DC-Networks – The Protocol

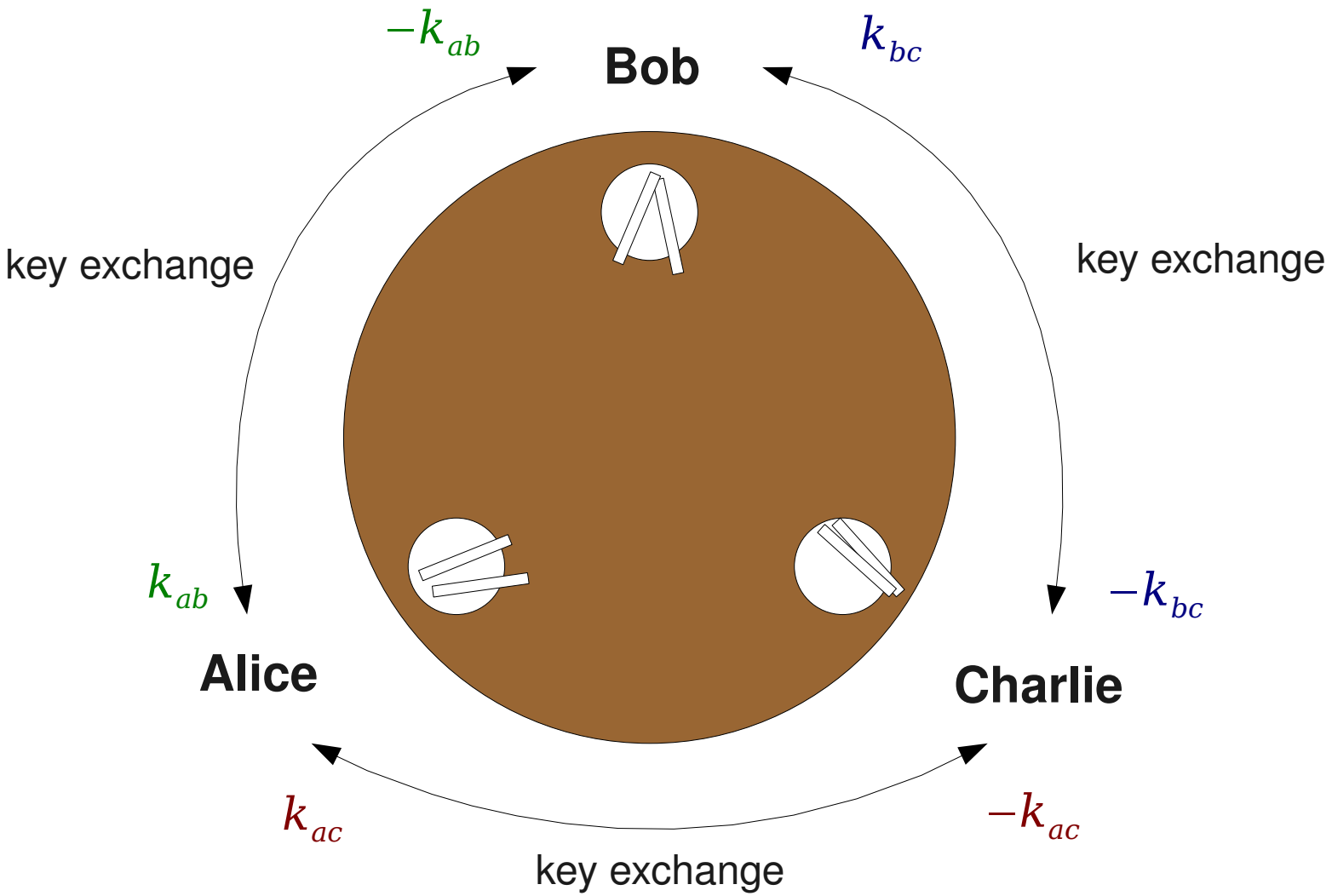
Immanuel Scholz

toc

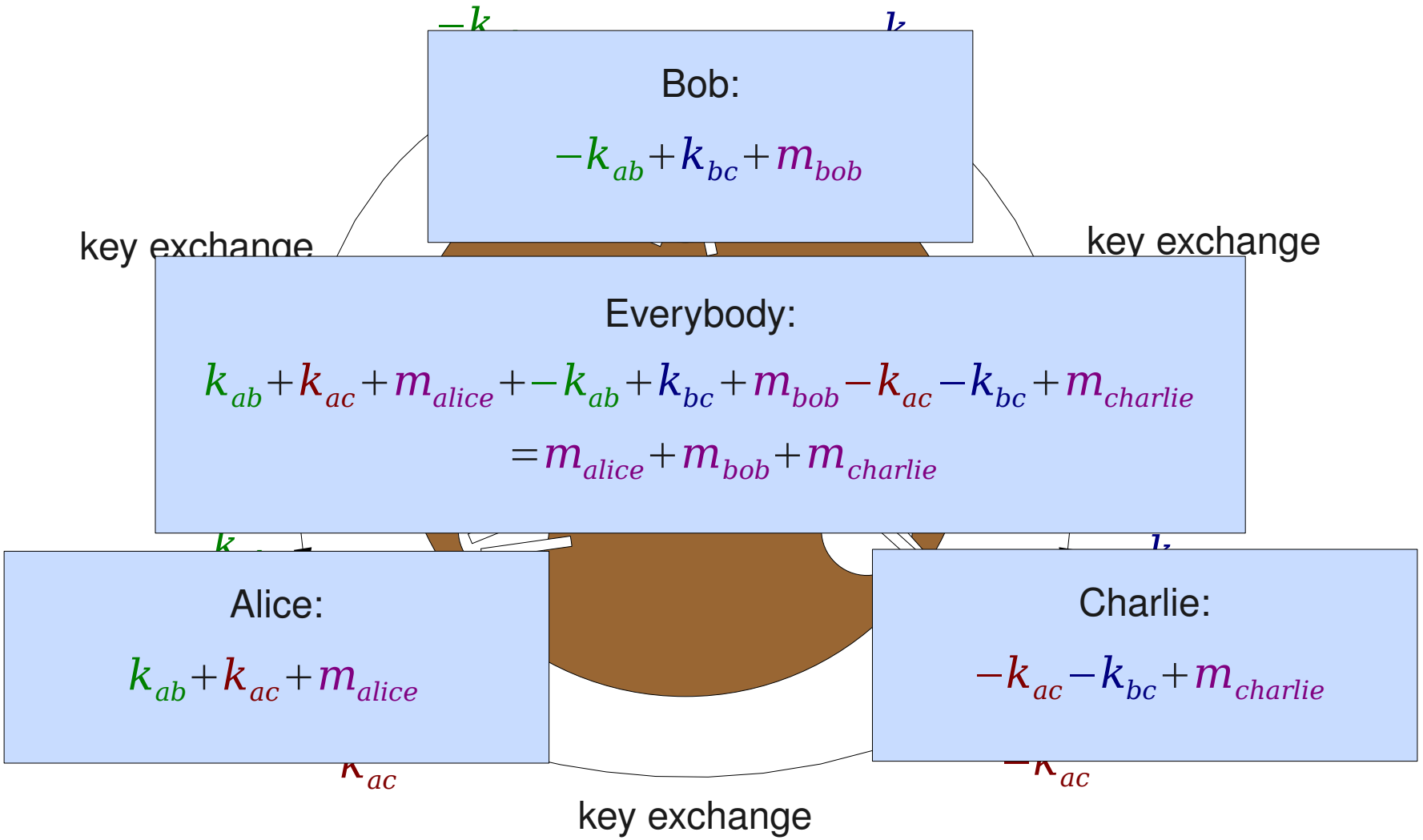
- Introduction
- Time
- Excluding bad clients
- Key Exchange
- On-demand disclosure
- Literature

Introduction

Is the meal paid by one of the cryptographers?



Is the meal paid by one of the cryptographers?



