Kryptografie: Mit Mathematik gegen Hacker und die NSA

Ueli Maurer

Departement Informatik, ETH Zürich

27. Schweiz. Tag über Mathematik und Unterricht, KS Wil, 7. Sept. 2016

Ziele des Vortrags

Ziele des Vortrags

1. Mathematik mit zentral wichtigen Anwendungen

Ziele des Vortrags

1. Mathematik mit zentral wichtigen Anwendungen

2. Coole mathematische Probleme
   in Mittelschulreichweite

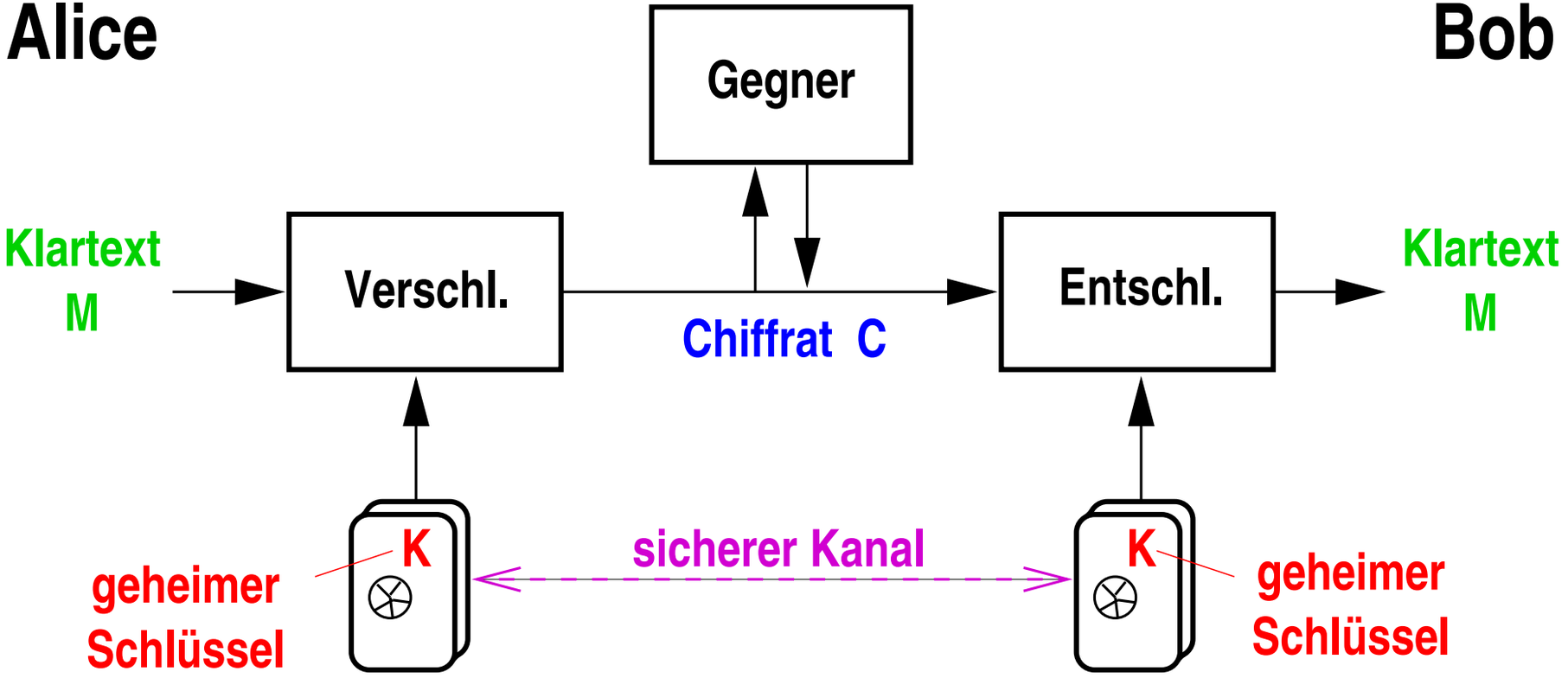
Ziele des Vortrags

1. Mathematik mit zentral wichtigen Anwendungen

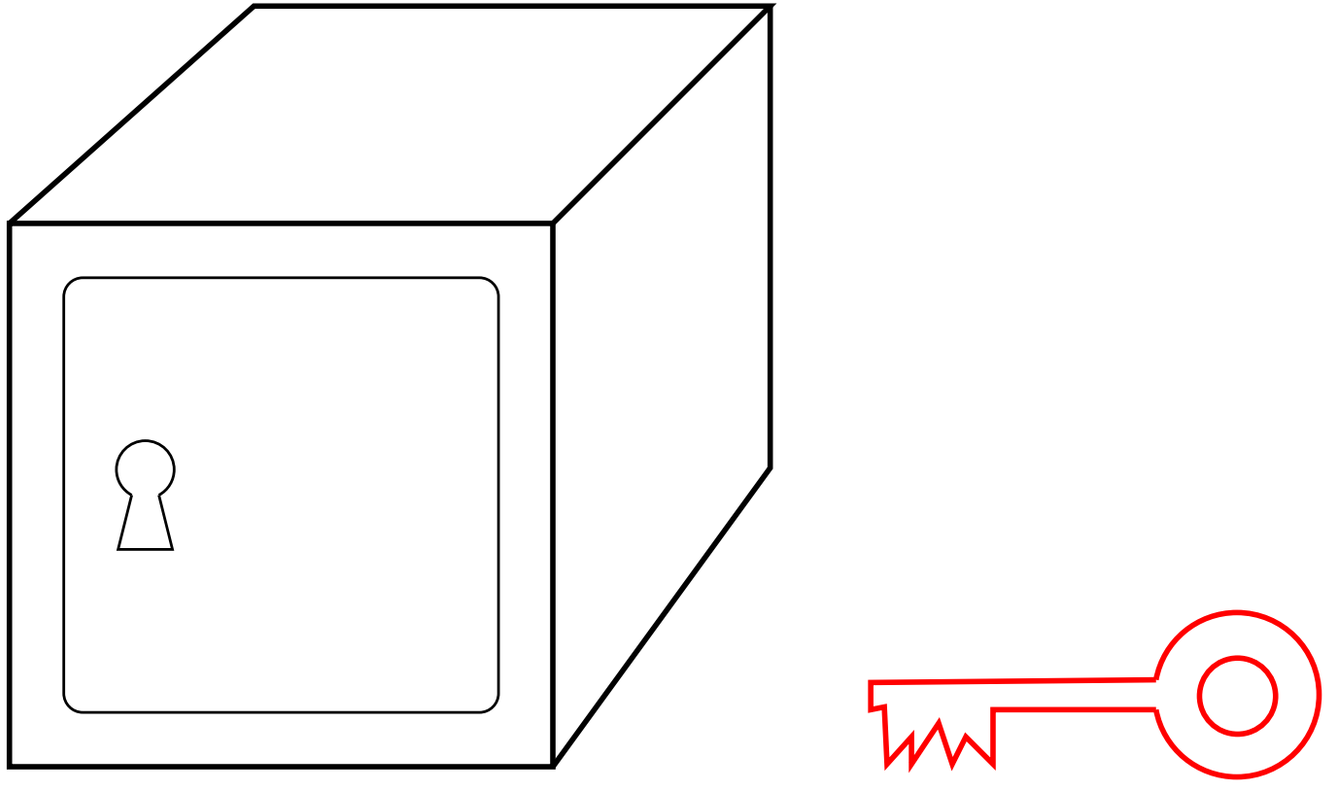
2. Coole mathematische Probleme
   in Mittelschulreichweite

3. Werbung für das ETH-Informatikstudium

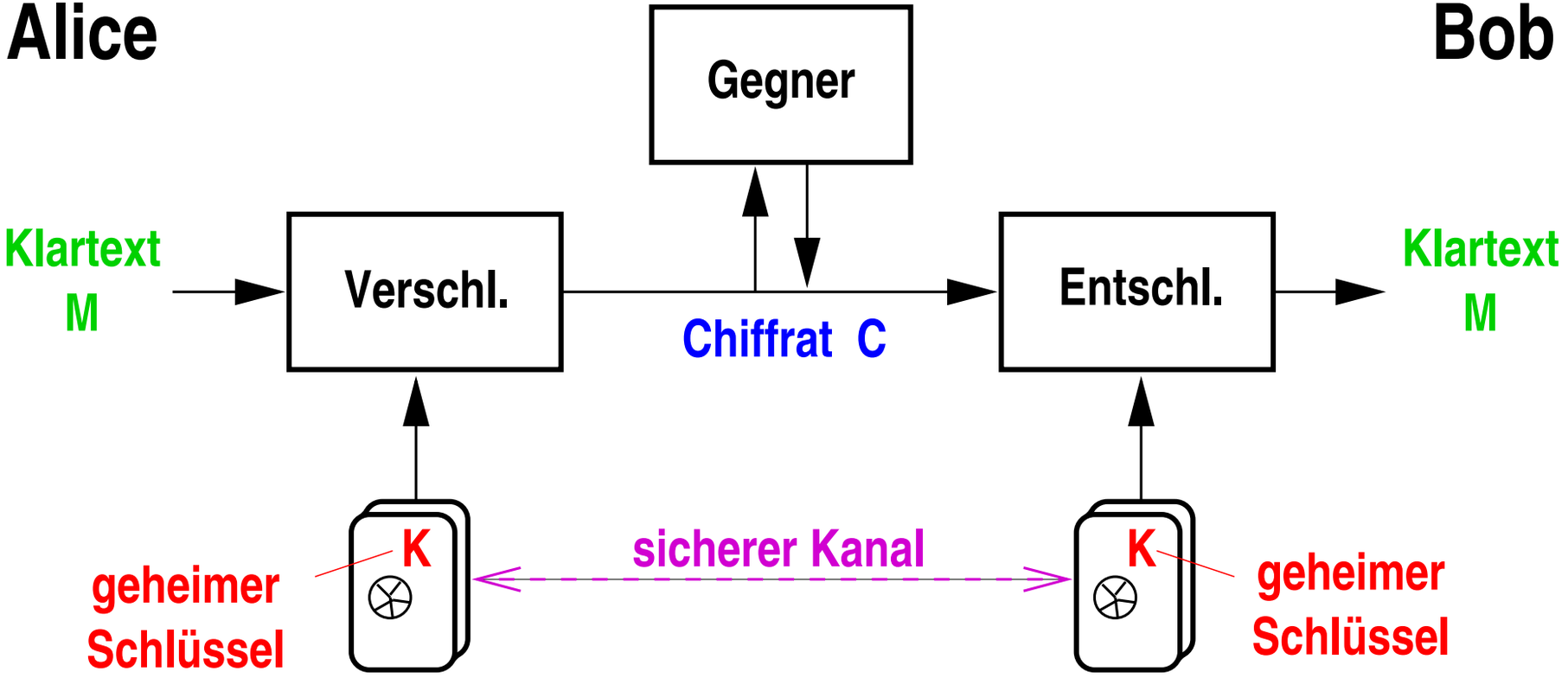
Verschlüsselungssystem

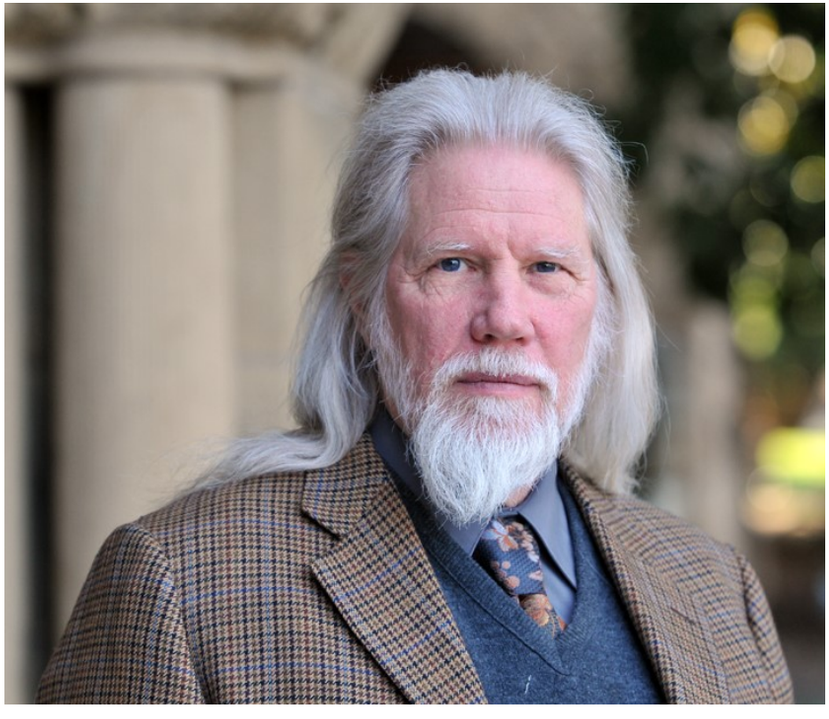


Verschlüsselung: mechanisches Analogon



Verschlüsselungssystem





Whitfield Diffie

Martin Hellman

Faktorisierung von Zahlen

$$323 = 17 \cdot 19$$

Faktorisierung von Zahlen

1098227363564981025449827

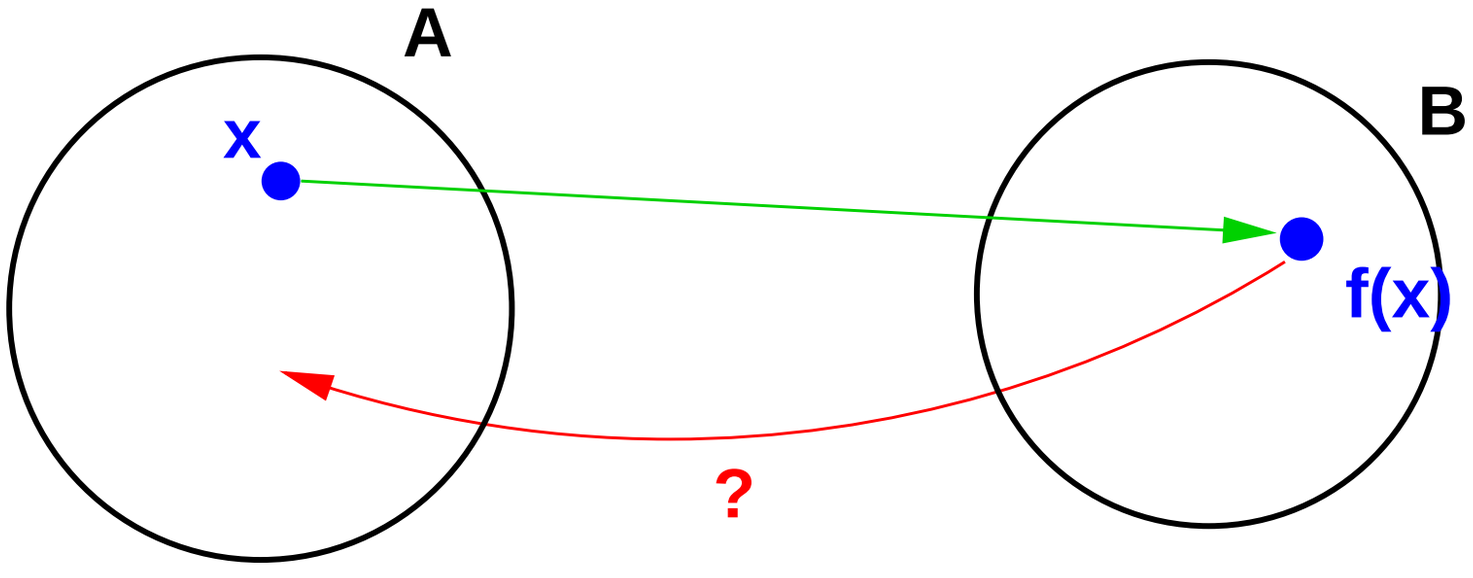
364098583475109184536287 6

341743634934764453302938 2

746711723976589866033564 3

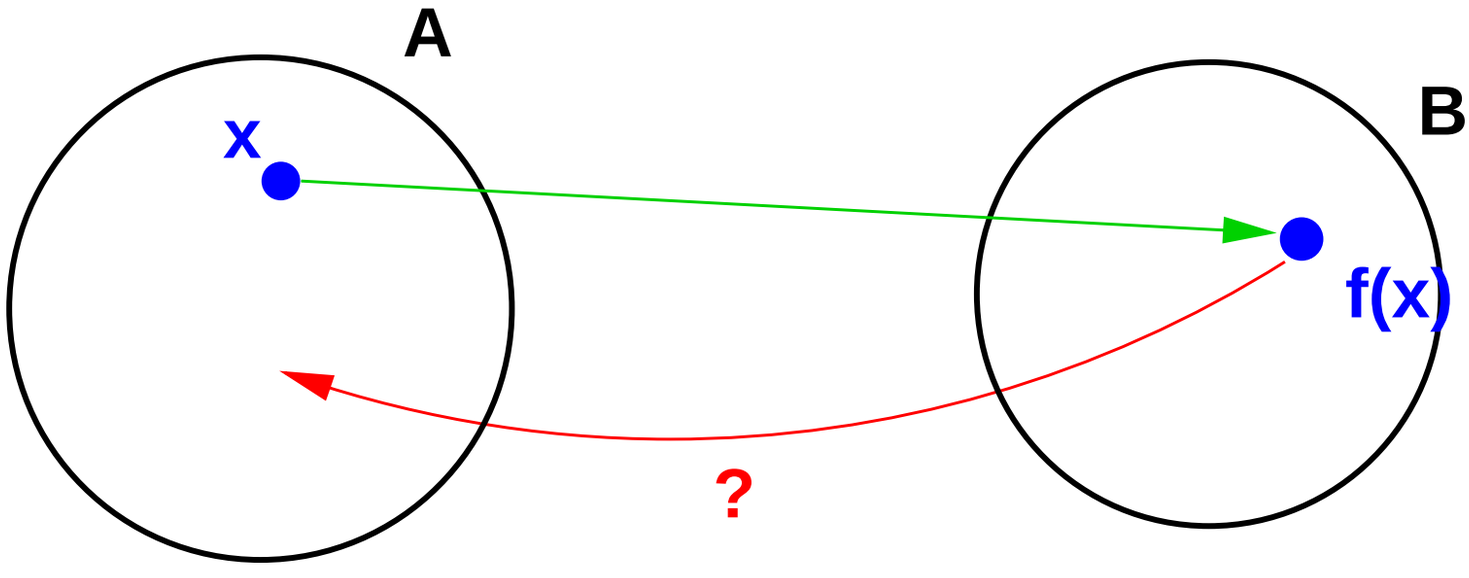
893364751029800354637562 3 = ? · ?

Einwegfunktion f



f(x) ist einfach berechenbar für jedes x.

