

## II. PET for Protecting User Identities

### Protecting User Identities at Communication Level:

- Sender anonymity by dummy traffic, DC-nets, Crowds
- Unlinkability of sender and recipient by Mix-nets
- Recipient anonymity by message broadcast and implicit addresses
- 

Explicit address: place in the network

Implicit address: attribute that can be recognised by addressee

Invisible implicit address: only visible to its addressee (realisation by public-key or symmetric encryption)

Visible implicit address: visible to all users (realisation by self-chosen pseudonyms as message-prefixes)

Public address: known to every user

Private address: sender received it secretly from the addressee

## Combinations of addressing modes and address distribution [Pfitzmann et al. 1987]:

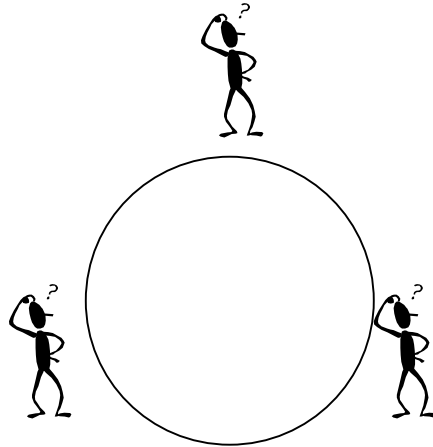
		address distribution		
		public address	private address	
addressing mode	implicit address	invisible	very costly, but necessary to establish contact	costly
		visible	linkable to holder	change after use
	explicit address		linkable to holder	linkable to holder

## 4. The DC-Net: Unconditional Sender and Recipient Untraceability

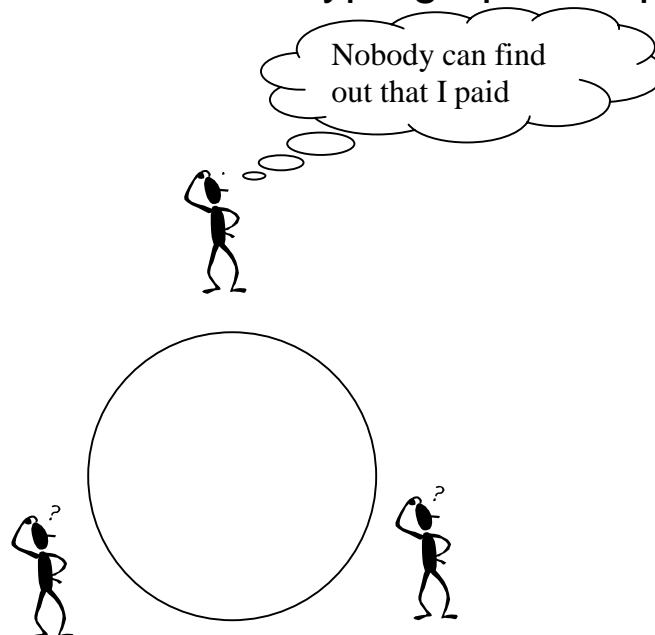
### 4.1 The Dining Cryptographers Problem [Chaum 1988]:

Who pays the dinner anonymously ?