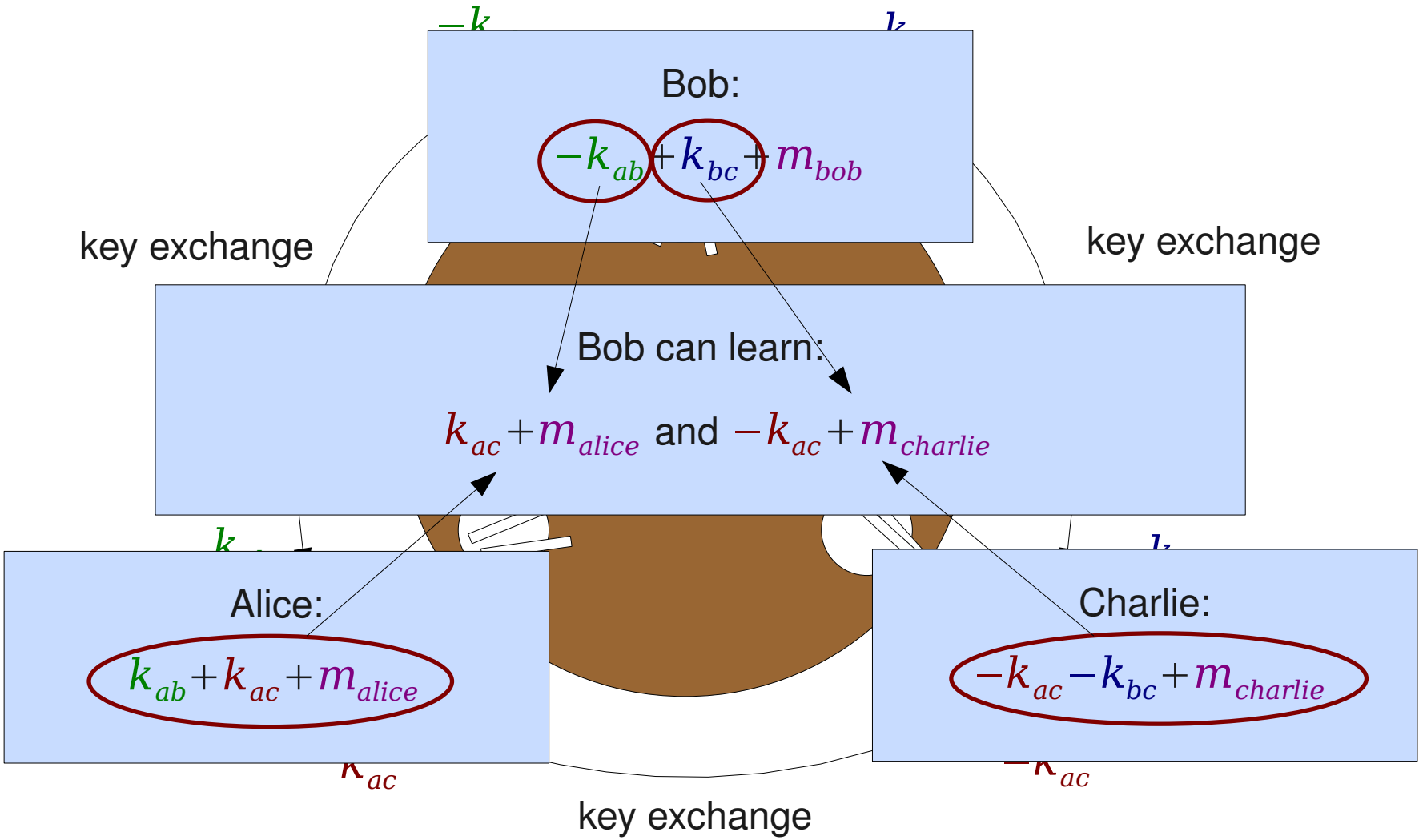
Remember the one-time-pad

$$k + m$$

If an attacker knows $k + m$ and if k is random, he does not learn anything about m