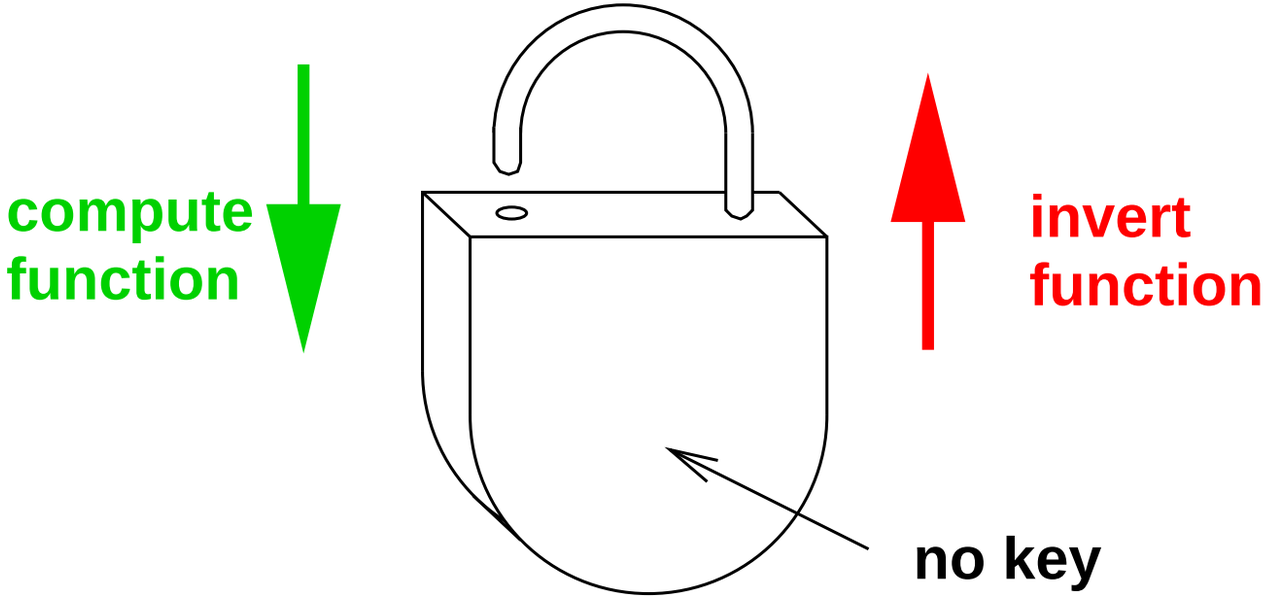
Einwegfunktion f



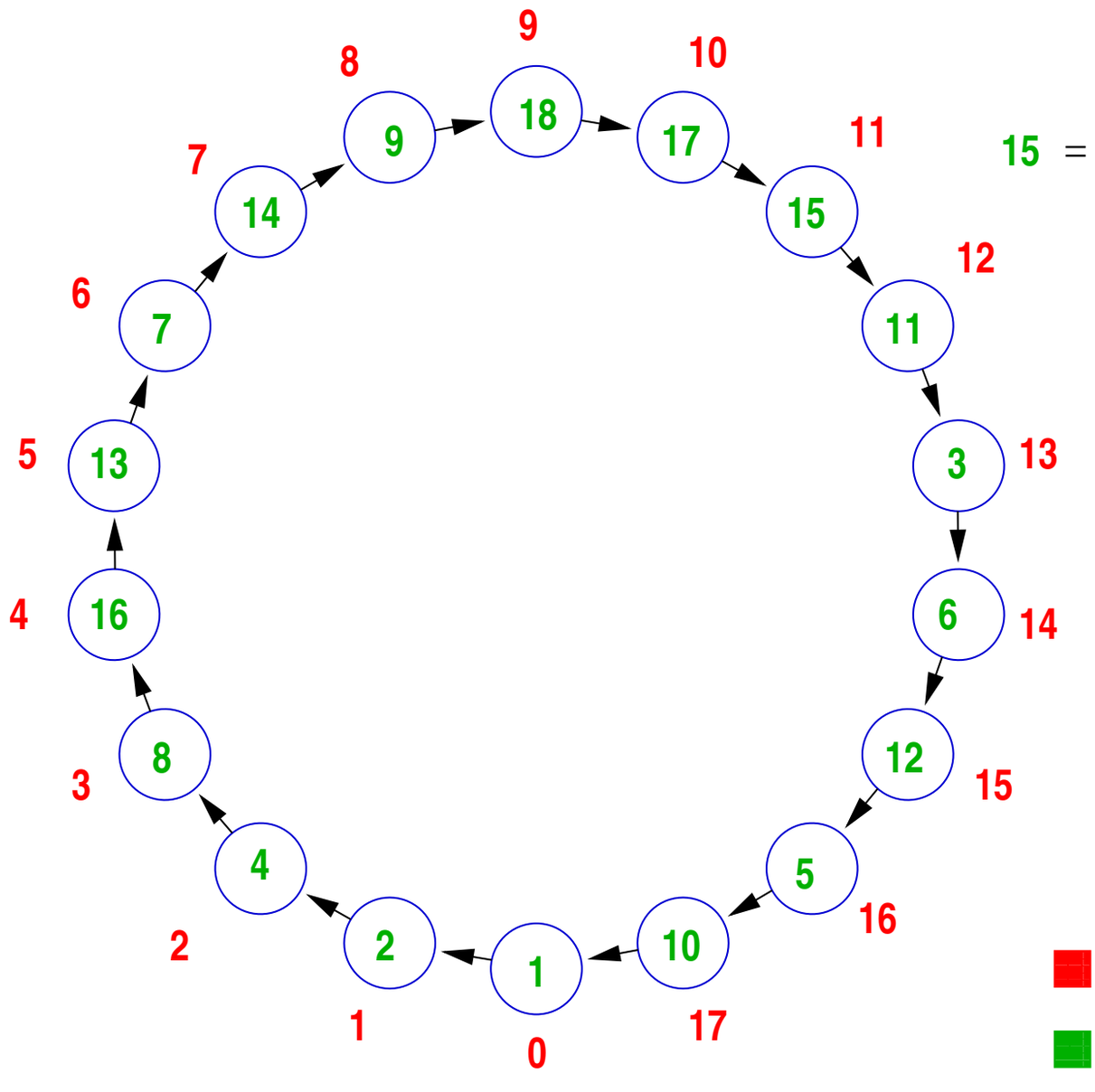
f(x) ist einfach berechenbar für jedes x.

Für zufälliges y ist es berechenmässig zu schwierig, ein x mit f(x) = y zu finden.

Einwegfunktion: Mechanisches Analogon



Potenzierung modulo 19

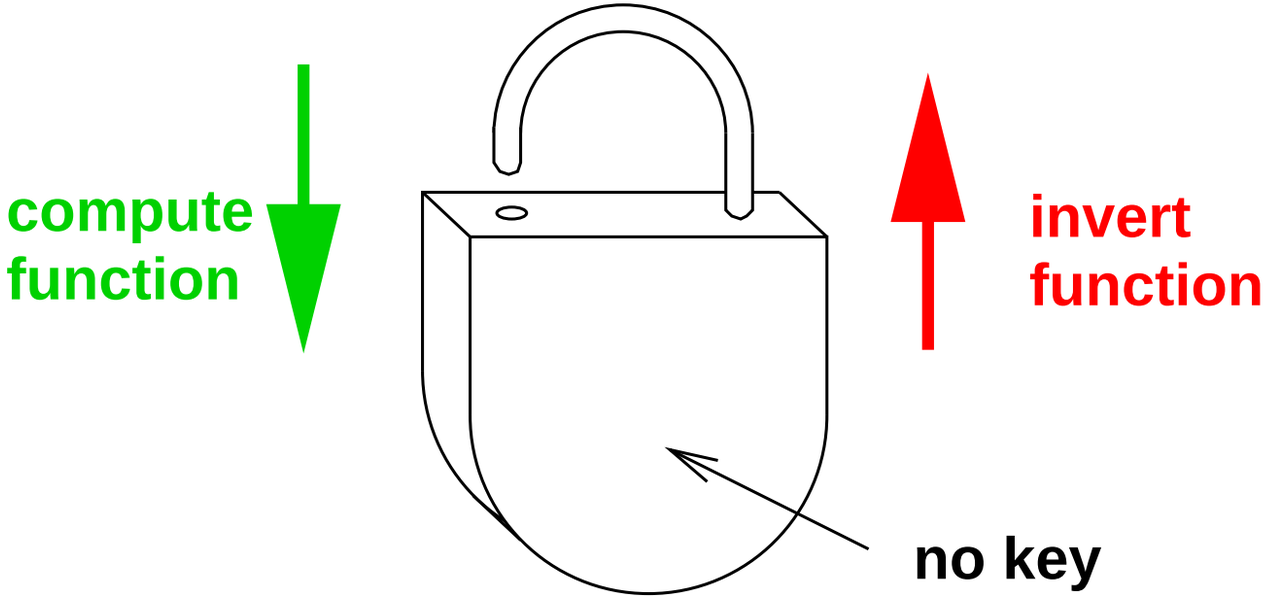


$15 = 2^{11} \pmod{19}$

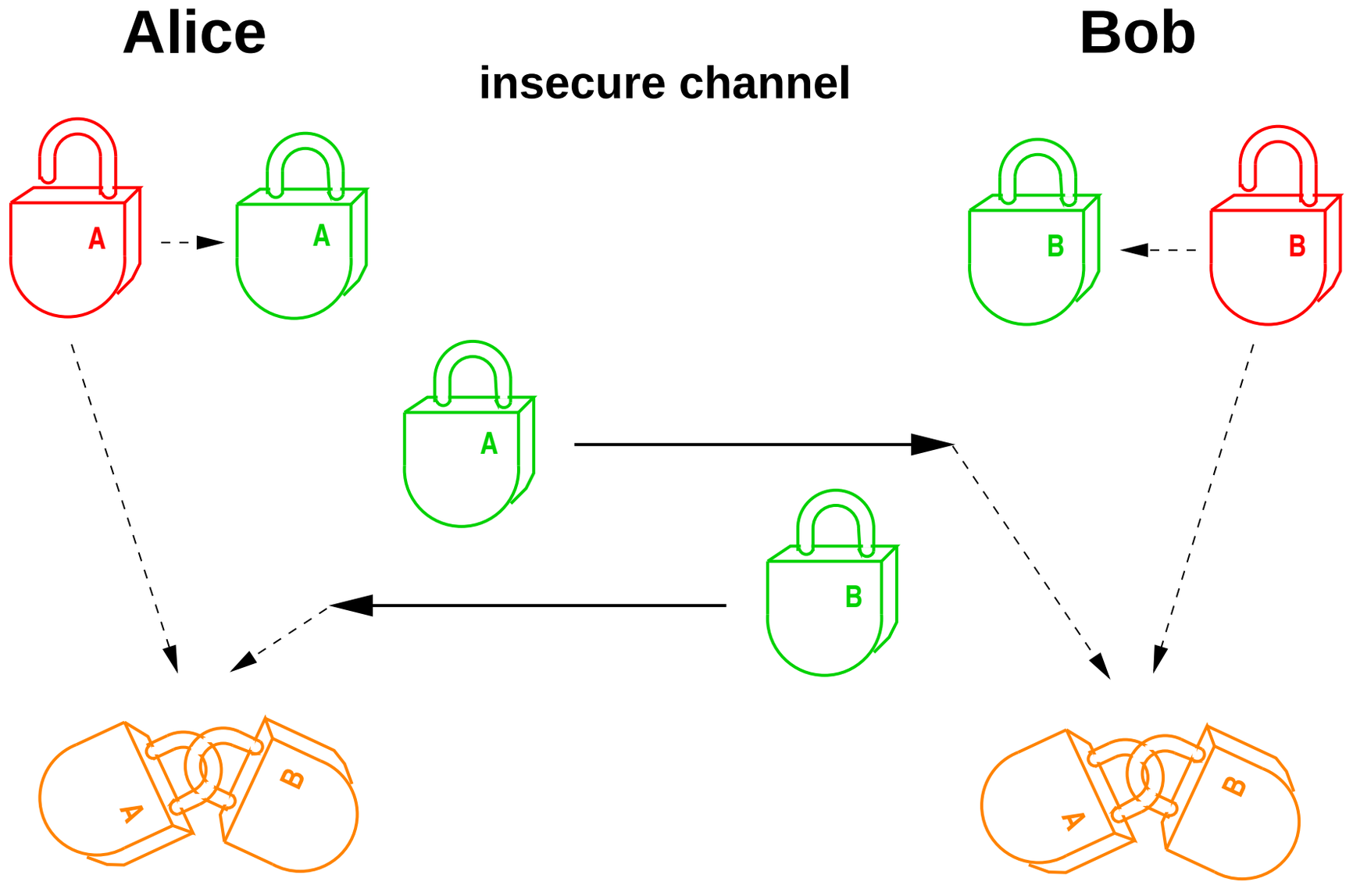
discrete logarithm

element of group

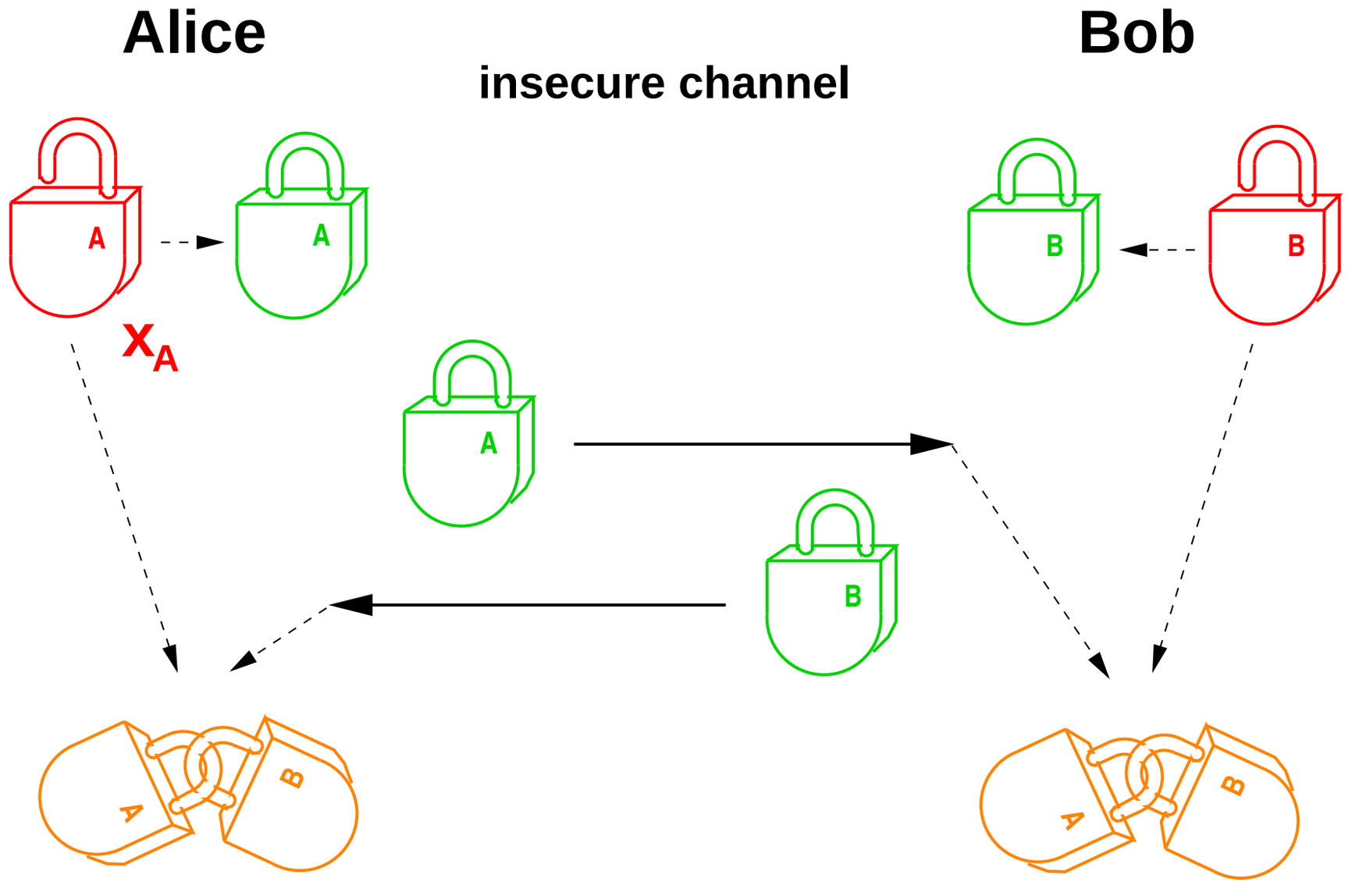
Einwegfunktion: Mechanisches Analogon



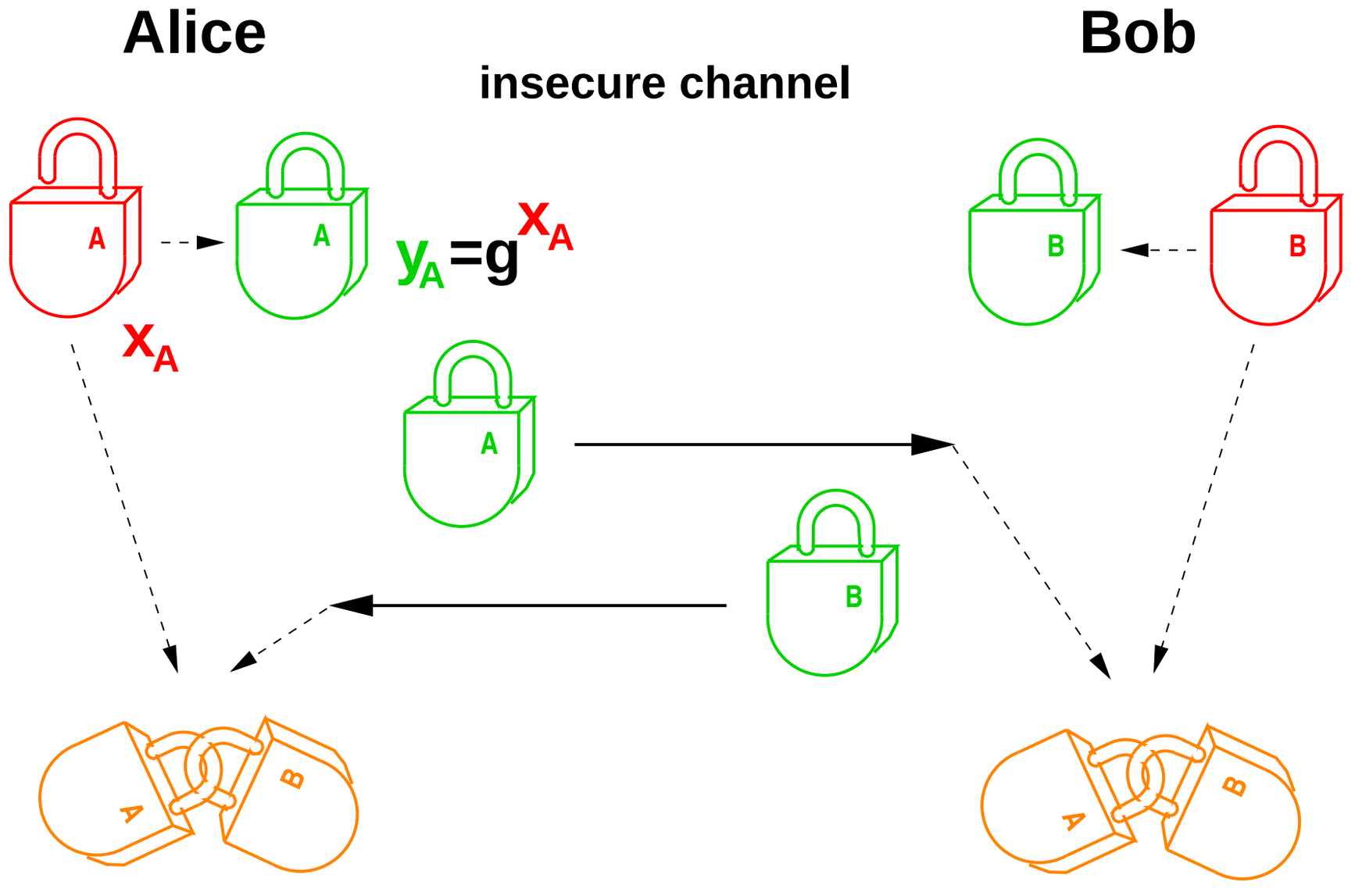
Diffie-Hellman Protokoll:



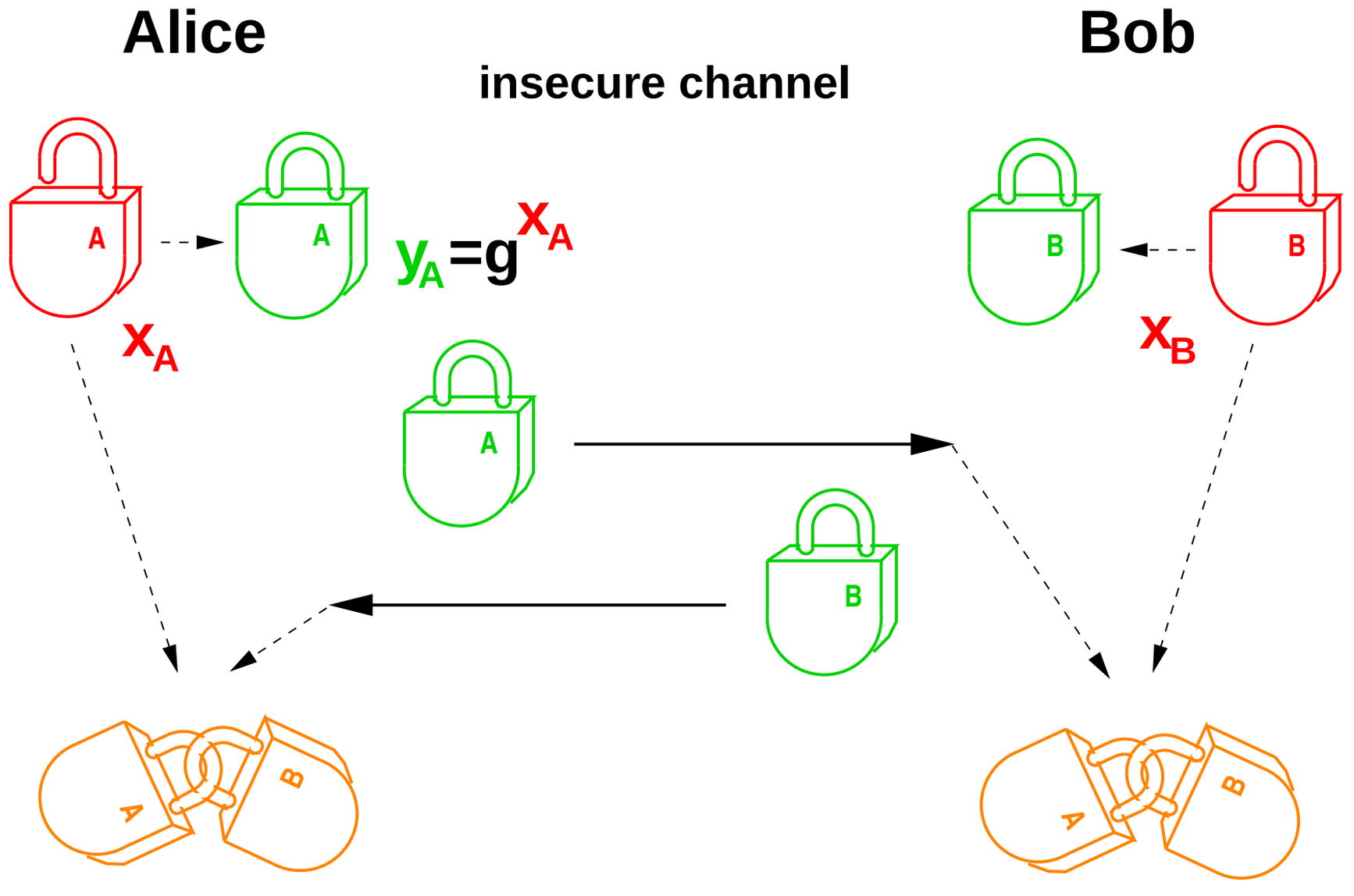
Diffie-Hellman Protokoll:



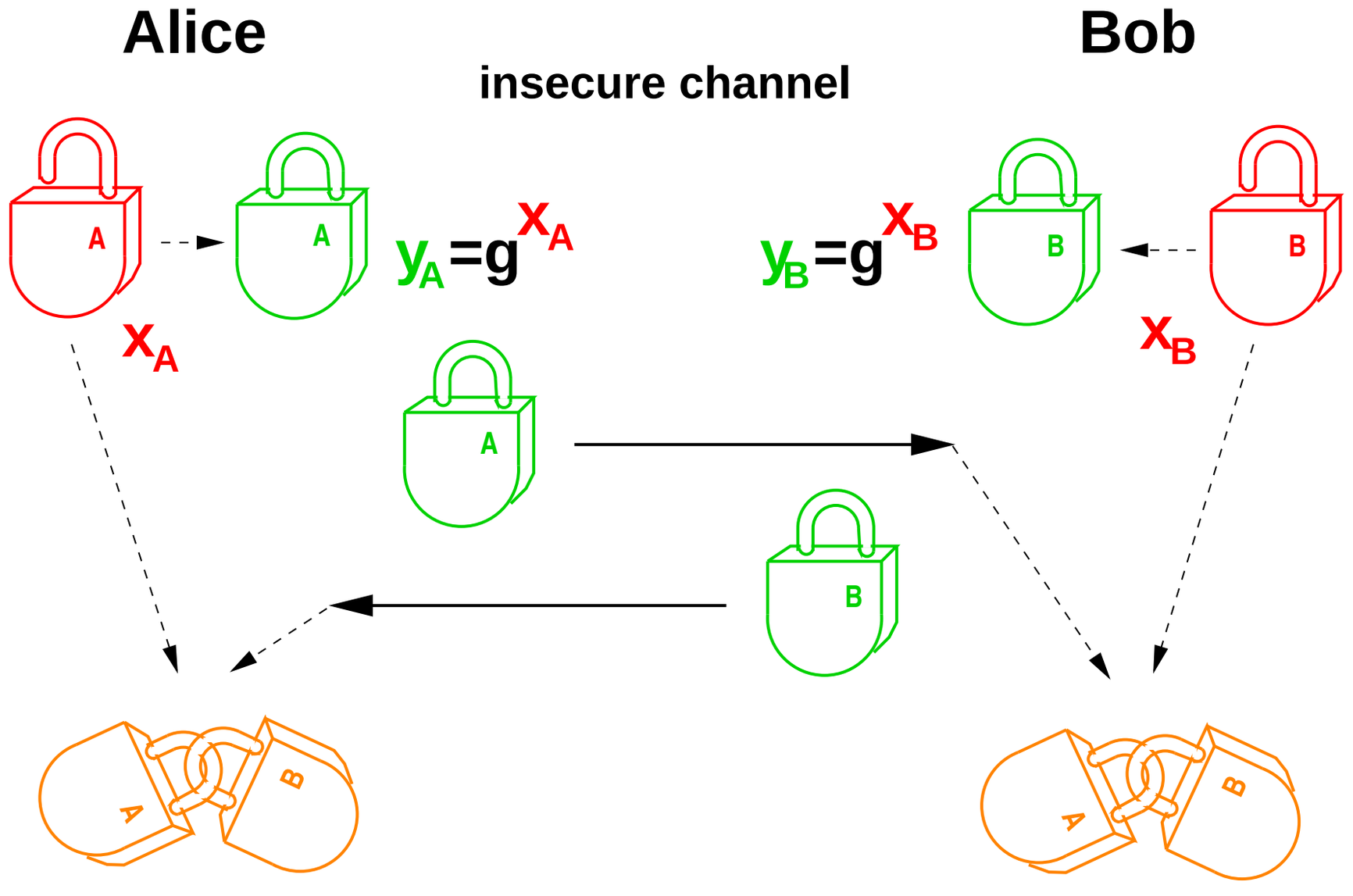
Diffie-Hellman Protokoll:



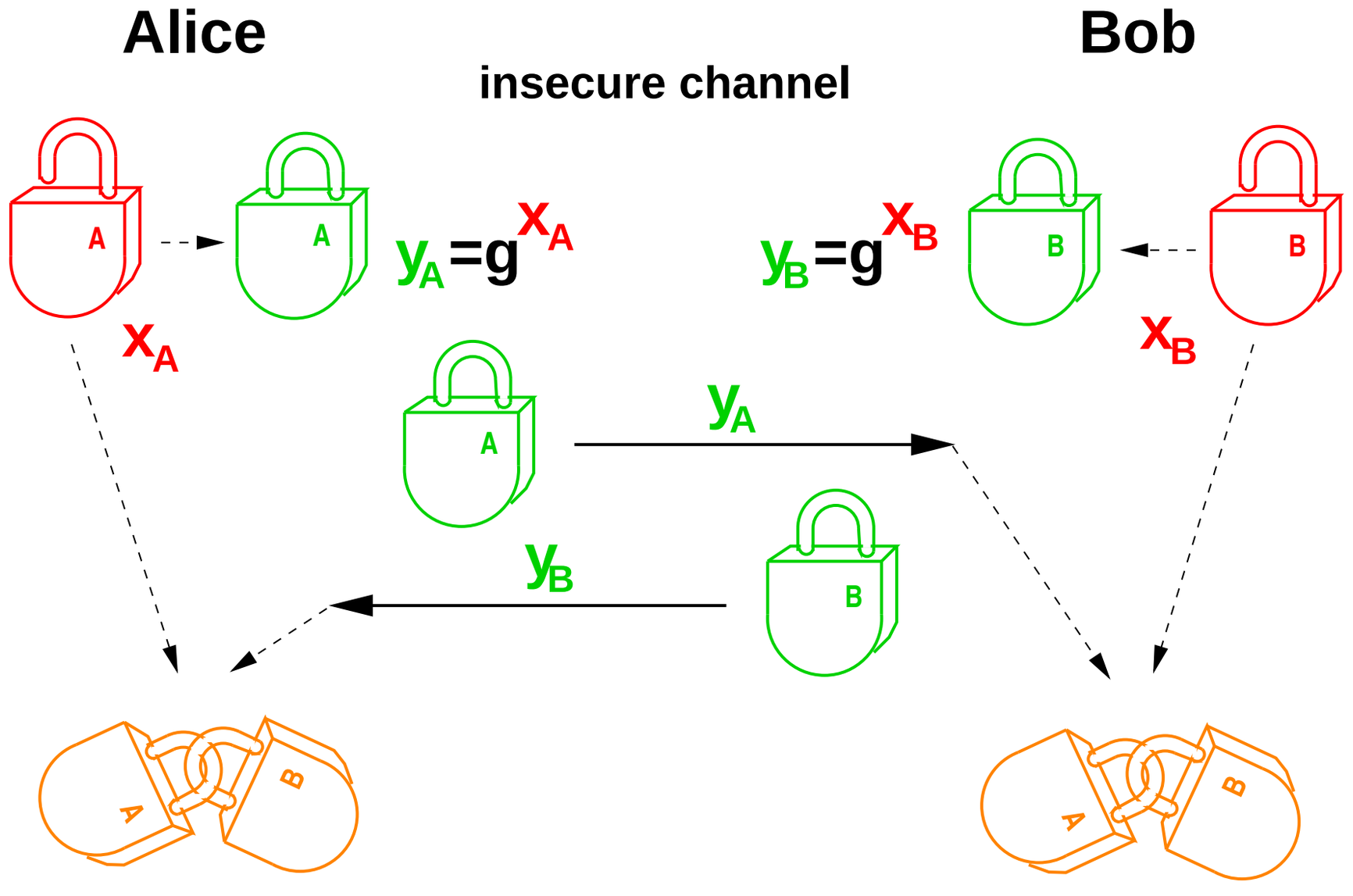
Diffie-Hellman Protokoll:



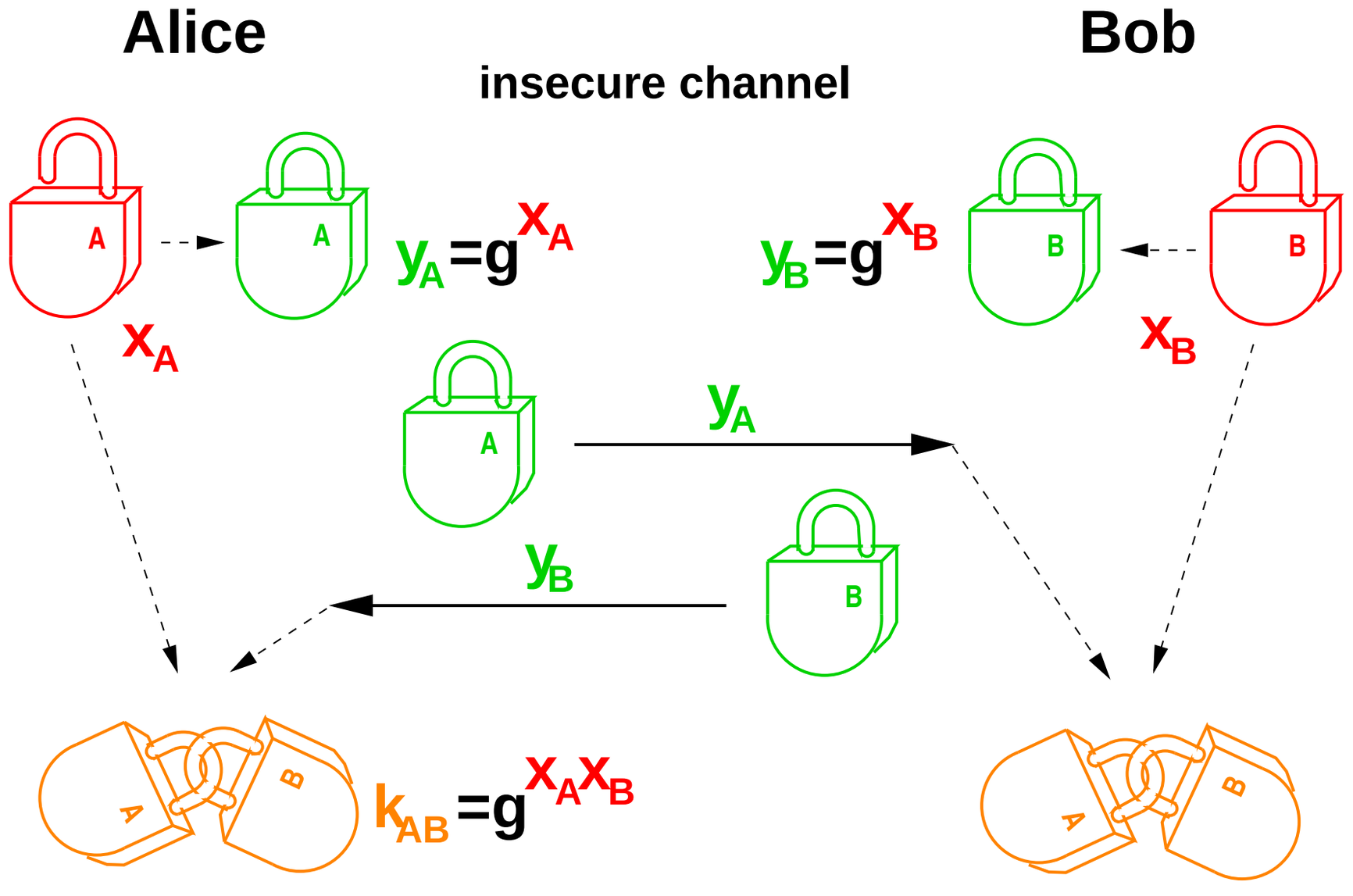
Diffie-Hellman Protokoll:



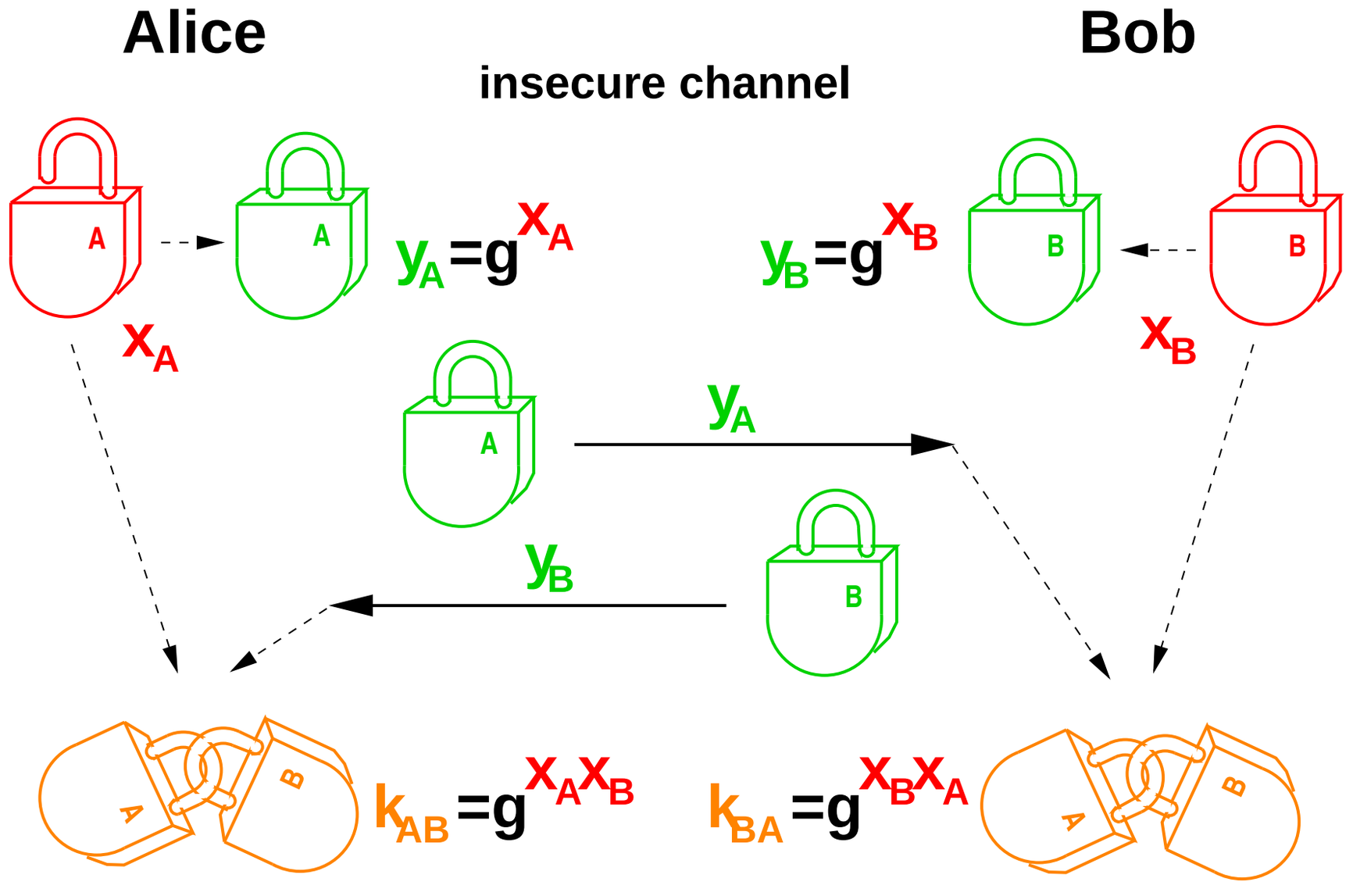
Diffie-Hellman Protokoll:



Diffie-Hellman Protokoll:



Diffie-Hellman Protokoll:



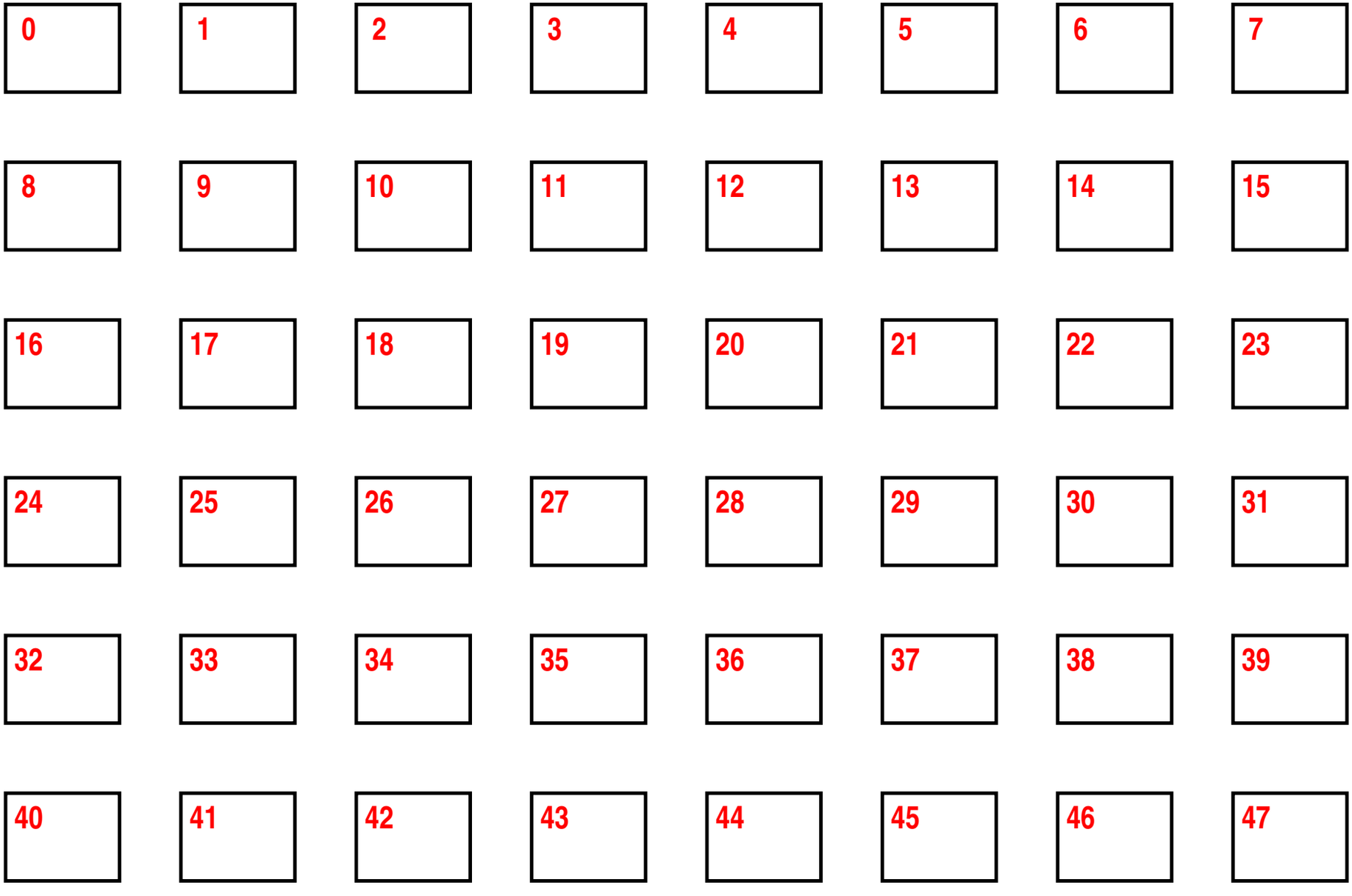
Discrete logarithm (DL) problem

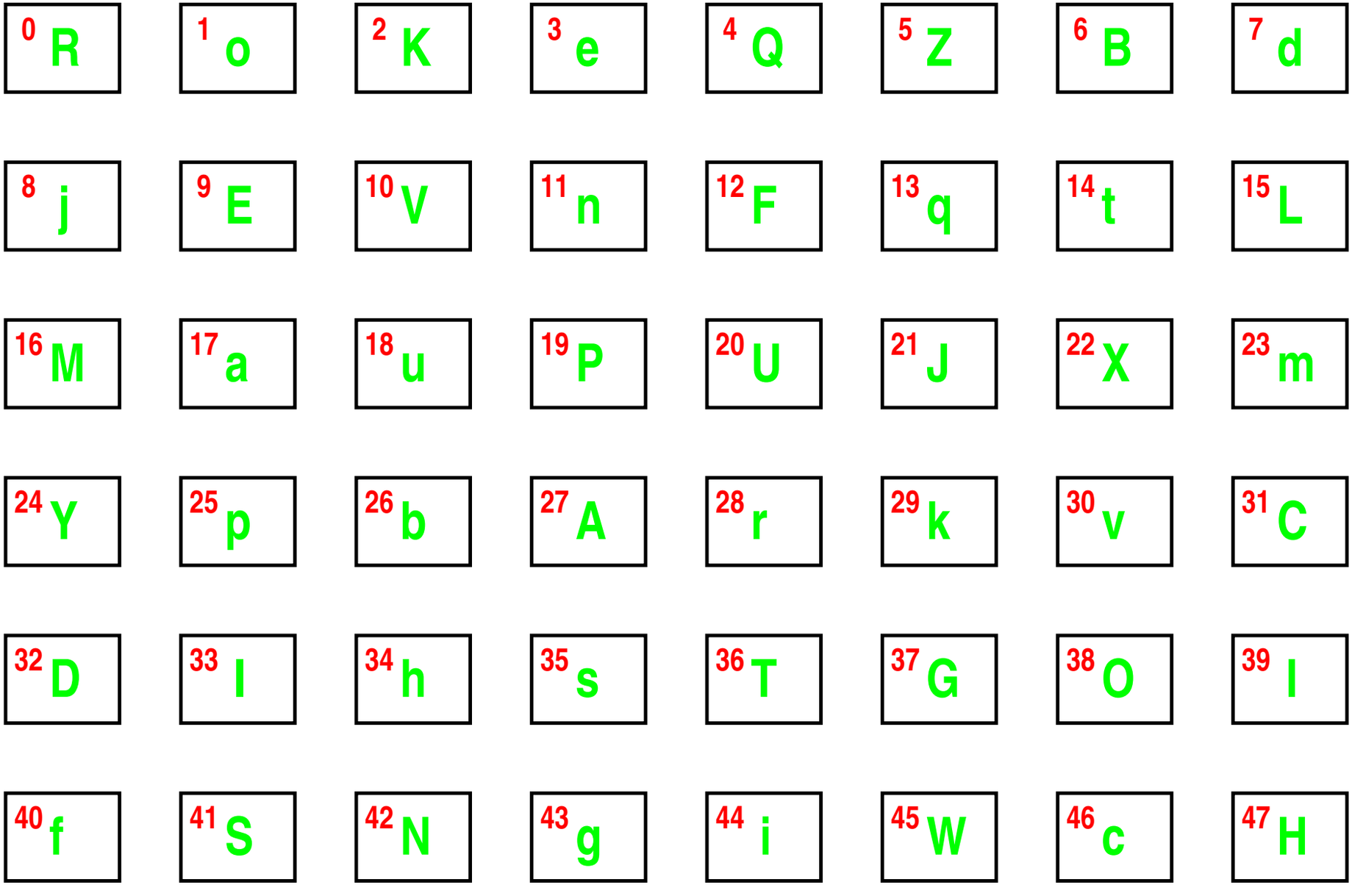
Cyclic group G of order n with generator g:

$$\mathrm{G} = \langle \mathbf{g} \rangle = \{\mathbf{g}^i : 0 \leq i < n\}$$

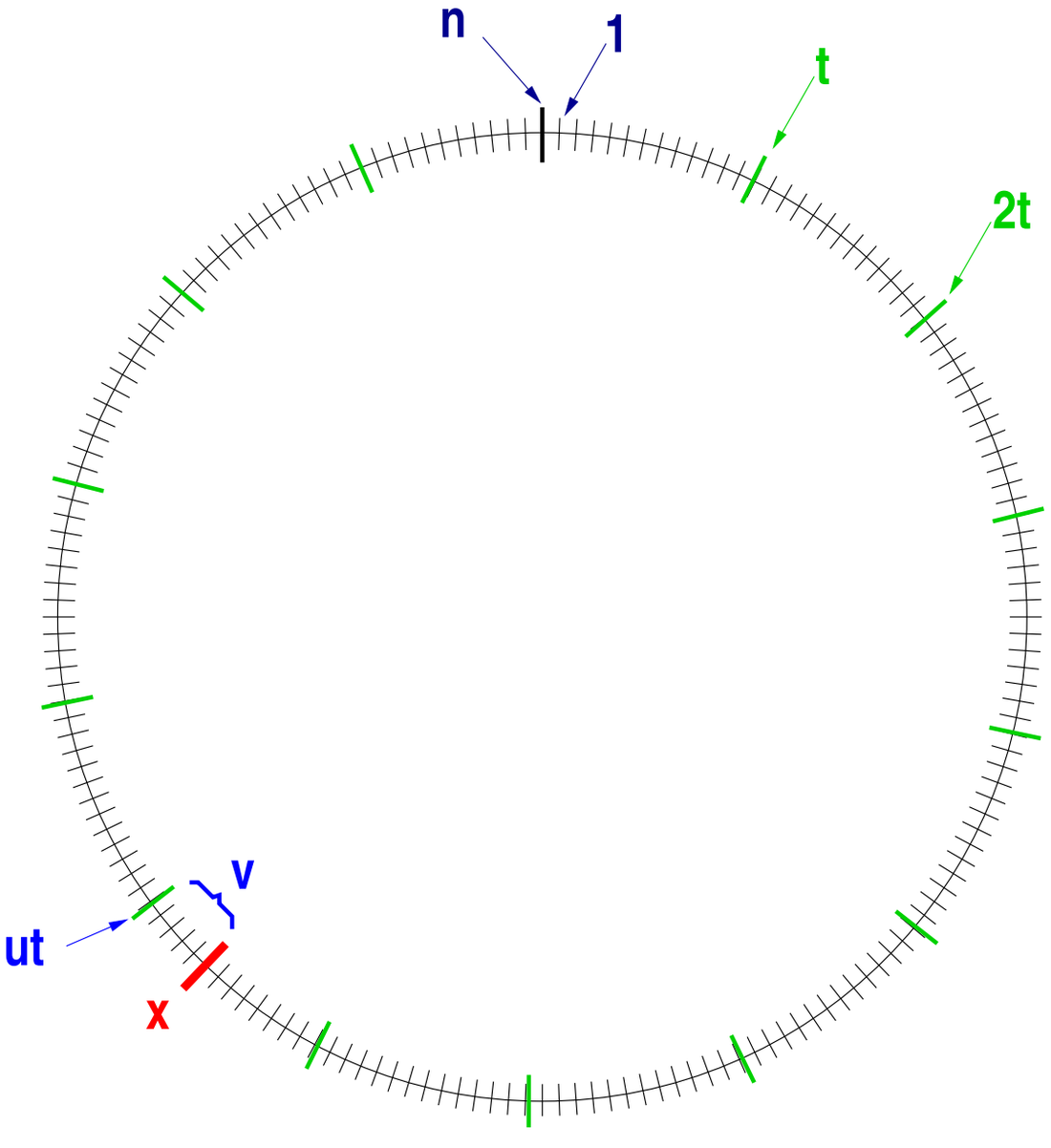
DL problem: Given $\mathrm{a} \in \mathrm{G}$, find x such that $\mathrm{a} = \mathbf{g}^{\mathbf{x}}$.

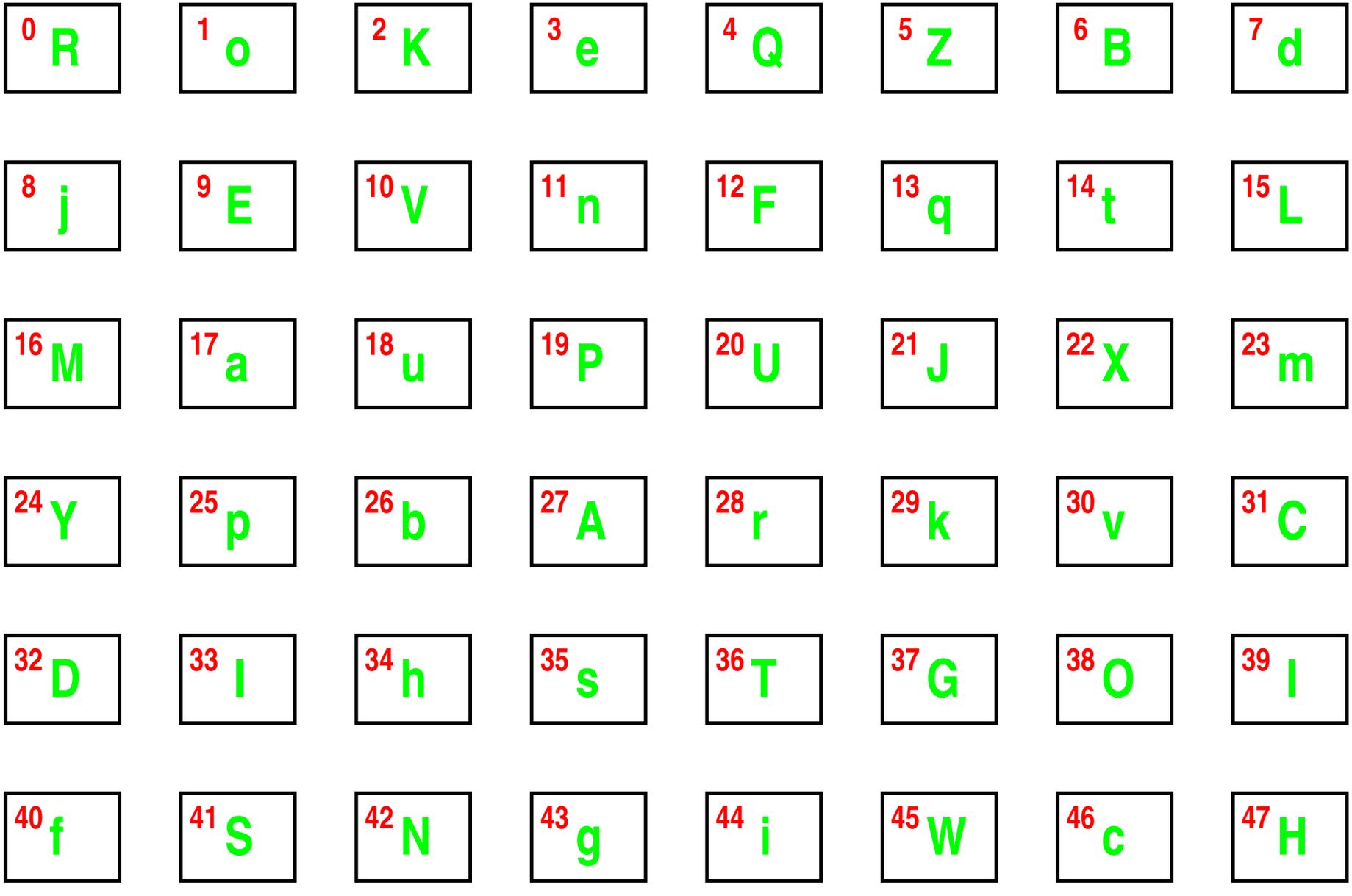
$\mathbf{x} \longrightarrow \mathbf{g}^{\mathbf{x}}$ is for many groups believed to be a OWF.