Equal number of differences: NSA is paying

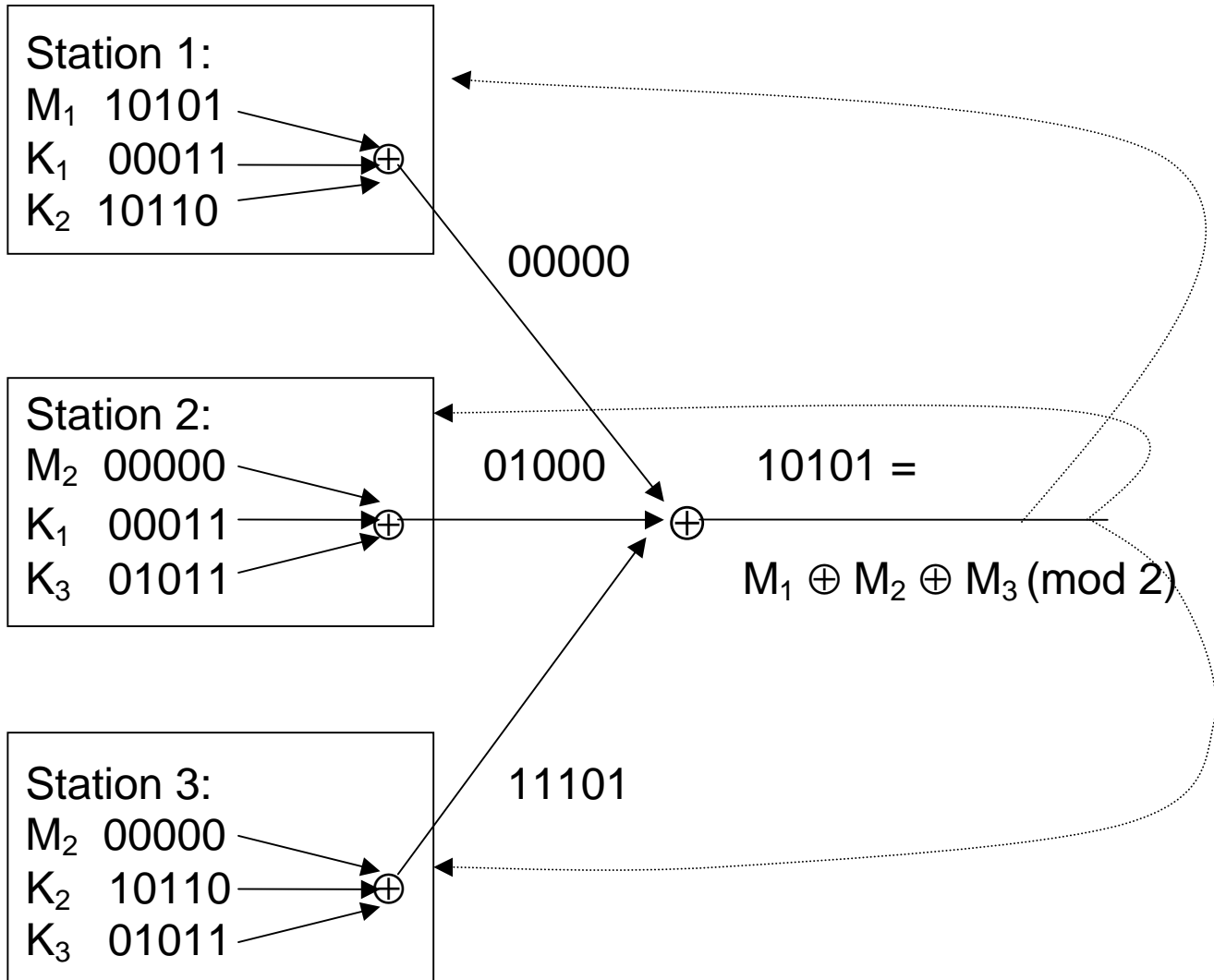


Odd number of differences: a cryptographer is paying



-> Nonpaying cryptographer learns nothing about which of the other two is paying

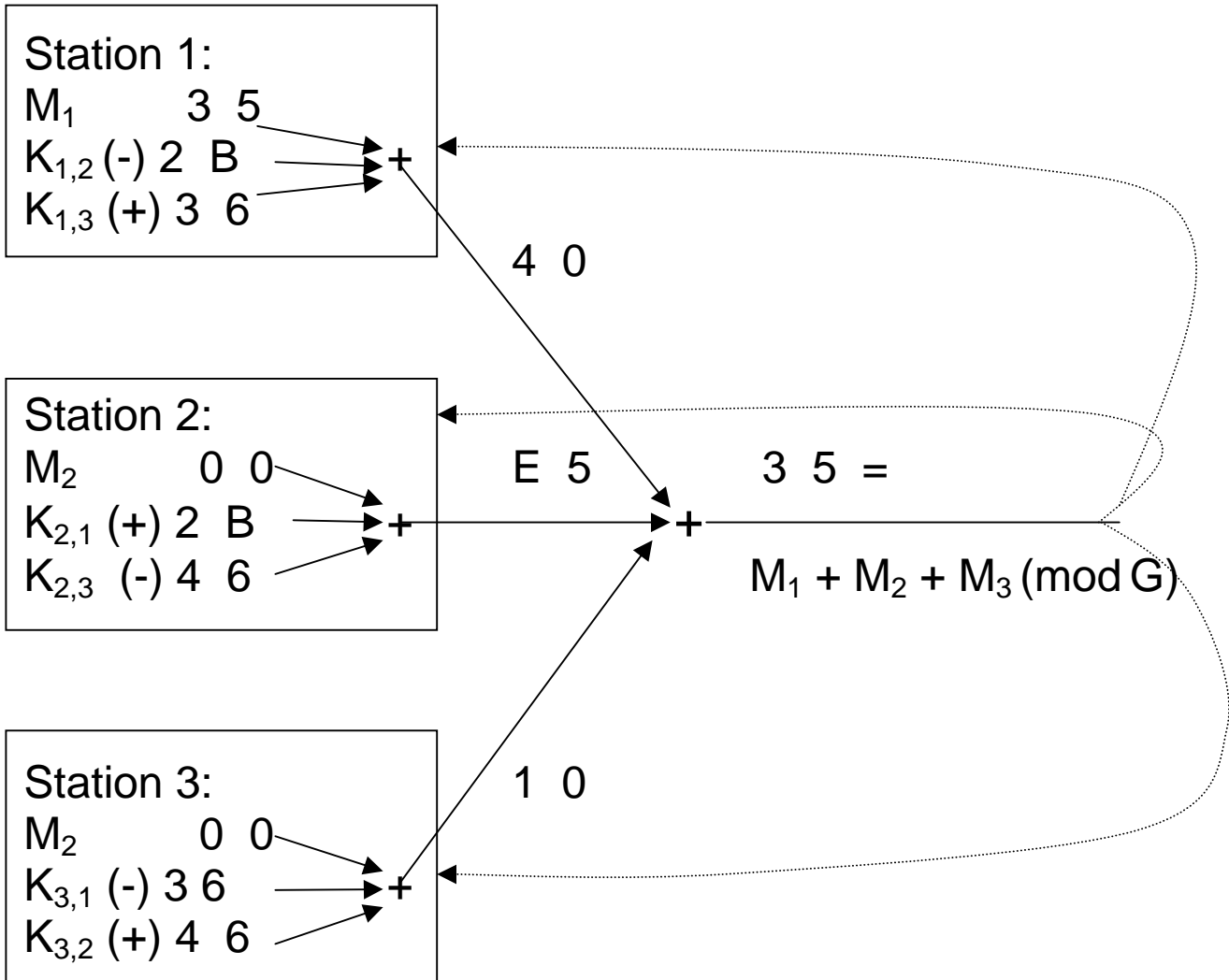
## 4.2 Binary superposed sending and broadcast



- Perfect sender anonymity through superposed sending (inversions are hidden by a one-time pad encryption)
- Recipient anonymity through broadcast and implicit addresses
