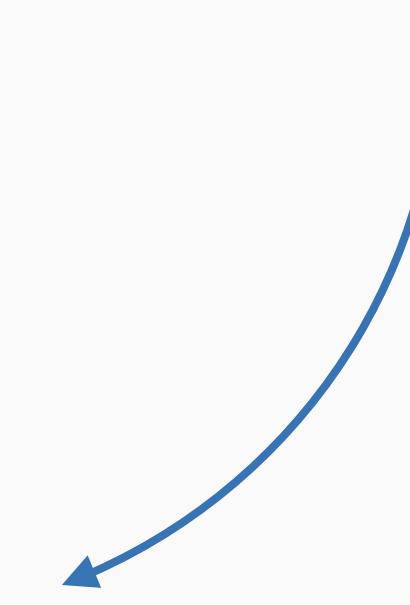
v2.0.0

```
new skE;
out(c,pk(skE));
new sk1;
new sk2;
new sk3;
out(c,sk3);
out(c,vk(sk1));
out(c,vk(sk2)); (
  !^2 Voter(sk1,id1,vote1,pk(skE)) |
  Voter(sk2,id2,vote2,pk(skE)) |
  Tally(skE,vk(sk1),vk(sk2),vk(sk3))
).
(* Should not find an attack. *)
query
• session_equiv(VotingSystem21(yes,no),VotingSystem21(no,yes)).
```



v2.0.0

Equivalence query (here:  
model of vote privacy)



DeepSec UI Dev Tools

DeepSec UI

ironumber  
epSec -  
Versio  
Git ha  
Git br  
Websit  
ading fi  
arting v  
arting v  
sult que  
rificati  
ironumber

DeepSec UI

Add file(s)... 1 file Clear file

Start Run

Results

Settings

Optional title

Start Run

Default Semantic : Private Classic Eavesdrop

Distributed : Auto Yes No

Number jobs :  Auto

Local workers :  Auto 2

Round timer : 120

Total worker 19

Reset

+ Add Distant Server 1

Hostname : shironumber@cassis-calc.loria.fr

Local path : ./deepsec

Workers :  Auto 17

DEEPSEC v2.0.0

23 / 27

DeepSec UI Dev Tools

DeepSec UI

ironumber  
epSec -  
Versio  
Git ha  
Git br  
Websit  
ading fi  
arting v  
arting v  
sult que  
rificati  
ironumber

DeepSec UI

Add file(s)...  
Helios\_revote\_ZKP21

Start Run

Results

Settings

Optional title

Start Run

Default Semantic :  
Private   Classic   Eavesdrop

Distributed :  
Auto   Yes   No

Number jobs :  Auto

Local workers :  Auto   2

Round timer : 120

Total worker 19

Reset

1 file   Clear file

+ Add Distant Server 1

Hostname : shironumber@cassis-calc.loria.fr

Local path : ./deepsec

Workers :  Auto   17

DEEPSEC v2.0.0

Browse files to verify

100 % Fri 11:48

23 / 27

The screenshot displays the DeepSec UI application window. On the left, there's a sidebar with navigation links: 'ironumber', 'epSec -', 'Versio', 'Git ha', 'Git br', 'Websit', 'ading fi', 'arting v', 'arting v', 'sult que', 'rificati', and 'ironumber'. The main area has a header 'DeepSec UI' and a sub-header 'DeepSec UI'. It features a central modal for file selection with a 'Add file(s)...' button and a list containing 'Helios\_revote\_ZKP21'. Below the modal are sections for 'Default Semantic' (with tabs for 'Private', 'Classic', and 'Eavesdrop'), 'Distributed' (with tabs for 'Auto', 'Yes', and 'No'), and configuration for 'Number jobs' (set to 'Auto'), 'Local workers' (set to 'Auto' with value '2'), and 'Round timer' (set to '120'). To the right, there's a 'Distant Server' configuration panel with fields for 'Hostname' (shironumber@cassis-calc.loria.fr), 'Local path' (./deepsec), and 'Workers' (set to 'Auto' with value '17'). A large blue arrow originates from the text 'Browse files to verify' at the bottom left and points to the 'Add file(s)...' button in the modal. The top of the slide features a blue banner with the text 'DEEPSEC v2.0.0'.