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47

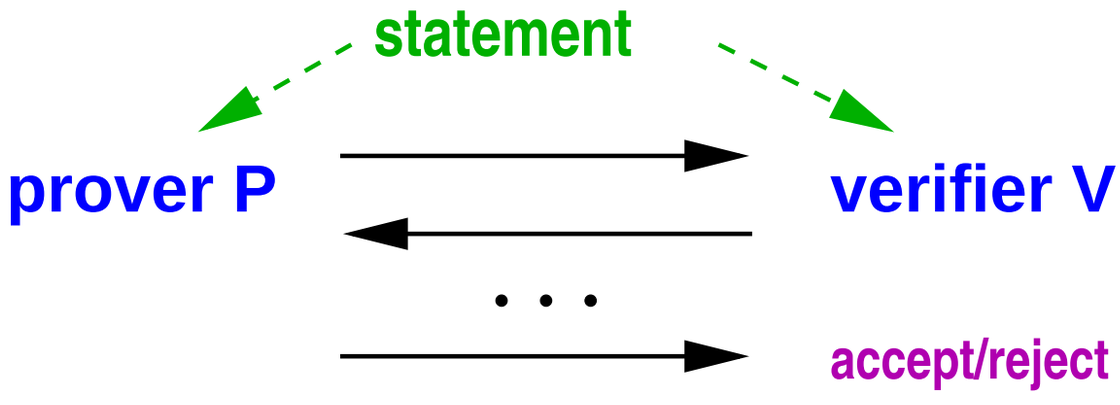

0 R
1 o
2 K
3 e
4 Q
5 Z
6 B
7 d
8 j
9 E
10 V
11 n
12 F
13 q
14 t
15 L
16 M
17 a
18 u
19 P
20 U
21 J
22 X
23 m
24 Y
25 p
26 b
27 A
28 r
29 k
30 v
31 C
32 D
33 I
34 h
35 s
36 T
37 G
38 O
39 l
40 f
41 S
42 N
43 g
44 i
45 W
46 c
47 H

Baby-step giant-step DL algorithm (1)

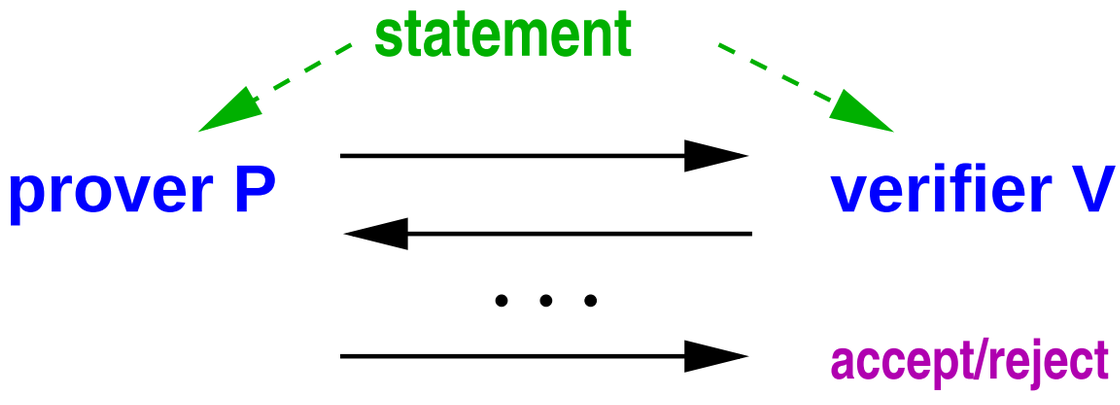



0 R
1 o
2 K
3 e
4 Q
5 Z
6 B
7 d
8 j
9 E
10 V
11 n
12 F
13 q
14 t
15 L
16 M
17 a
18 u
19 P
20 U
21 J
22 X
23 m
24 Y
25 p
26 b
27 A
28 r
29 k
30 v
31 C
32 D
33 I
34 h
35 s
36 T
37 G
38 O
39 l
40 f
41 S
42 N
43 g
44 i
45 W
46 c
47 H

Interactive proofs

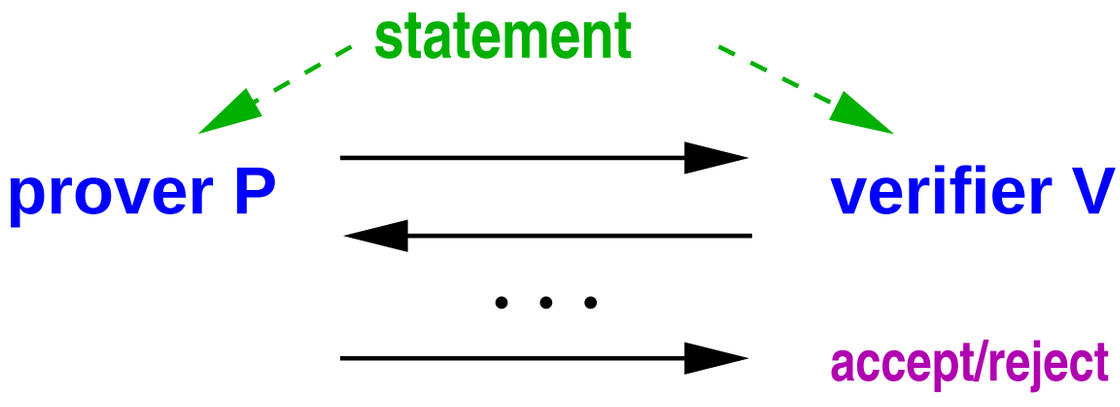


Interactive proofs



Motivations for interactive proofs:

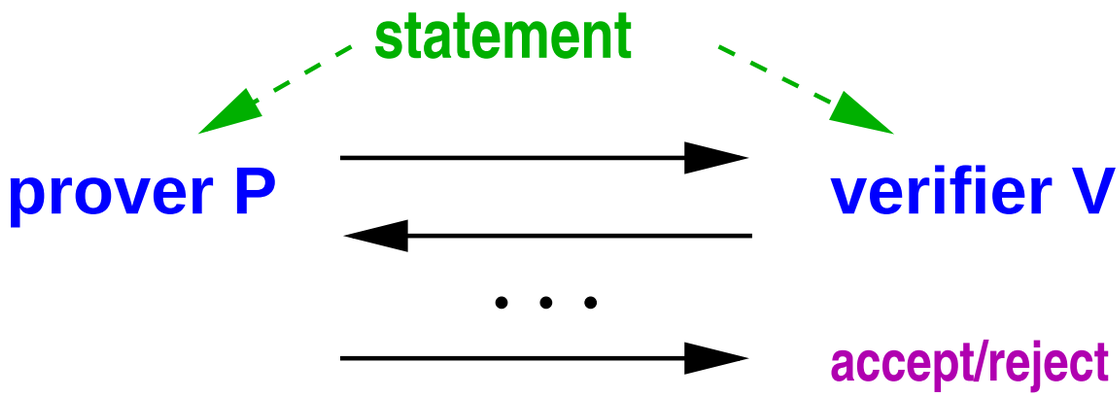
Interactive proofs



Motivations for interactive proofs:

- **Interactive proofs can be zero-knowledge.**

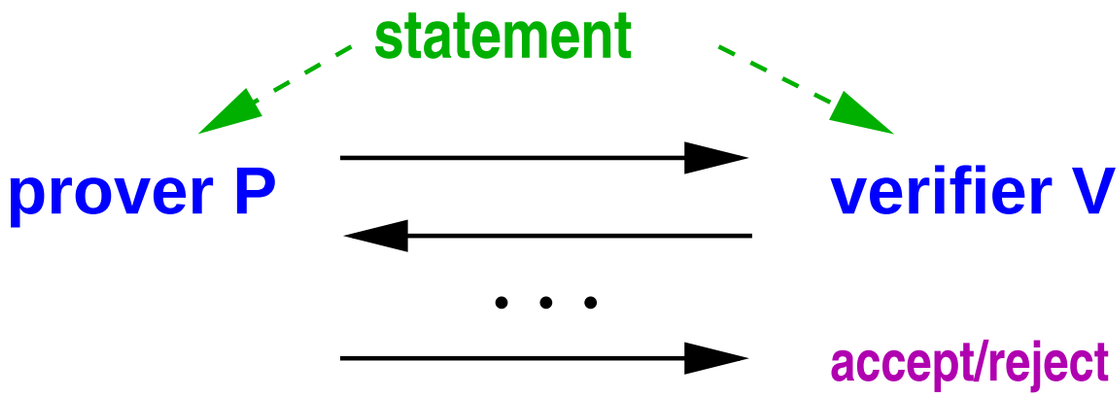
Interactive proofs



Motivations for interactive proofs:

- **Interactive proofs can be zero-knowledge.**
- **Interactive proofs are more powerful than static proofs**

Interactive proofs



Motivations for interactive proofs:

- **Interactive proofs can be zero-knowledge.**
- **Interactive proofs are more powerful than static proofs**
- **Applications:**
  - **Digital signature schemes**
  - **entity authentication**
  - **secure multi-party computation**

Proving knowledge of a DL (Schnorr)

Prover Peggy | **Verifier Vic**

knows $x \in \mathbb{Z}_q$ | $z = h^x$

$k \in_R \mathbb{Z}_q$

$t = h^k$

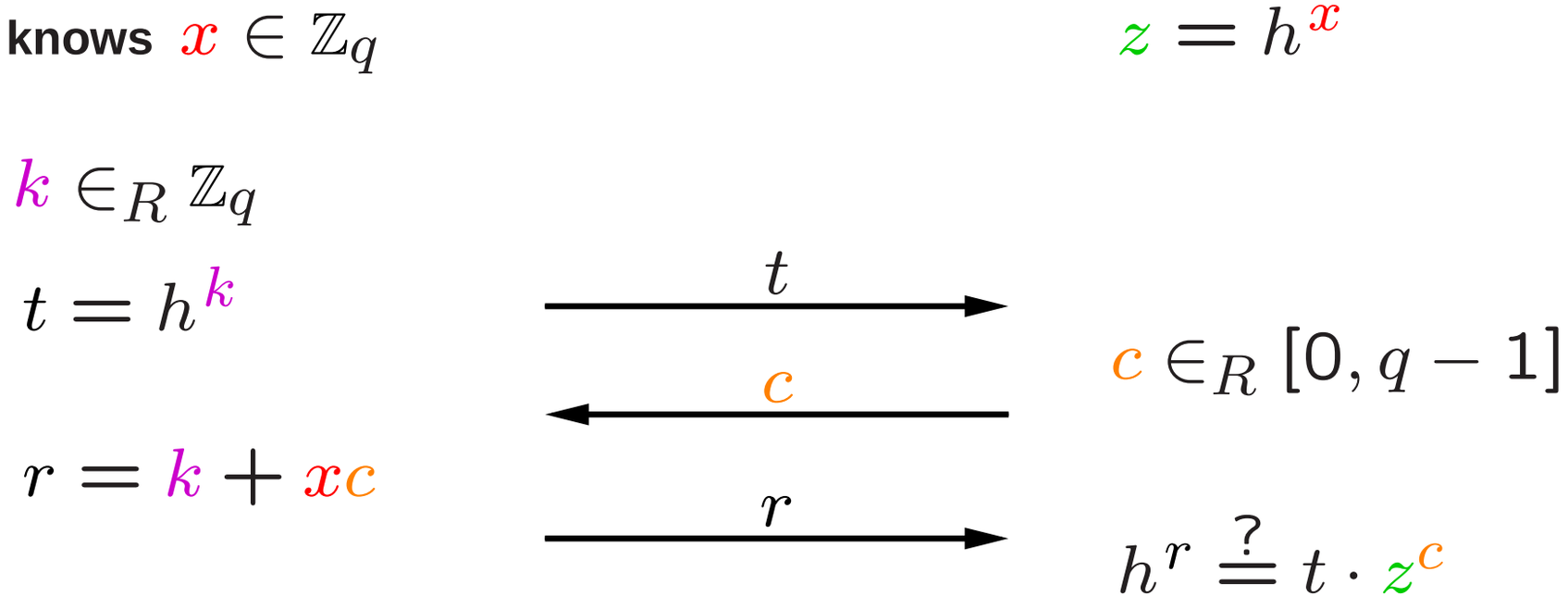
t →

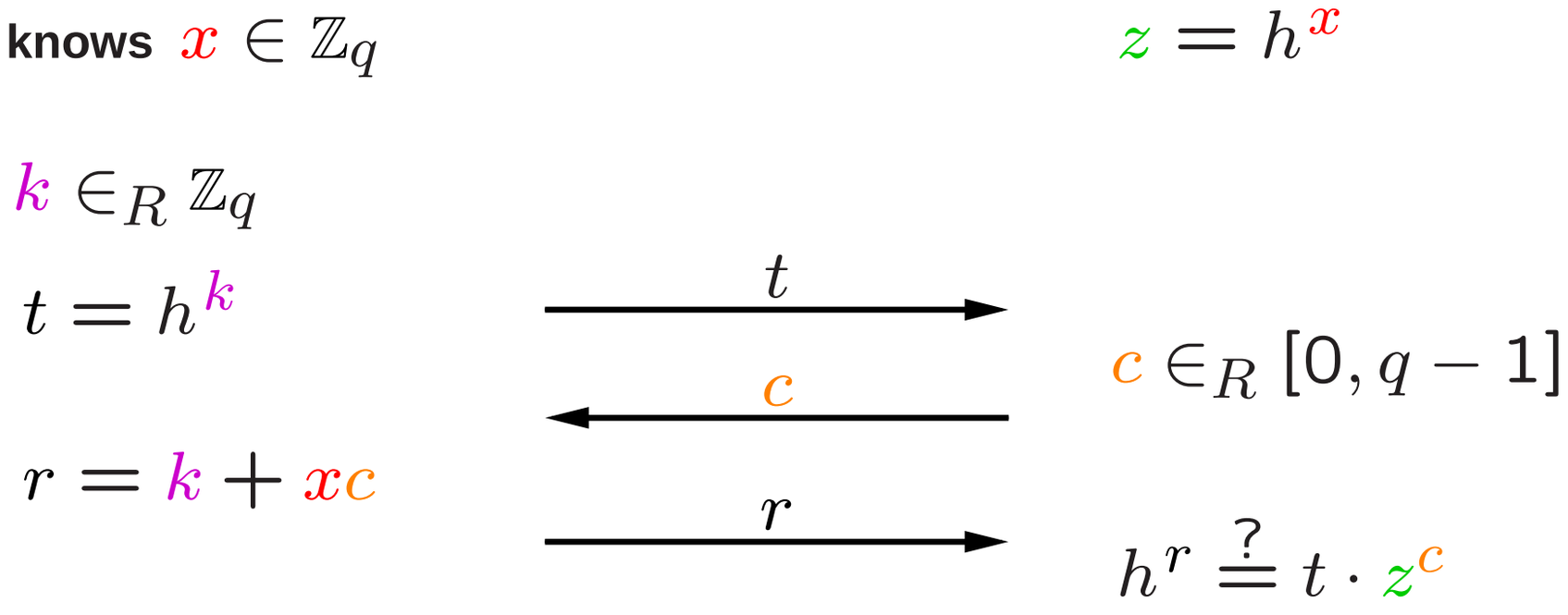
← c ; $c \in_R [0, q-1]$

$r = k + xc$

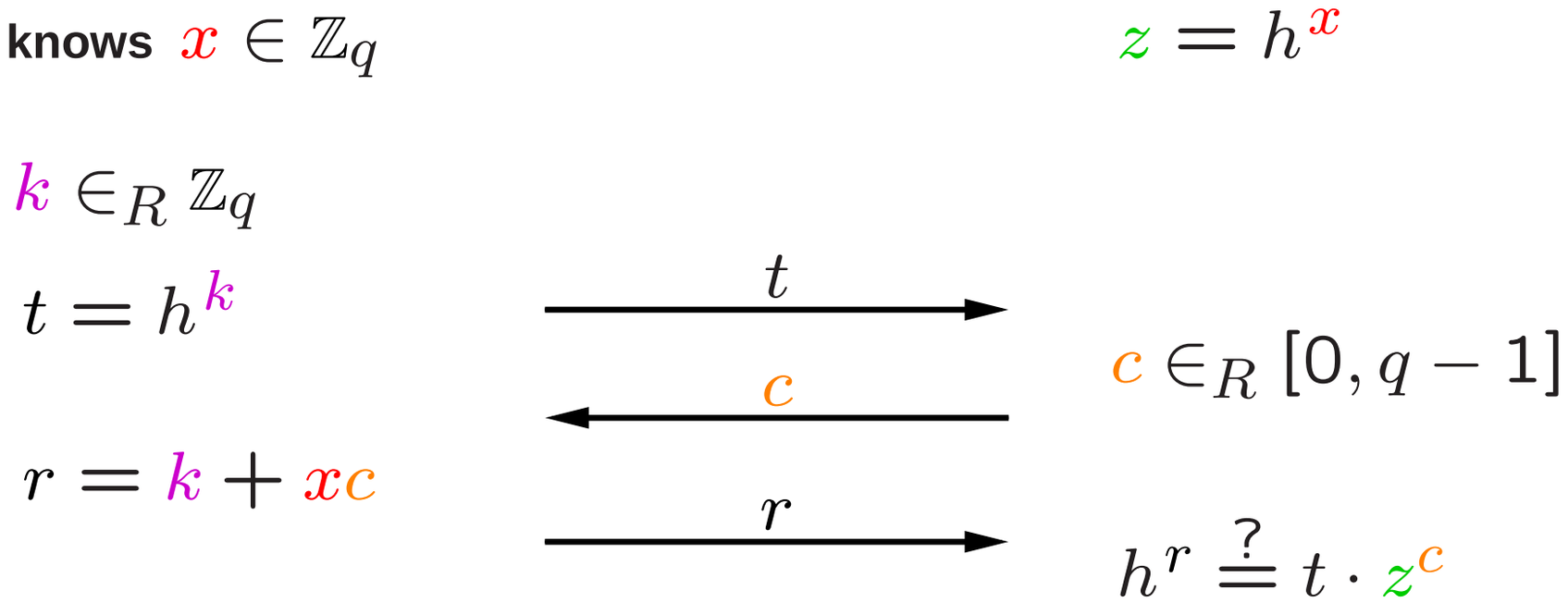
r →

$h^r \stackrel{?}{=} t \cdot z^c$


knows $x \in \mathbb{Z}_q$
$z = h^x$
$k \in_R \mathbb{Z}_q$
$t = h^k$
t
$c \in_R [0, q-1]$
c
$r = k + xc$
r
$h^r \stackrel{?}{=} t \cdot z^c$



Fact: P can answer any 2 challenges $\Rightarrow$ P knows x.



Fact: P can answer any 2 challenges $\Rightarrow$ P knows x.