In other words: Key k “hides” the cleartext m

Bob does not learn anything



Sender and receiver anonymity

Everybody exchanges keys with everybody

- Sender Anonymity

Everybody gets every message

- Receiver Anonymity

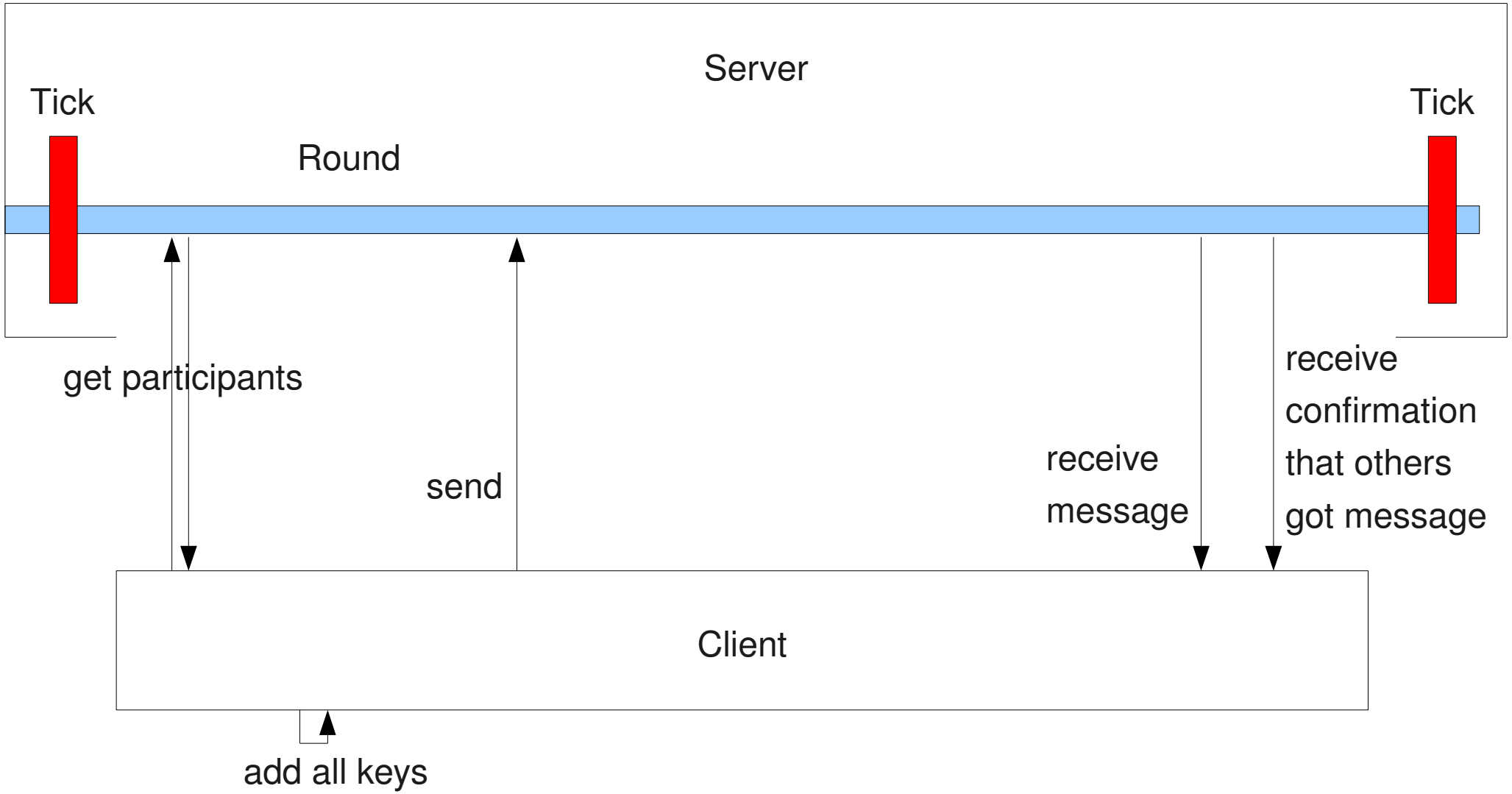
Time

Rounds, separated by ticks

In each round, each client needs to:

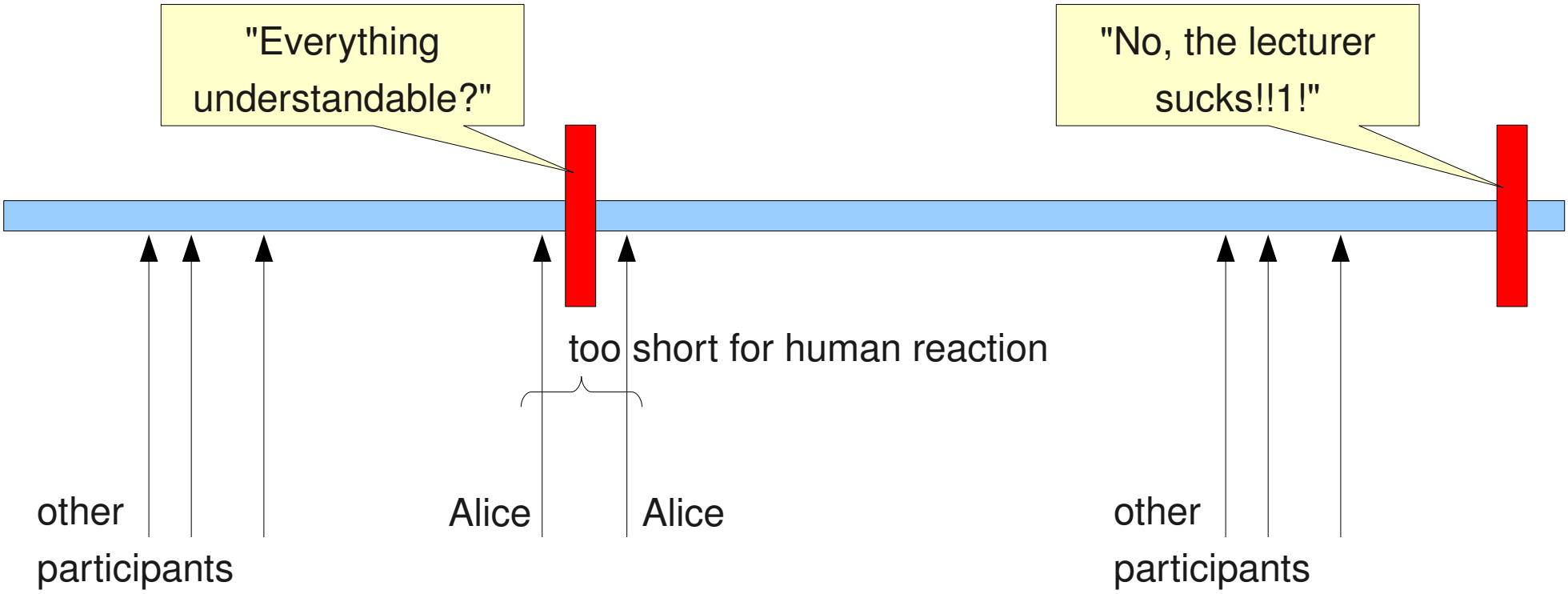
- Get the list of all participants (may have changed)
- Prepare and send the new message
- Get the message of the last round
- Know, that all others have received the message of the last round

The protocol



Timing attack

- Attacker knows who sends *at which time*
- A client answering too fast did not send the answer



Timing attack

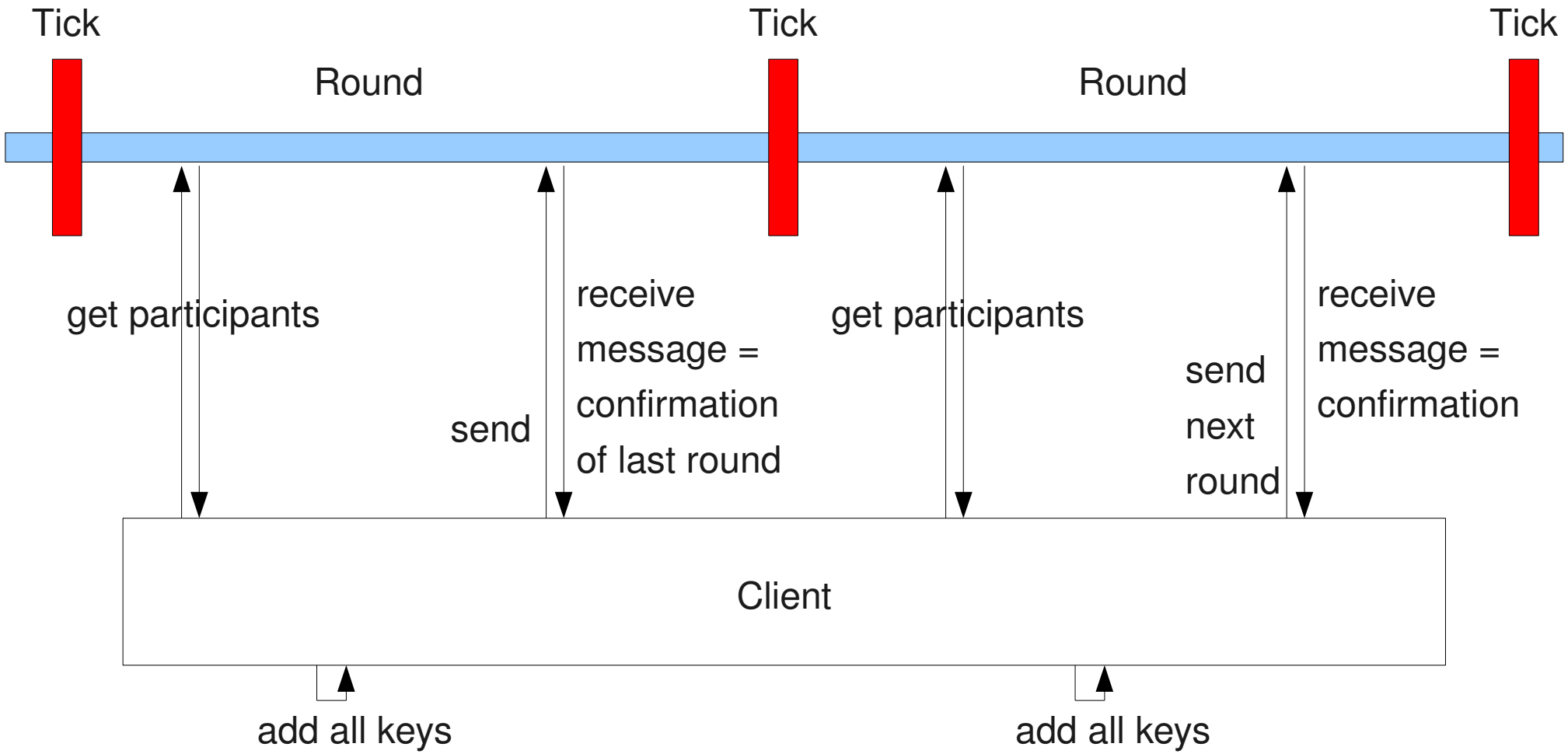
Solution: Server return message *after* client sends