- Message secrecy through encryption

General Superposed sending and broadcast :

Alphabet: (0,1,2,...,A,B,C,D,E,F)



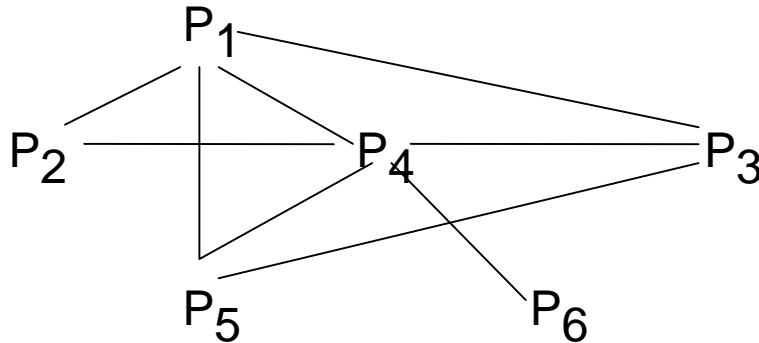
### 4.3 Generalised Approach: Superposed sending

$P = \{ P_1, \dots, P_m \}$  : set of all participants

$F$ : Alphabet of the DC-net

$(F, +)$  : Finite abelian group

Key graph  $G$ : undirected self-loop free graph with nodes  $P$



$\{P_1, \dots, P_n\}$  directly connected in  $G$ .