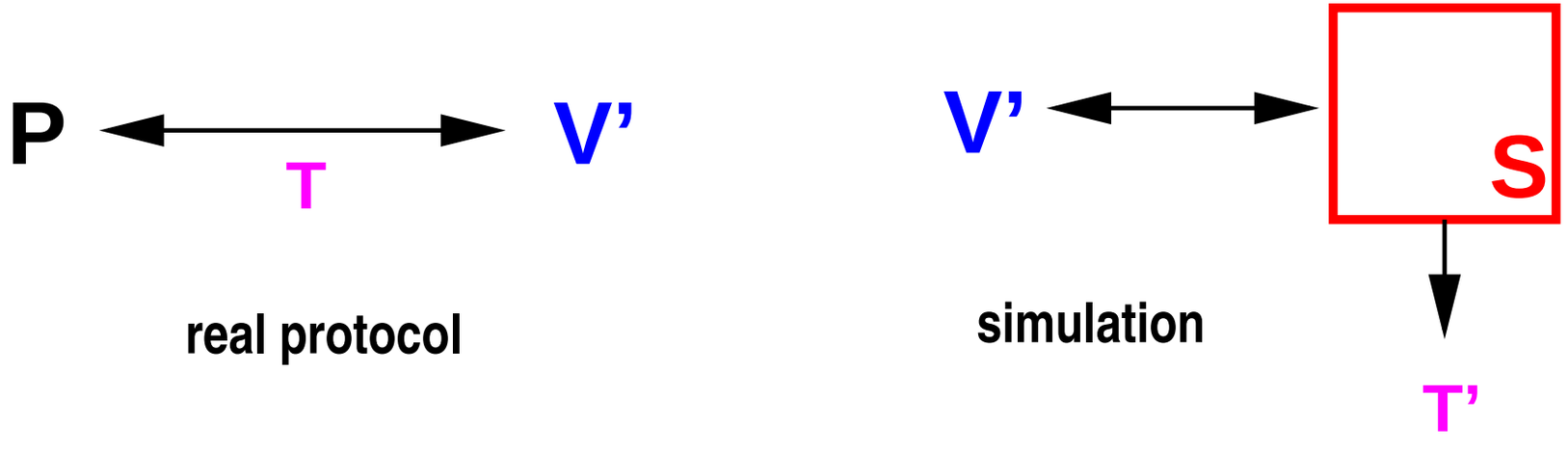
Proof:

$$h^r = t \cdot z^c, \quad h^{r'} = t \cdot z^{c'} \Rightarrow h^{r-r'} = z^{c-c'} = h^{x(c-c')}$$

$$\Rightarrow r - r' \equiv x(c - c') \pmod{q}$$

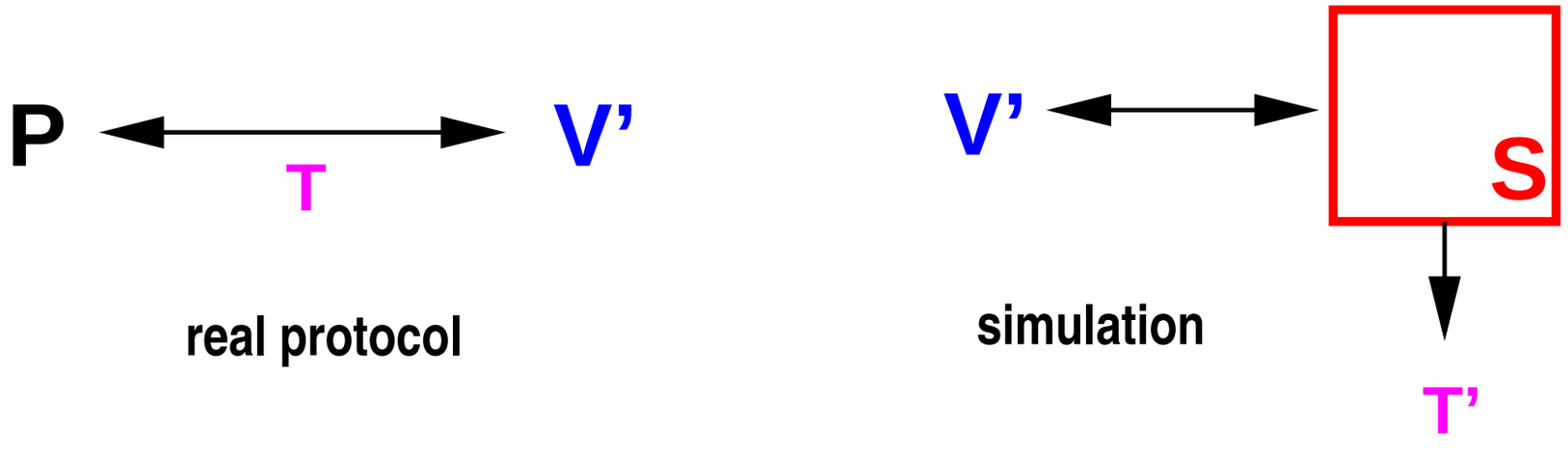
$$\Rightarrow x \equiv \frac{r - r'}{c - c'} \pmod{q}$$

Definition of zero-knowledge (ZK)



Definition: A protocol (P,V) is called (black-box) *zero-knowledge* if there exists an efficient simulator S with access to a (possibly) dishonest efficient verifier V' such that for every V' it outputs a simulated transcript T' which is indistinguishable from the real transcript T.

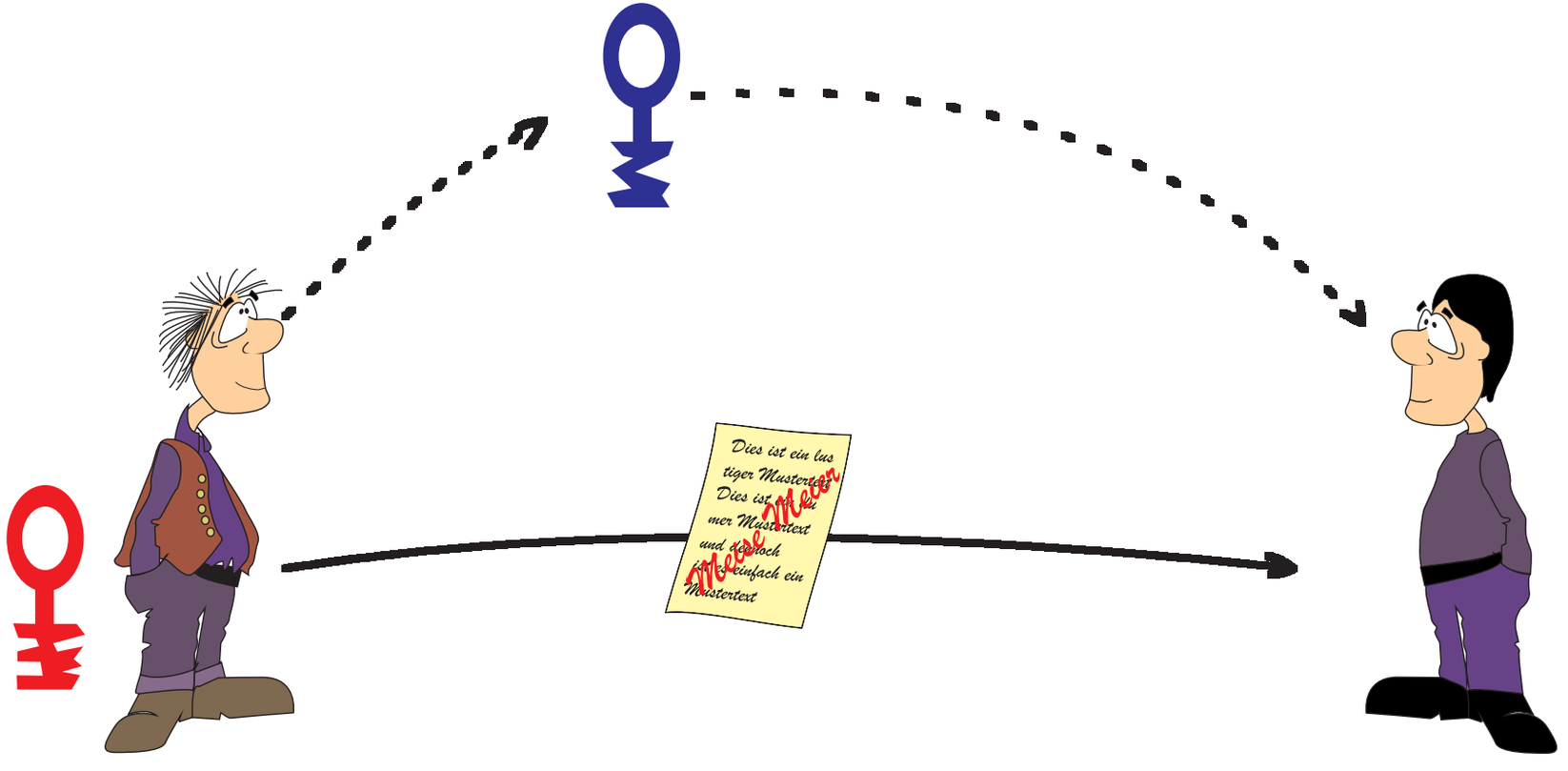
Definition of zero-knowledge (ZK)



Definition: A protocol (P,V) is called (black-box) *zero-knowledge* if there exists an efficient simulator S with access to a (possibly) dishonest efficient verifier V' such that for every V' it outputs a simulated transcript T' which is indistinguishable from the real transcript T.

Types of ZK: perfect, statistical, computational.