- All clients receive messages delayed at least one round

Nice side effects:

- Only client → server communication (“firewall-proof”)
- Client receives implicit confirmation
  (or else last round message is broken)

The protocol



Excluding bad clients

Broken and malicious clients

Broken clients

- Server logs out any client that did not send
- Ban lists, reputation systems etc...

Malicious clients

- Anonymous reservation scheme and trap messages
- Google “DSuKrypt.pdf”, page 191-192, 202-203

Key Exchange

Remember: Diffie & Hellman

Alice	Server	Bob
	p, q	
choose x_{alice}		choose x_{bob}
$q^{x_{alice}} \bmod p$	publish → ←	$q^{x_{bob}} \bmod p$
$k_{ab} = (q^{x_{bob}})^{x_{alice}} = q^{x_{alice} x_{bob}}$		$k_{ab} = (q^{x_{alice}})^{x_{bob}} = q^{x_{alice} x_{bob}}$

Diffie & Hellman

- No direct client to client communication needed
- Complete key graph
- Easy distribution
- Exchange seeds for random number generator
  - Unfortunately, keys become “provable”
- But: Clients can always exchange real one-time-pads (they even do not have to tell the server)

On-demand disclosure

Original scheme by David Chaum

- Participants sign round keys
- Alice and Bob exchange their signatures
- In case of “disclosure”, all participants publish their keys
- Alice verifies Bob's key and vice versa

"Watchmen" scheme

- n predefined watchmen may work together to disclose sender
- Client splits round key into n parts
- Send one part to each watchman
- In case of “disclosure”, all watchmen post their parts to reconstruct the round key