For each single sending step (round) :

- Each  $P_i, P_j$  directly connected in  $G$  choose a key  $K_{i,j} := K_{j,i}$  from  $F$  randomly and keep it secret
- Each  $P_i$  chooses  $M_i \in F$  and outputs his local sum:

$$O_i := M_i + \sum_{j=1}^{i-1} K_{i,j} - \sum_{j=i+1}^n K_{i,j}$$

and receives as input the global sum:

$$S := \sum_{i=1}^n O_i = \sum_{i=1}^n \left( M_i + \sum_{j=1}^{i-1} K_{i,j} - \sum_{j=i+1}^n K_{i,j} \right) \quad (3.1)$$

$$\Rightarrow S = \sum_{i=1}^n M_i \quad \text{since each key is both added and subtracted exactly once}$$

Lemma (superposed sending):

Let: A: subset of participants controlled by the attacker

Graph  $G \setminus (P \times A)$  is connected

$(O_1, \dots, O_n) \in F^n$ : output of a single round

Then:

For each vector  $(M_1, \dots, M_n) \in F^n$  which is consistent with the attacker's a priori knowledge about the  $M_i$

with 
$$\sum_{j=1}^n O_j = \sum_{j=1}^n M_j$$

the same number of key combinations exist which satisfy equation (3.1) and which is consistent with the attacker's a priori knowledge about  $K_{i,j}$ .

=>

An attacker gets no additional information about  $M_i$  besides  $\sum M_i$

a posteriori probability

$$p((M_1, \dots, M_n) | (O_1, \dots, O_n))$$

=

a priori probability

$$p((M_1, \dots, M_n) | \sum M_i) = p((M_1, \dots, M_n) | \sum O_i)$$

⇒ sender is perfectly anonymous  
(unconditionally untraceable)

## 4.4 Anonymity preserving multi-access protocols

Message: fixed number  $c$  of characters (containing also an implicit address)

slot:  $c$  consecutive rounds, in which a message is transferred

### a.) Slotted ALOHA:

- If  $P_i$  wants to send a message, he does so in the next slot
- If  $P_j$  also sends a message
  - $P_i$  (as well as  $P_j$ ) detects collision ( $S \neq M_j$ ).
  - $P_i$  retransmits his message after a random number of slots.

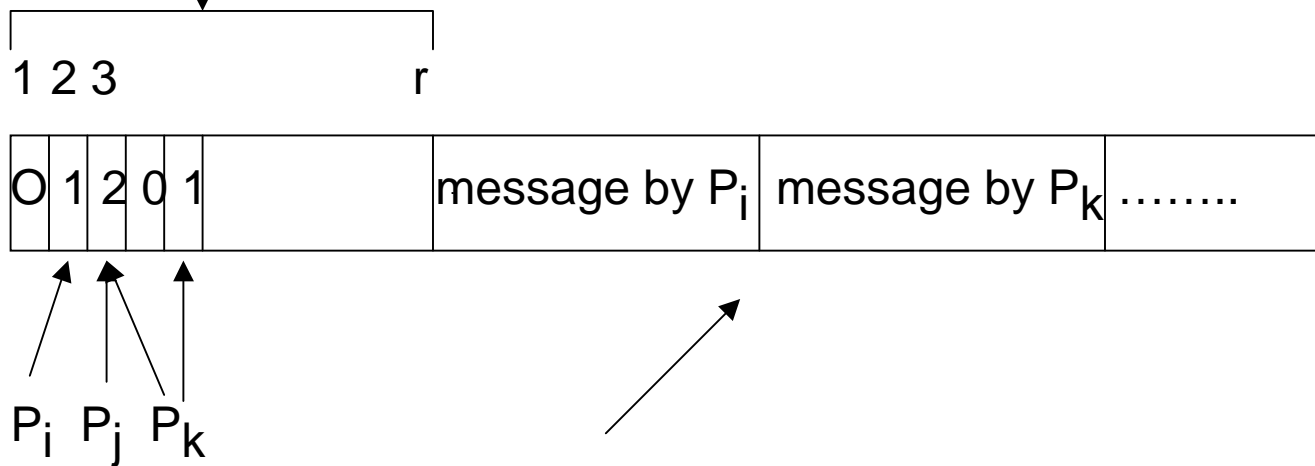
Retransmitted messages should be differently end-to-end encrypted. Otherwise:

If attacker monitors output messages  $X+Y$ ,  $X$ ,  $Y$

- attacker assumes:  $X$  and  $Y$  were sent by different participants

b.) Reservation map technique:

Reservation frame: slot of  $r$  rounds, used to reserve the following up to  $r$  slots



Message slots with reservation character  $\neq 1$  are skipped

## 4.5 Implementation Details:

### Establishing Keys:

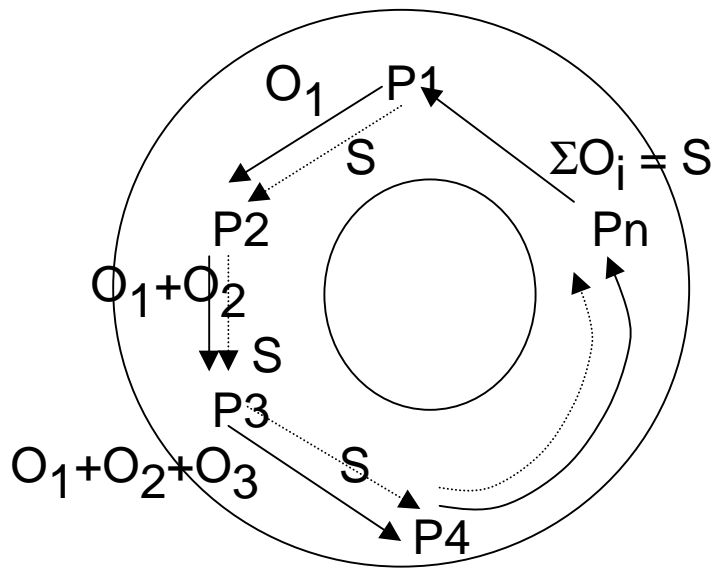
#### a.) Solution providing information-theoretic security:

- Storage of two identical long random-bit streams on separate optical disks
- Supplying one such disk to a communication partner

#### b.) Solution providing complexity-theoretic security:

Pseudorandomly generated keys

## Implementation-Example: Local-Area Ring Networks



First cycle:  $\longrightarrow$

Each participant adds his own output to the input he receives, forwards sum to the next participant

Second Cycle:  $\cdots \longrightarrow$

Result (global sum) is sent around to each participant

On average: fourfold increase in bandwidth

## 4.6 Active attacks on sender & recipient anonymity

For two way communication:

Perfect sender anonymity cannot be realised without perfect recipient anonymity (and vice versa)

Assume:

- Attacker  $P_a$  communicates with honest participant X
- X will answer message M by sending message  $M'$

$P_a$  can identify X as the sender of  $M'$



$P_a$  can identify X as the recipient of M

Attack scenario:

- Centre of star network collaborates with attacker  $P_a$
- $P_a$  delivers message M only to X and meaningless messages to all other participants
  - >  $P_a$  can check whether X is the receiver of M  
(in this case he responds with  $M'$ )
- More efficient attack by successively partitioning the set of participants
  - >  $P_a$  can identify X in  $\log(n)$  rounds

## 4.7 DC<sup>+</sup>-net using fail-stop broadcast [Waidner 1990]

### Fail-stop broadcast:

Message transmission is stopped as soon as two honest participants receive different input characters