The screenshot shows the DeepSec UI application interface. On the left, a sidebar lists various project-related links. The main area has a title bar "DeepSec UI". It features a file upload section with a blue "Add file(s)..." button and a list containing "Helios\_revote\_ZKP21". To the right of this is a "Default Semantic" dropdown with "Private" selected, and tabs for "Classic" and "Eavesdrop". Below this are sections for "Distributed" (with "Yes" selected), "Number jobs" (checkbox checked, "Auto" selected), "Local workers" (checkbox checked, "Auto" selected, value "2"), and "Round timer" (value "120"). A "Total worker 19" summary is shown below these. At the bottom is a "Reset" button. To the right of the main panel is a sidebar titled "DEEPSEC" with "v2.0.0" and a "Clear file" button. A large blue arrow points from the text "Browse files to verify" towards the file upload section. Another large blue arrow points from the text "Options for distributing The computation" towards the "Distributed" and "Local workers" settings.

DeepSec UI Dev Tools

DeepSec UI

Add file(s)...

1 file

Clear file

DeepSec - Version Git ha Git br Websit

ading fi arting v arting v sult que rificati ironumbe

ironumber epSec - Version Git ha Git br Websit

Browse files to verify

Start Run

Results

Settings

Optional title

Start Run

Default Semantic :

Private Classic Eavesdrop

Distributed :