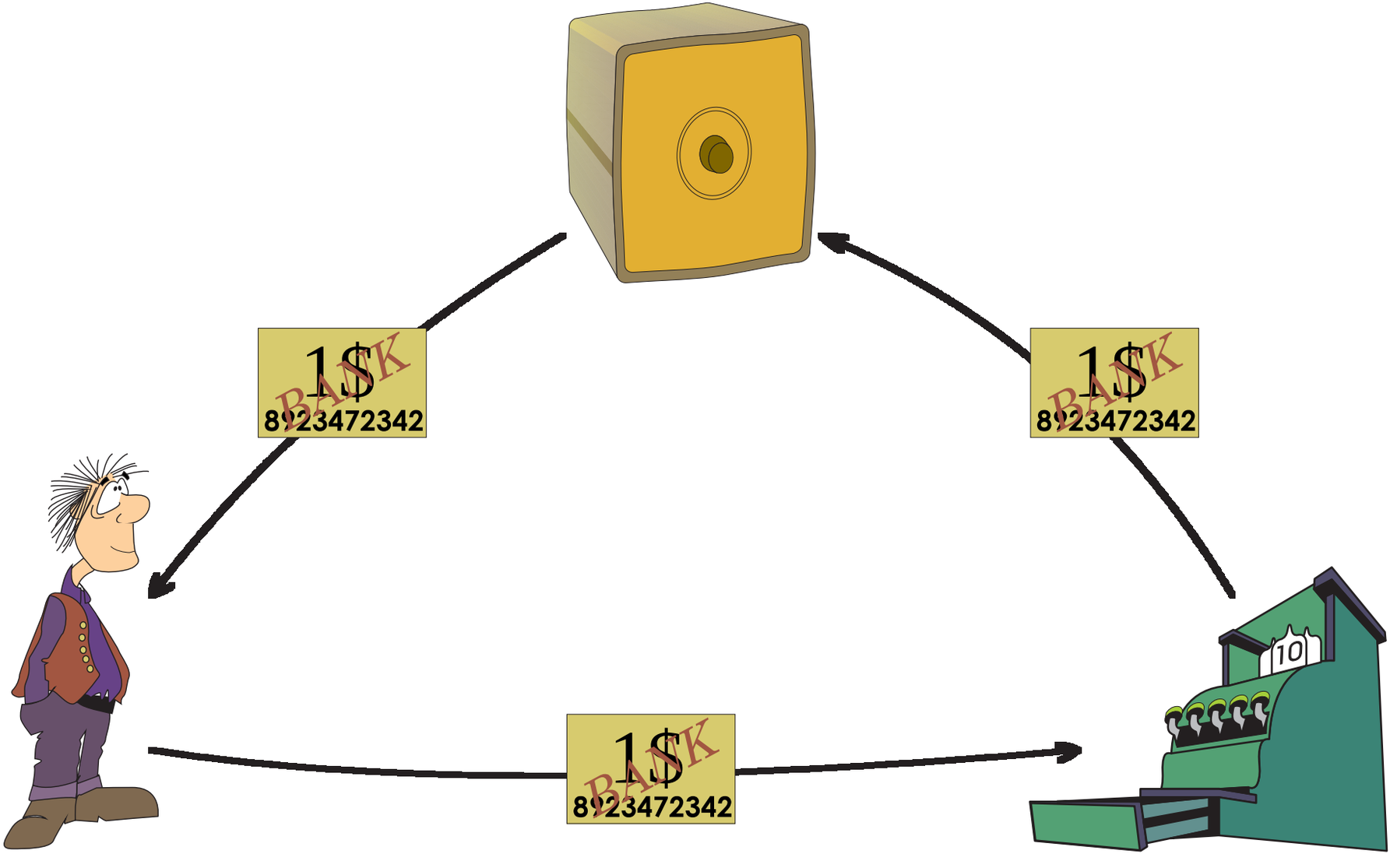
Digitale Signaturen



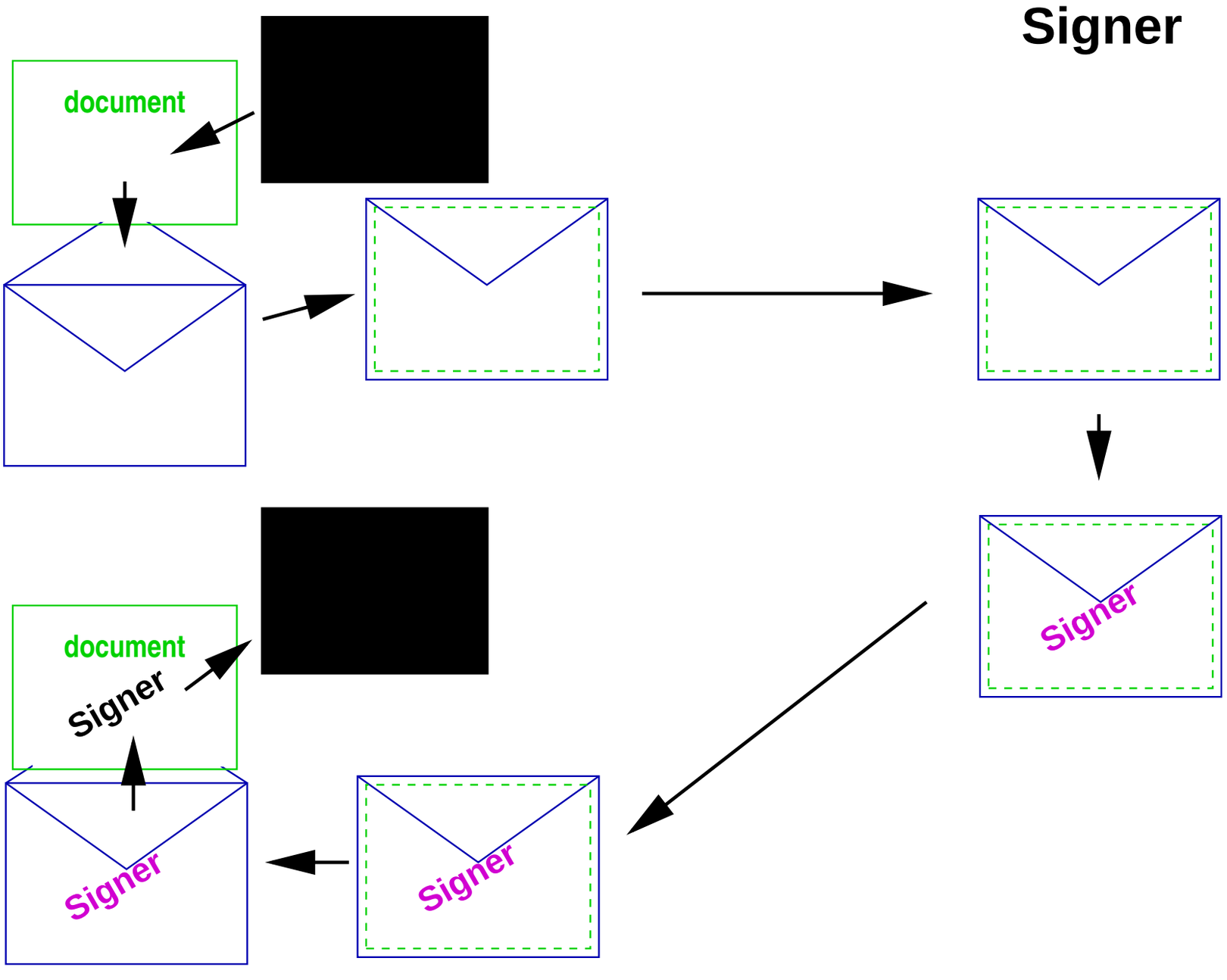
sign(,) =

verify(,) = { yes / no

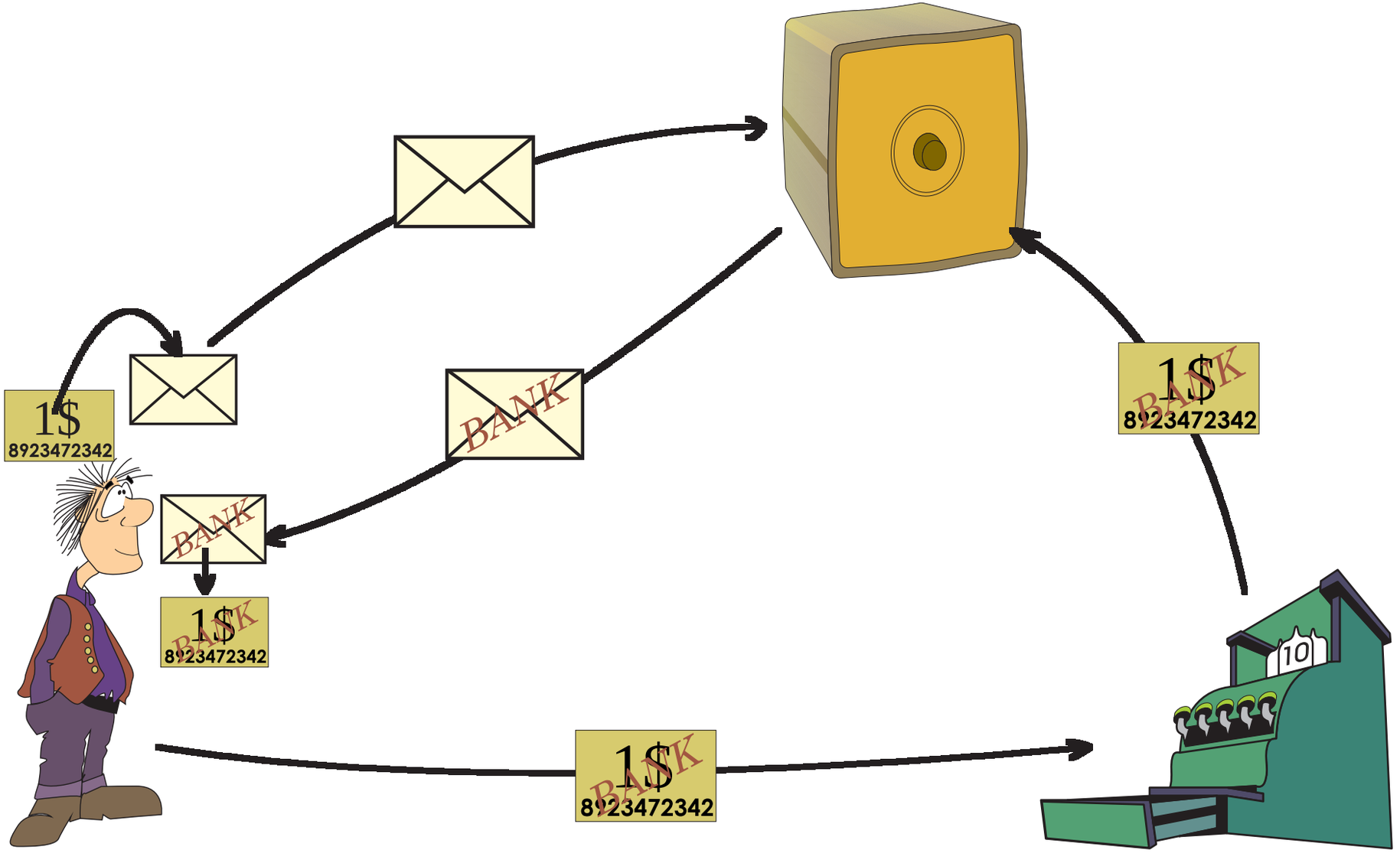
Digitales Geld



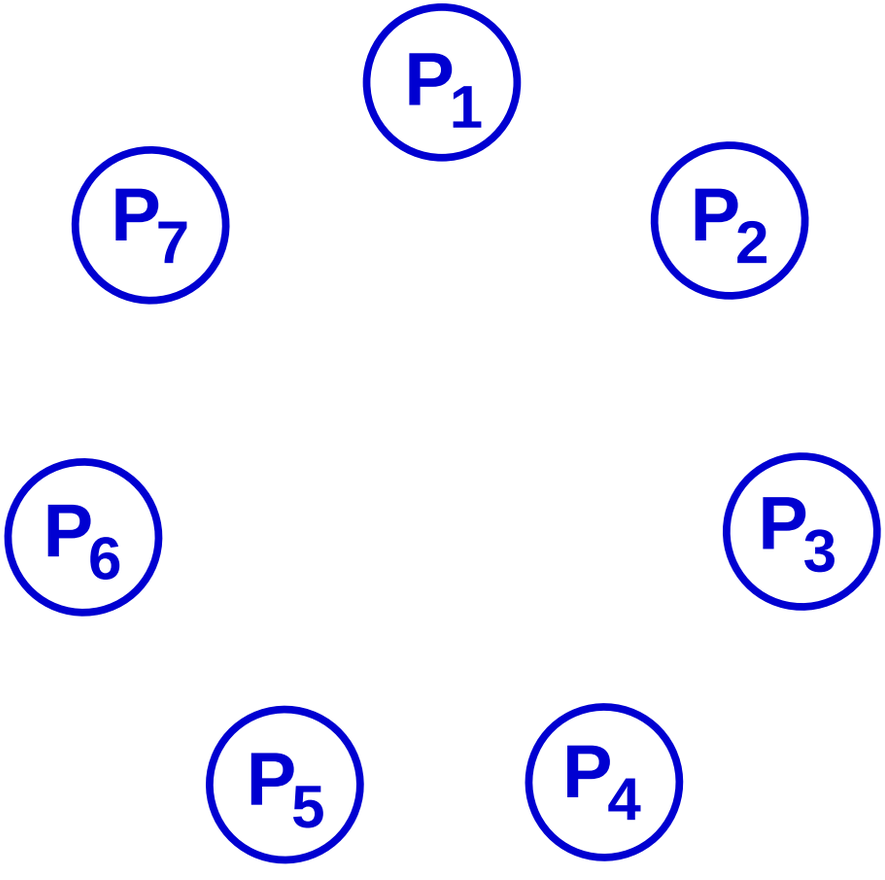
Blinde digitale Signaturen


Signer
document
Signer
document
Signer
Signer
Signer

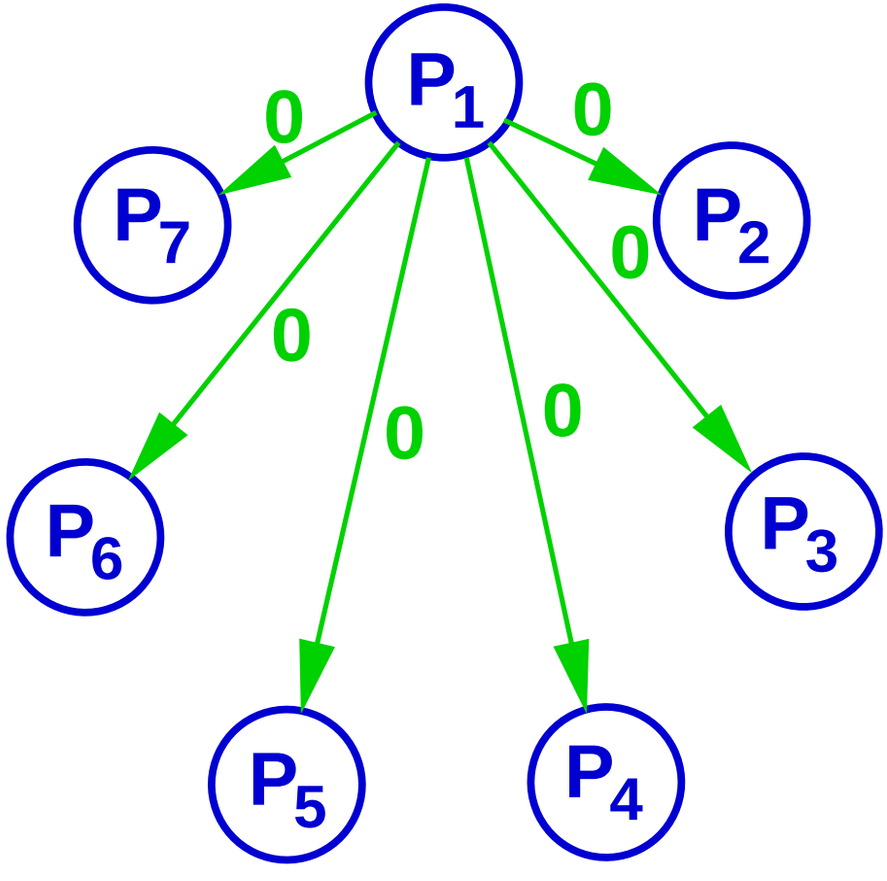
Anonymes digitales Geld (E-cash)



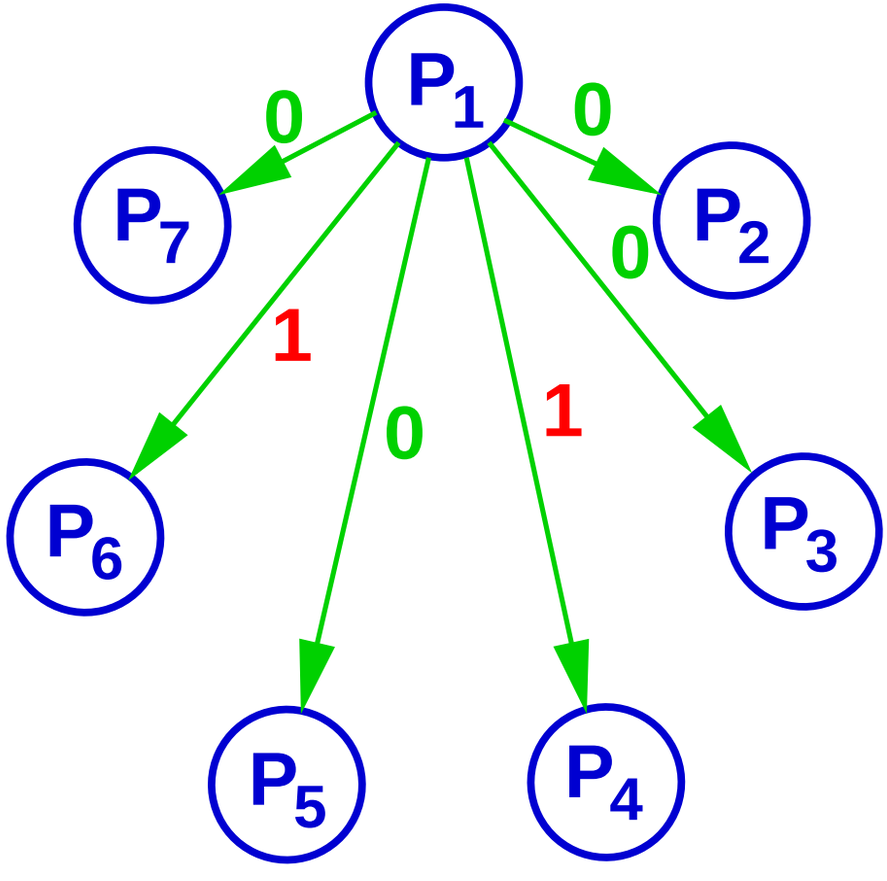
Broadcast / Byzantine agreement



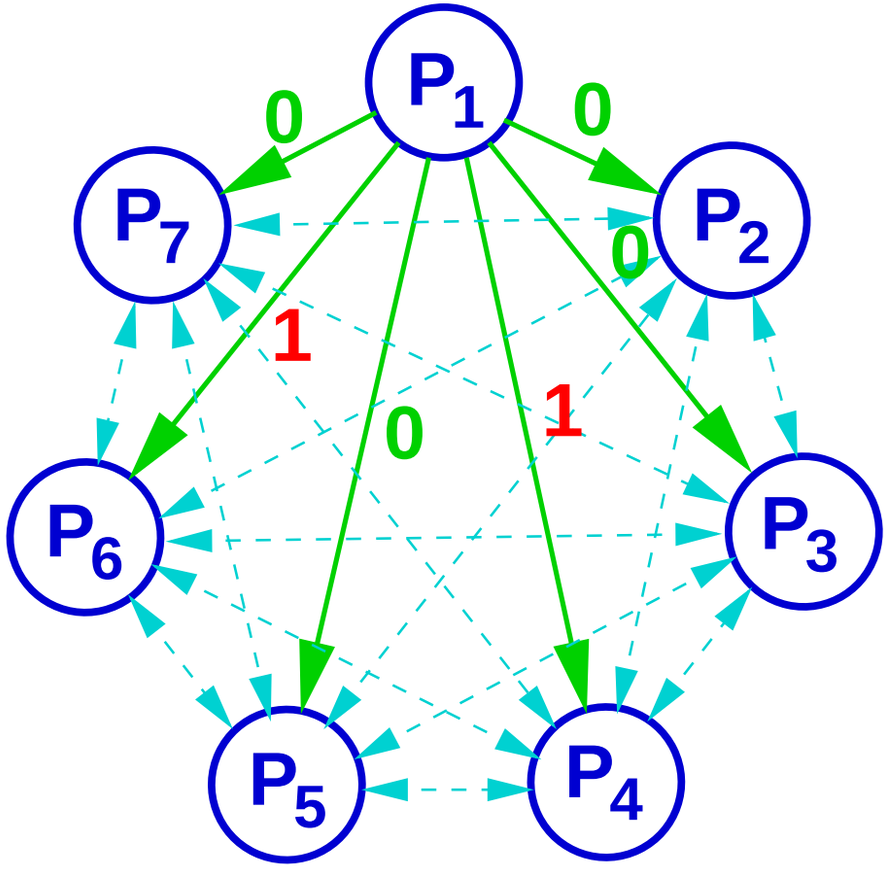
Broadcast / Byzantine agreement



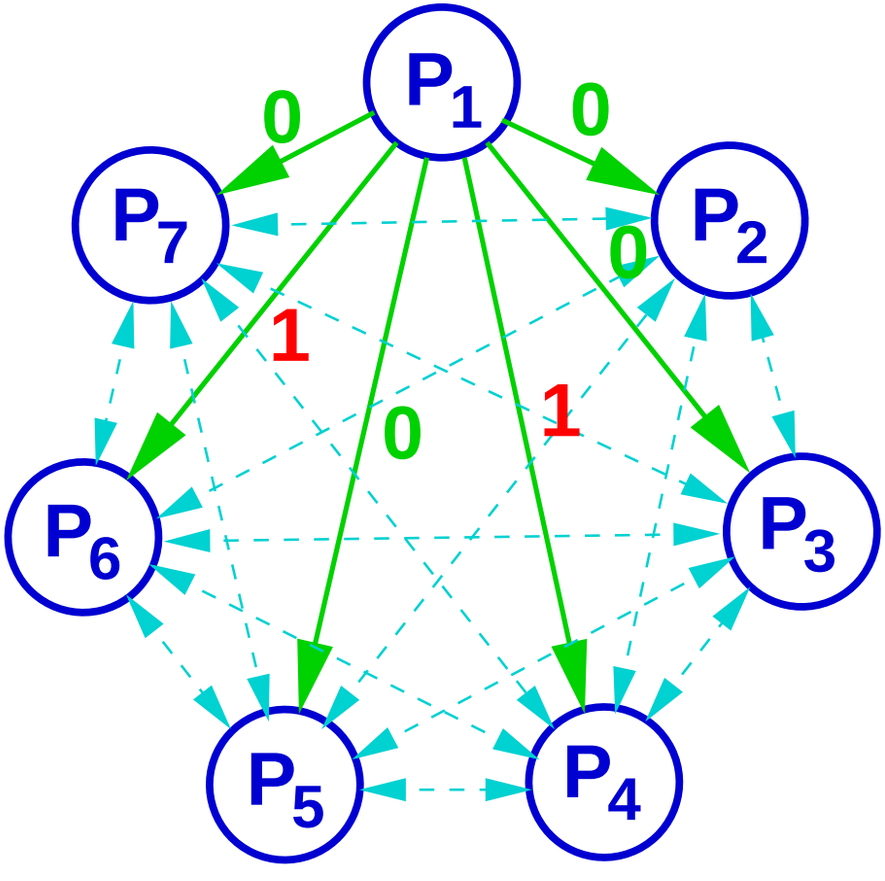
Broadcast / Byzantine agreement



Broadcast / Byzantine agreement

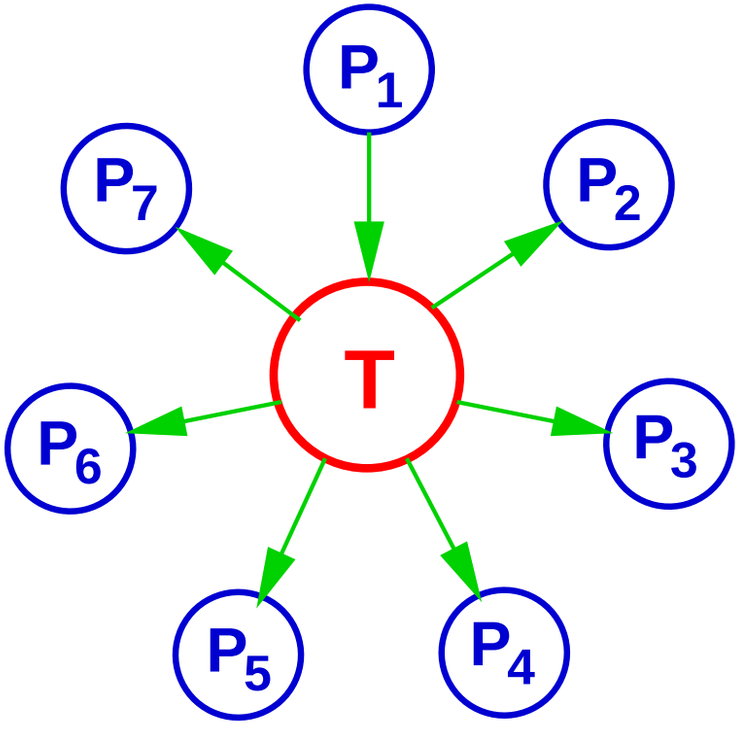


Broadcast / Byzantine agreement

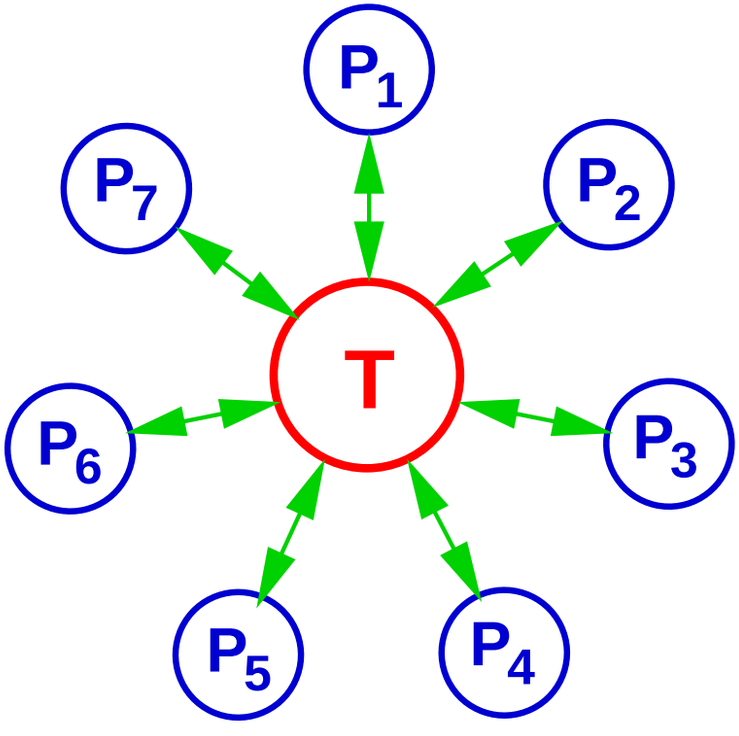


Theorem [LSP80]**:** **Among n players, broadcast is achievable if and only if t < n/3 players are corrupted.**