How to split a key

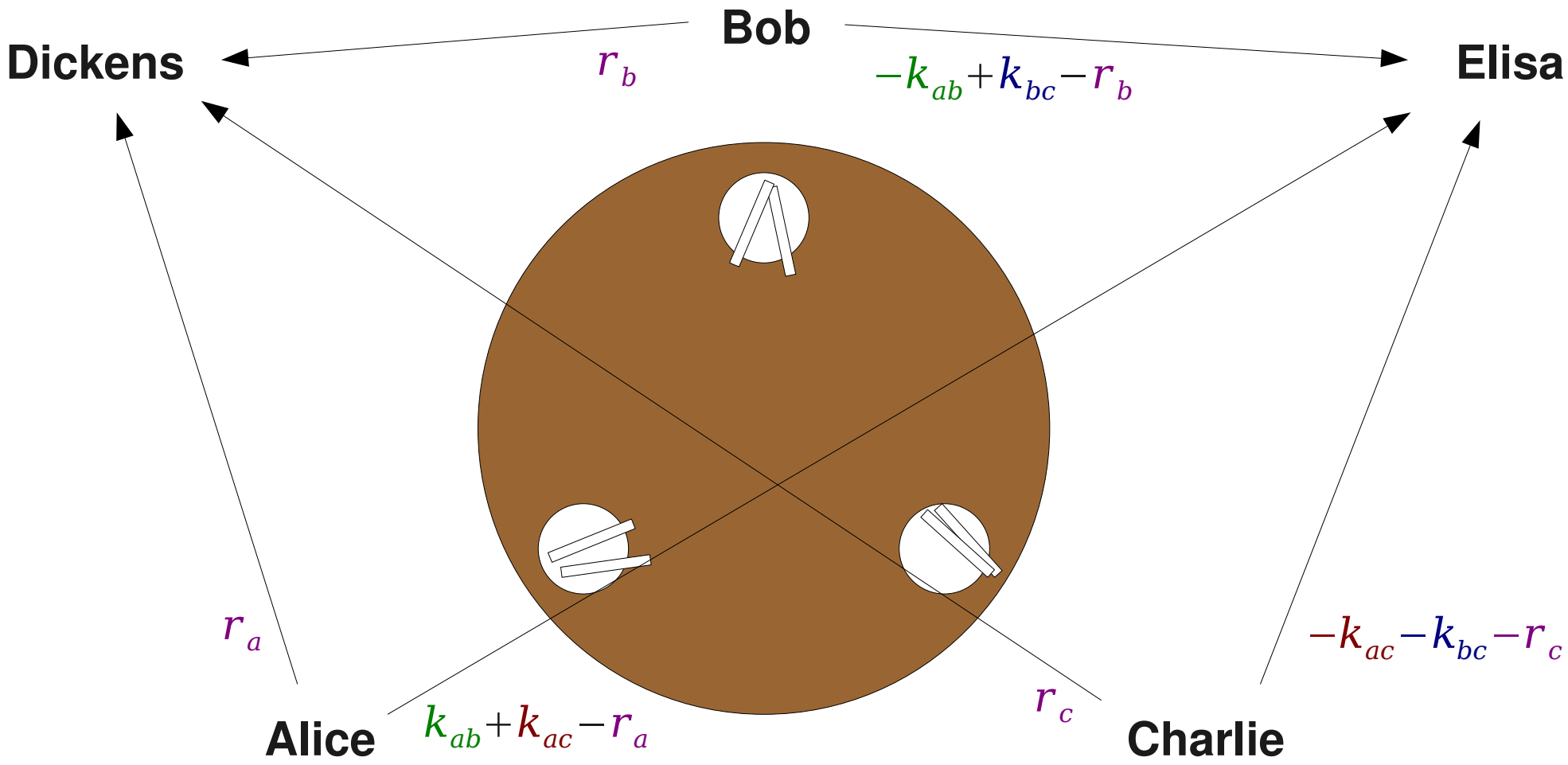
How to split a key k into n parts k_1 to k_n?

- Generate $n-1$ independent random numbers r_1 to r_{n-1}
- $k_i = r_i$ for all but k_n
- $k_n = k + \sum_{i=1}^{n-1} -r_i$