Auto Yes No

Number jobs :  Auto

Local workers :  Auto 2

Round timer : 120

Total worker 19

Reset

+ Add Distant Server 1

Hostname : shironumber@cassis-calc.loria.fr

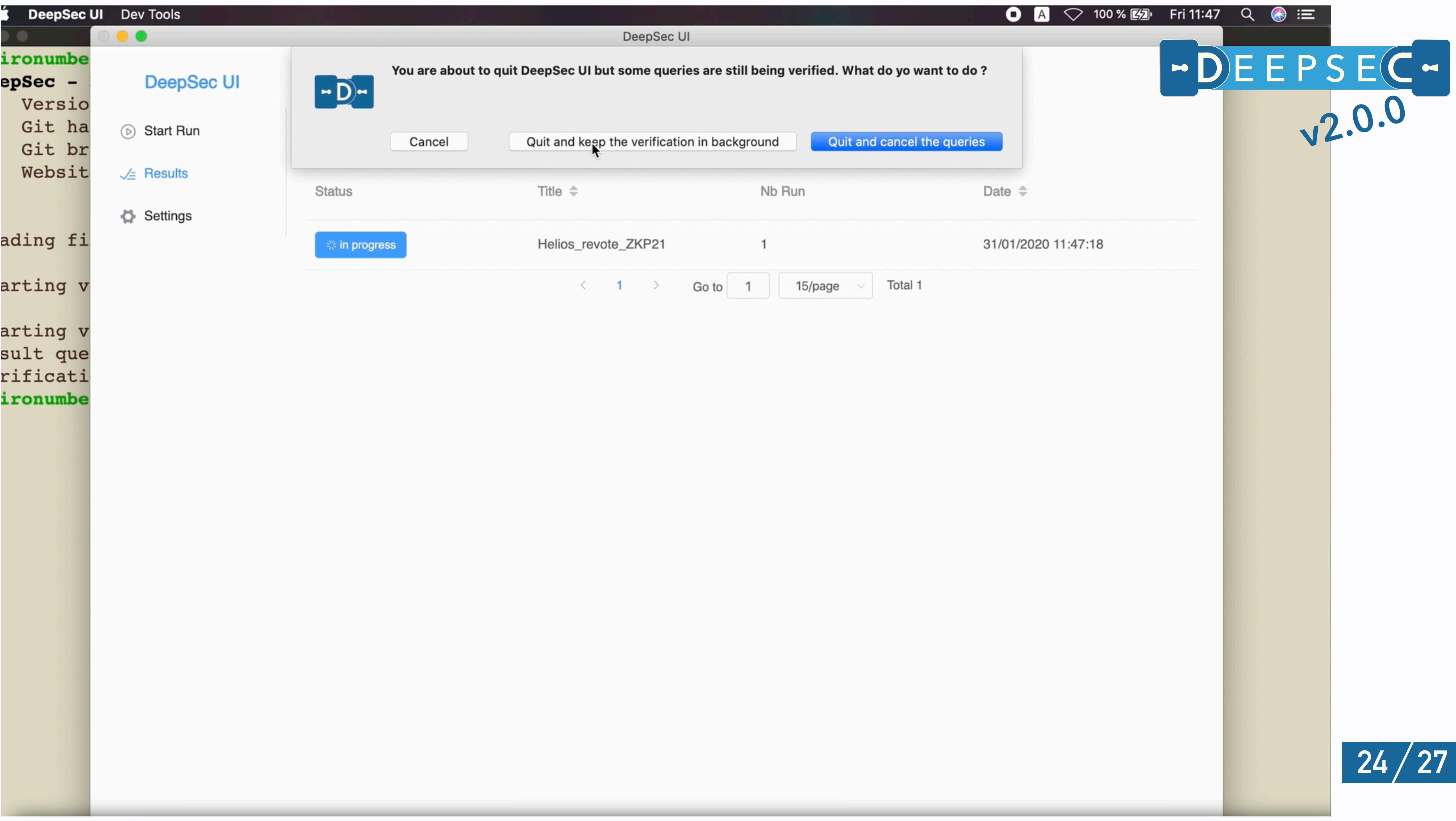
Local path : ./deepsec

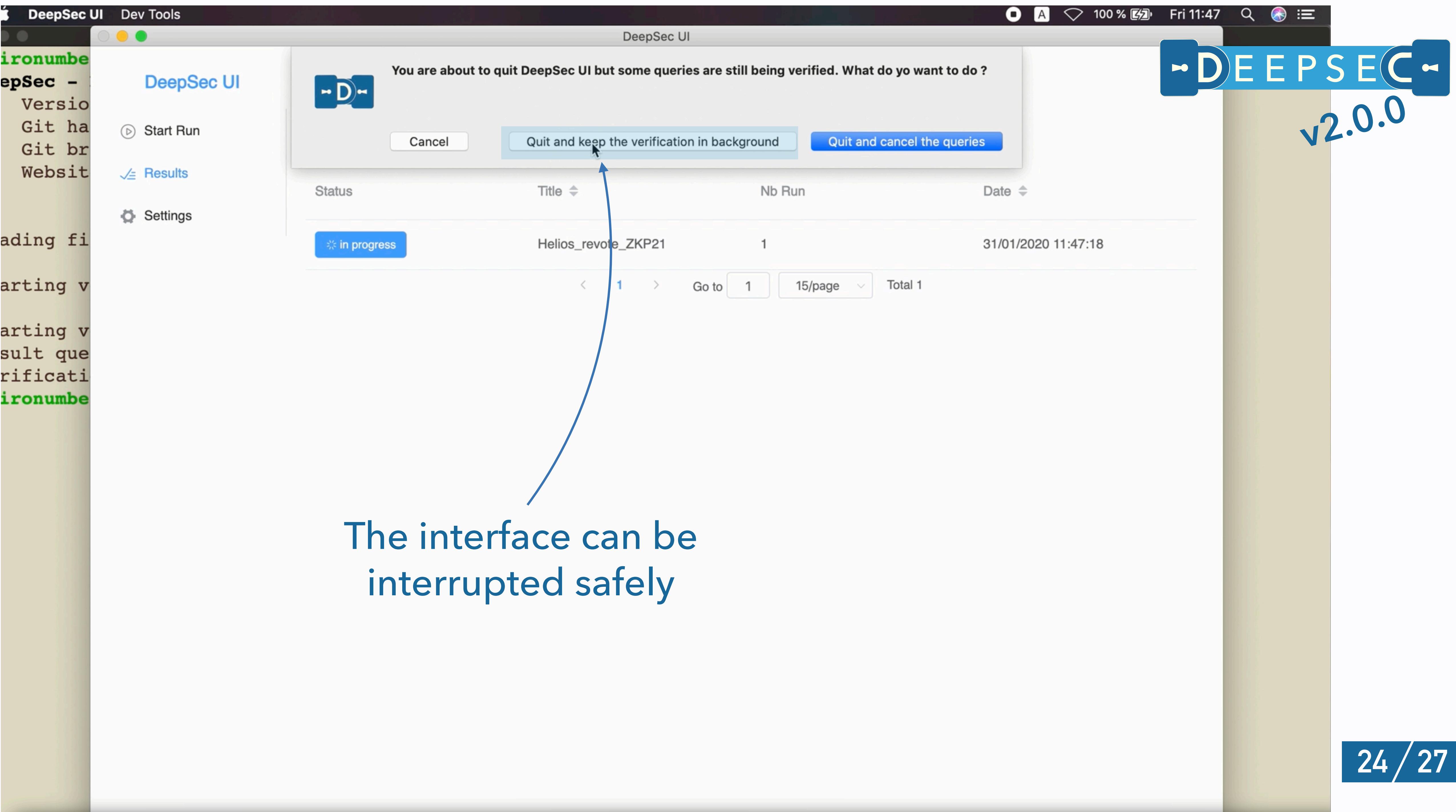
Workers :  Auto 17

DEEPSEC v2.0.0

Clear file

23 / 27





DeepSec UI Dev Tools

DeepSec UI

Attack on Process 2

```
1 | ( 
2 |   in(mn,x1);
3 |   in(mn,x2);
4 |   in(mn,x3);
5 |   (
6 |     out(ch,adec(x3,skT))
7 |   ) | (
8 |     out(ch,adec(x1,skT))
9 |   ) | (
10|     out(ch,adec(x2,skT))
11|   )
12|   ) | (
13|     out(ch,(a,aenc(no,r,pk(skT)))) red box
14|   ) | (
15|     new r2;
16|     let ballot2 = aenc(yes,r2,pk(skT)) in
17|       out(bb,(b,ballot2));
18|       out(ch,(b,ballot2))
19|   ) | (
20|     in(ch,b3);
21|     let (=c,v3) = b3 in
22|       out(mn,v3)
23|   ) | (
24|     let (=a,v1) = (a,aenc(no,r,pk(skT))) in
25|       out(mn,v1)
26|   ) | (
27|     in(bb,b2);
28|     let (=b,v2) = b2 in
29|       out(mn,v2)
30|   )
```

DEEPSEC v2.0.0

Default I/O All

« < Prev Next > »

Trace - Step 8 / 21

- 5 - out(ch,ax<sub>1</sub>)
- 8 -  $\tau$  internal communication
- 9 - out(ch,ax<sub>2</sub>)

Frame

Private names: bb, r, mn, skT

ax<sub>1</sub> → pk(skT)

25 / 27