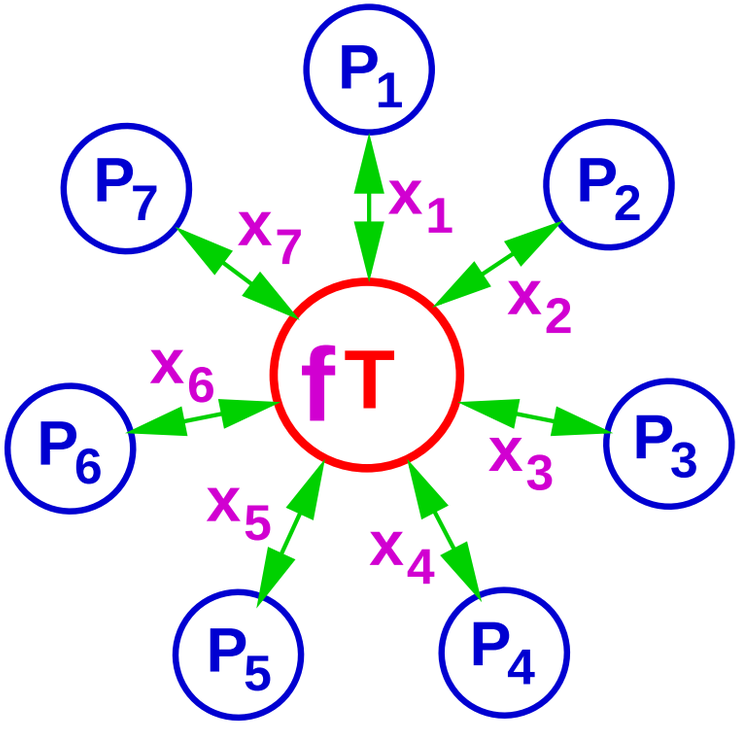
Generalization: Secure computation



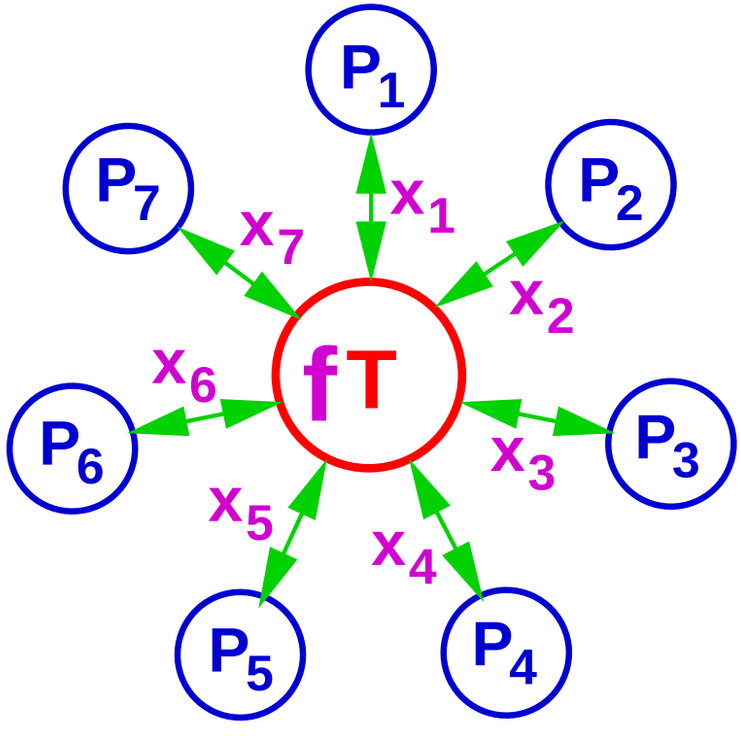
Generalization: Secure computation



Generalization: Secure computation

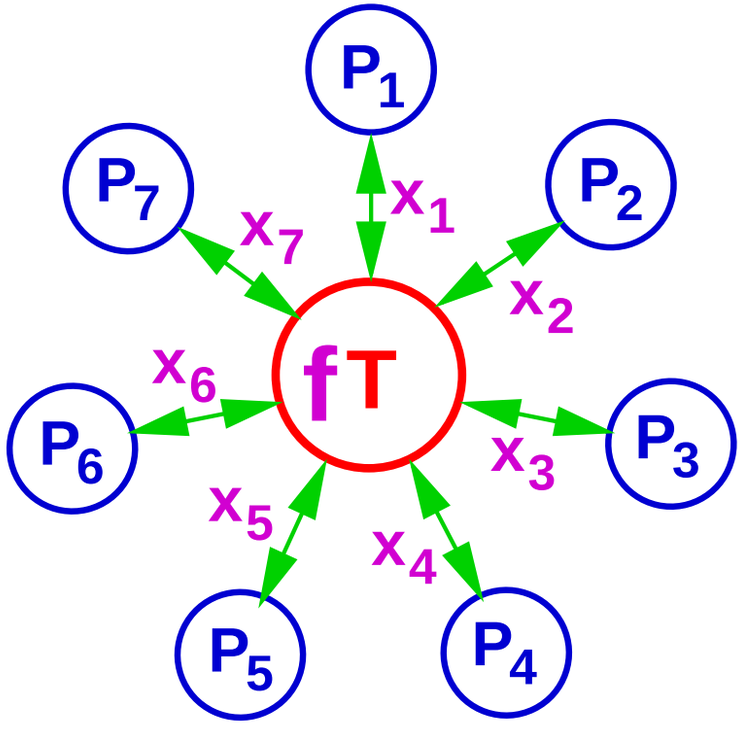


Generalization: Secure computation



T computes a function $f(x_1, \ldots, x_7)$ of the inputs.

Generalization: Secure computation



T computes a function $f(x_1, \ldots, x_7)$ of the inputs.

New operations of T:
- **receive secret input**
- **keep secret state**
- **perform operations on state**

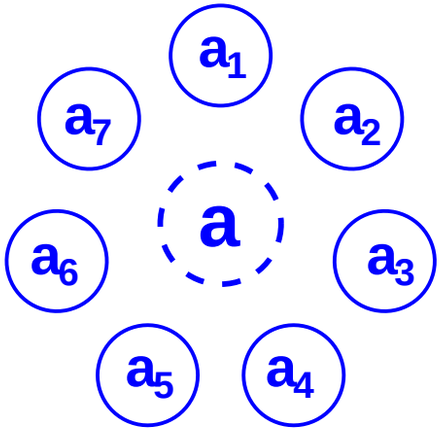
Some applications

- The millionaires' problem
- Preventing software piracy
- On-line auctions
- E-voting
- Secure aggregation of databases

Secret sharing

(t,n)-secret sharing: **Share a value a among n players such that**

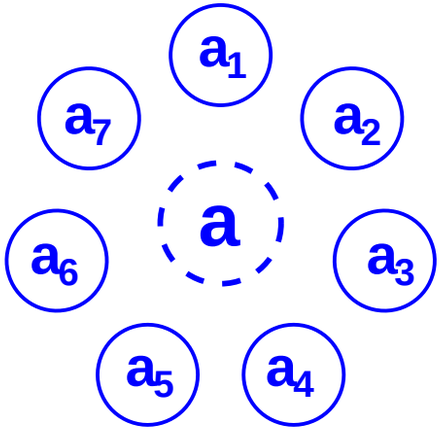
- **any t players have no information about a,**
- **any t+1 players can reconstruct a.**

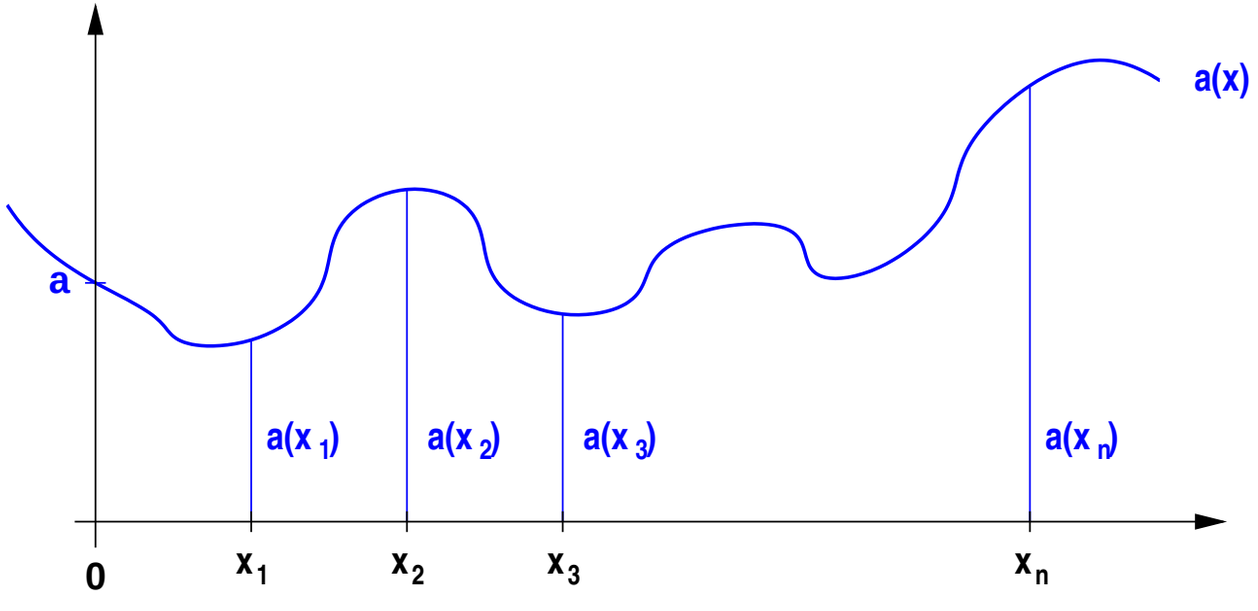


Secret sharing

(t,n)-secret sharing: Share a value a among n players such that

- **any t players have no information about a,**
- **any t+1 players can reconstruct a.**

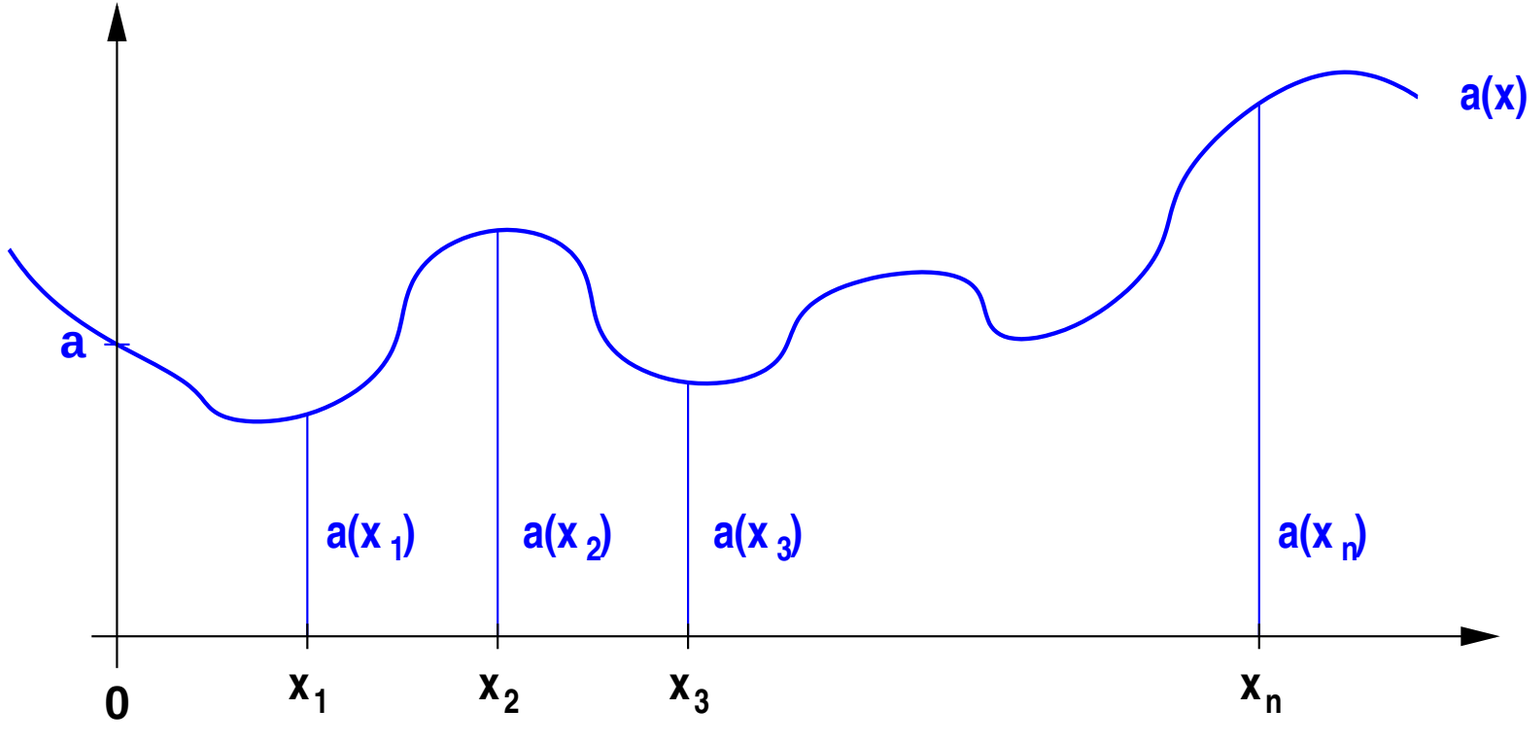




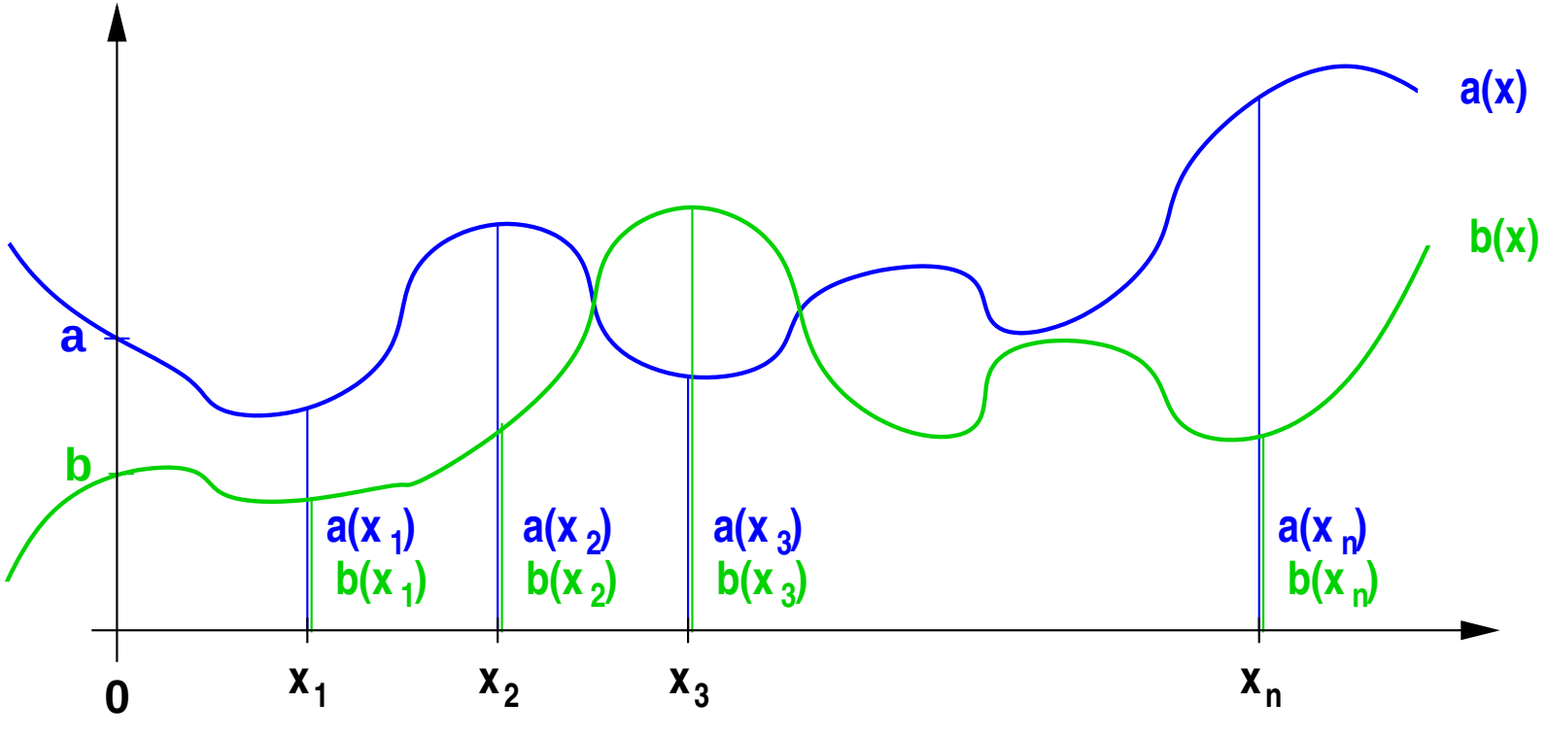
Every player is assigned a fixed value x_i from a finite field.

To share the value a, choose a random polynomial a(x) of degree t such that a(0)=a. The share of the i-th player is $a(x_i)$.

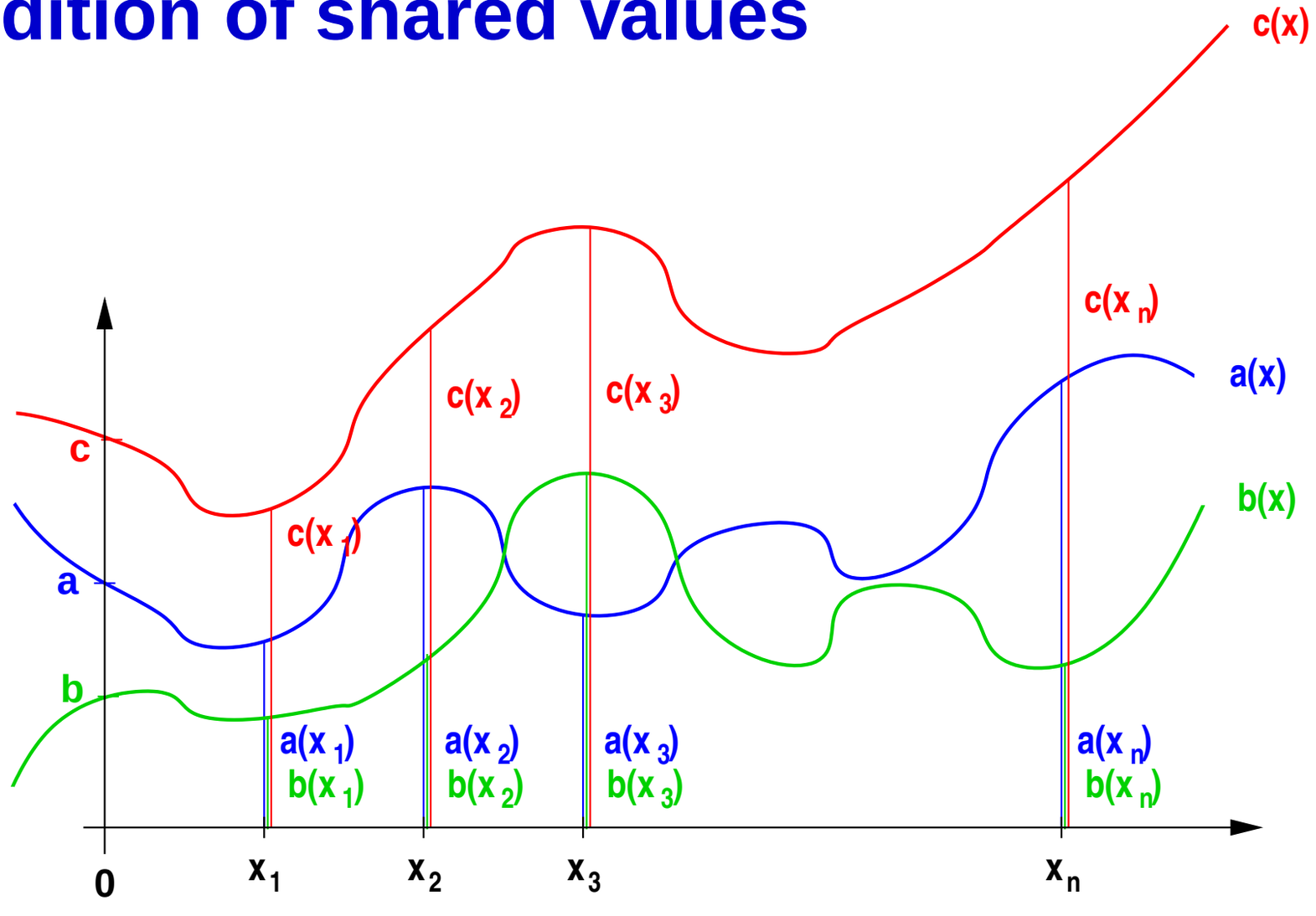
Addition of shared values



Addition of shared values



Addition of shared values



$c(x_i) = a(x_i) + b(x_i) \Rightarrow c(x) = a(x) + b(x) \Rightarrow c(0) = a(0) + b(0)$

Complexity of DL and factoring algorithms