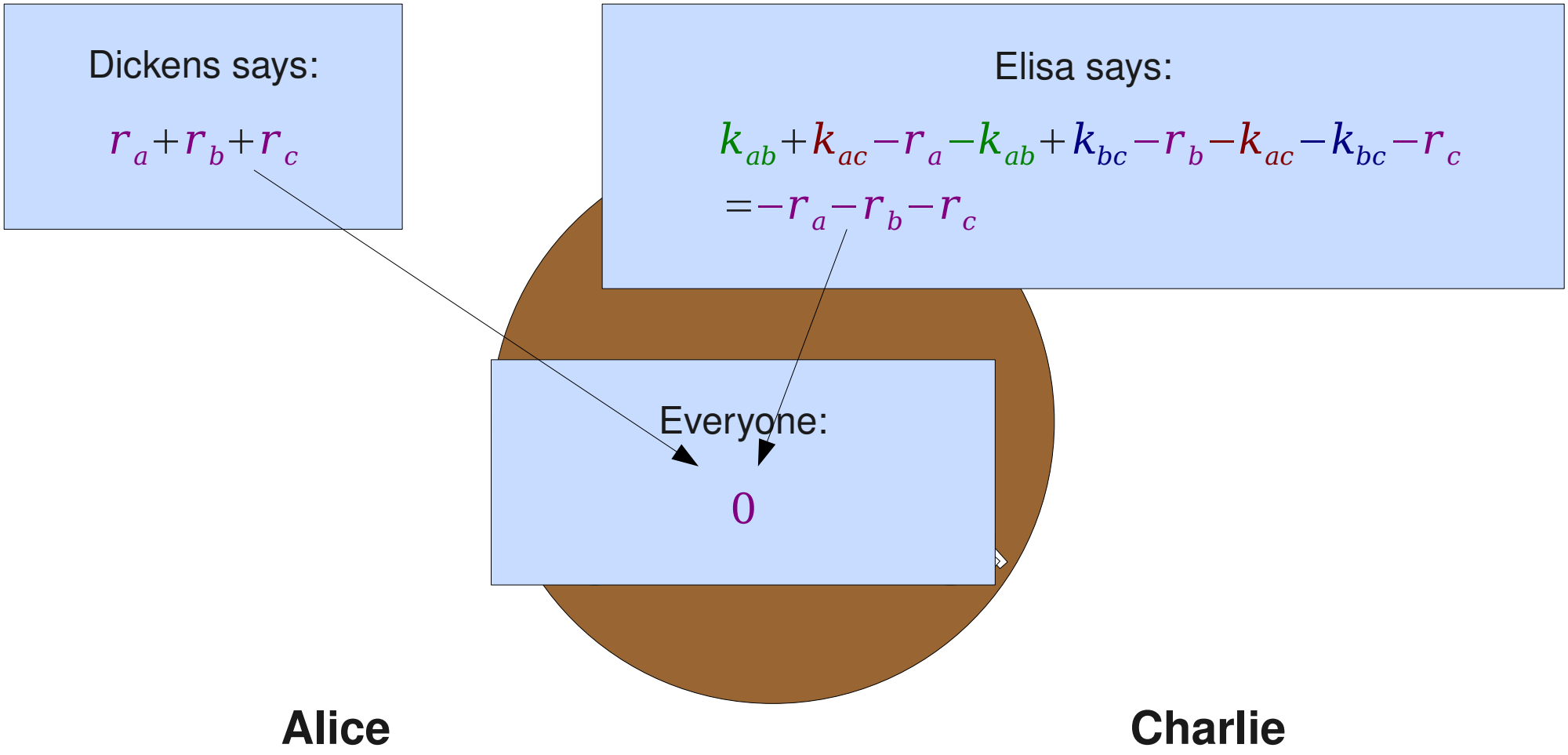
To join parts together, add them

$$\sum_{i=1}^{n} k_i = \sum_{i=1}^{n-1} k_i + k_n = \sum_{i=1}^{n-1} r_i + k + \sum_{i=1}^{n-1} -r_i = k$$

Split keys to Dickens and Elisa



Split keys to Dickens and Elisa



Why do the watchmen learn nothing?

- The first $n-1$ watchmen receive random numbers. Obviously, they learn nothing about k
- $k+\sum_{i=1}^{n-1} -r_i$ is a one-time-pad with $\sum_{i=1}^{n-1} -r_i$ as key
- The published values $r_a+r_b+r_c$ and $-r_a-r_b-r_c$ only consist of random numbers

Problem: Conspiration

- Alice and Bob exchange an additional key l_{ab}
- They add l_{ab} to their round message, but not to the key sent to Dickens and Elisa
- Sum of Dickens and Elisa is still 0
- On “disclosure”, both round keys from Alice and Bob mismatch (by l_{ab} resp. $-l_{ab}$)
- Of course, both could be considered “bad guys”

Threshold secret sharing scheme

- Reduce number of watchmen necessary for disclosure, e.g. “5 out of 10”
- Google “DSuKrypt.pdf”, page 131-133
- Using Adi Shamirs polynomial interpolation, some restrictions occur
  - All participants must use same set of watchmen (may be necessary anyway)
  - Each watchman must get the same x-coordinates from every participant

Comparison

Comparing the disclosure scheme by David Chaum:

- Both schemes are insecure against conspirations
- Using watchmen, a conspiration can not be formed after the message is published
- Using watchmen, no communication to the participants needed after message is published
  - The disclosure can be kept secret

Serverless

Why do we need a server at all?

- Participants publish their rounds on MySpace, Blog, Forum etc.
- Login by invitation: Someone send the new participants DH-key + Publishing URL within the DC-Network
- Auto-Logout by “not publishing once”

- Who is going to do the *data retention*? ;-)

Thanks

Interesting stuff:

- D.Chaum, “The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability”
  (Journal of Cryptology, vol. 1 no. 1, 1988, pp. 65-75 or http://www.cs.cornell.edu/People/egs/herbivore/dcnets.html)
- A.Pfitzmann, “Datensicherheit und Kryptographie”
  (http://dud.inf.tu-dresden.de/~pfitza/DSuKrypt.html, pp 176-199 and 123-125. German language)