DeepSec UI Dev Tools

DeepSec UI

Attack on Process 2

```
1 | ( 
2 |   in(mn,x1);
3 |   in(mn,x2);
4 |   in(mn,x3);
5 |   (
6 |     out(ch,adec(x3,skT))
7 |   ) | (
8 |     out(ch,adec(x1,skT))
9 |   ) | (
10|     out(ch,adec(x2,skT))
11|   )
12|   ) | (
13|     out(ch,(a,aenc(no,r,pk(skT)))))
14|   ) | (
15|     new r2;
16|     let ballot2 = aenc(yes,r2,pk(skT)) in
17|       out(bb,(b,ballot2));
18|       out(ch,(b,ballot2))
19|   ) | (
20|     in(ch,b3);
21|     let (=c,v3) = b3 in
22|       out(mn,v3)
23|   ) | (
24|     let (=a,v1) = (a,aenc(no,r,pk(skT))) in
25|       out(mn,v1)
26|   ) | (
27|     in(bb,b2);
28|     let (=b,v2) = b2 in
29|       out(mn,v2)
30|   )
```

DEEPSEC v2.0.0

Default I/O All

« < Prev Next > »

Trace - Step 8 / 21

5 - out(ch,ax<sub>1</sub>)  
8 - τ internal communication  
9 - out(ch,ax<sub>2</sub>)

Frame

Private names: bb, r, mn, skT

ax<sub>1</sub> → pk(skT)

When query finished: result dynamically displayable (here: attack trace)

25 / 27

# Conclusion

# Conclusion

- + A new equivalence exploiting the structure of practical privacy statements
- + Decision of trace equivalence improved by orders of magnitude on concrete examples
- Future work: Complete procedure for trace equivalence guided by equivalence by session