### a.) Deterministic fail-stop key generation:

Keys  $K_{i,j}$ ,  $K_{j,i}$  depend completely on the input characters received by  $P_i$ ,  $P_j$

=>

differences between input characters  $I_i$  and  $I_j$  will disturb superposed sending

### Requirements for key generation scheme:

#### *(SuS) Superposed Sending:*

If for all rounds  $s = 1, \dots, t-1$  the input characters  $I_i^s$ ,  $I_j^s$  received by  $P_i$ ,  $P_j$  are equal, then the keys  $K_{i,j}^t$  and  $K_{j,i}^t$  for round  $t$  are equal and randomly selected from  $F$ .

$$[ \forall s < t: I_i^s = I_j^s ] \Rightarrow K_{i,j}^t = K_{j,i}^t \in_R F$$

#### *(FS) Fail Stop:*

If  $P_i$  and  $P_j$  receive different input characters  $I_i^s \neq I_j^s$  in round  $s < t$ , then the keys  $K_{i,j}^t$  and  $K_{j,i}^t$  for round  $t$  are independently and randomly selected from  $F$ .

$$[ \exists s < t: I_i^s \neq I_j^s ] \Rightarrow K_{i,j}^t \in_R F, K_{i,j}^t - K_{j,i}^t \in_R F$$

## Example for deterministic fail-stop key generation scheme:

$(F, +, \bullet)$ : finite field

$(a^1, a^2, \dots, a^{t_{\max}}), (b^1, b^2, \dots, b^{t_{\max}-1})$ :  
sequences whose elements are randomly  
selected from  $F$  and secretly exchanged  
between  $P_i$  and  $P_j$ .

Define for  $t = 1, \dots, t_{\max}$ :

$$K_{i,j}^t := a^t + \sum_{k=1}^{t-1} b^{t-k} \bullet l_{i,j}^k$$

$$K_{j,i}^t := a^t + \sum_{k=1}^{t-1} b^{t-k} \bullet l_{j,i}^k$$

It can be proven that above scheme satisfies (SuS) and (FS)

### However:

Each key generation scheme which deterministically satisfies (SuS) and (FS) requires at least

- The storage of all  $t_{\max} - 1$  received input characters
- $2 \bullet t_{\max} - 1$  secretly exchanged keys

b.) Probabilistic fail-stop key generation:

*Probabilistic version of (FS):*

If two honest participants receive two different input characters in round  $t$ , they will disturb superposed sending for the next  $d$  rounds

Example for probabilistic fail-top key generation scheme:

$$(a^1, a^2, \dots), (b^3, b^4, \dots) \in {}_R F^{\mathbb{N}}, c \in {}_R F$$

$$b^1, b^2, K_{i,j}^0, l_i^0 := 0$$

Then define for  $t \in \mathbb{N}$ :

$$K_{i,j}^t := a^t + (c + K_{i,j}^{t-1}) \bullet (b^t + l_i^{t-1})$$

## Problem:

Each participant can untraceably and permanently disrupt the entire DC-net

## Trap protocol providing servability [Chaum 1988]:

### Requirements:

- (1) The key-sharing graph is publicly agreed on
- (2) Each participant's output cannot be changed after hearing other participant's outputs
- (3) Some rounds can be contested without compromising the untraceability for non-disrupting senders
- (4) Reservation map technique is used

- Each  $P_i$  must reserve exactly one slot  $s_x$  in each reservation phase
- $P_i$  can either use  $s_x$  to send real message or a trap message
- If  $P_i$  wishes to send a trap, he announces an encrypted version of the message "*I will use the slot reserved by index  $x$  to send trap  $y$* " (trap proof)
- If trap set by  $P_i$  is disturbed =>  
 $P_i$  publishes trap proof in clear + encryption key

⇒ Attacker  $P_a$  can be prosecuted:

- Each honest  $P_j$  published  $M_j^t, K_{j,k}^t$   
used for rounds  $t$  of slot  $s_x$
- $P_a$  either admits having sent  $M_a^t \neq 0$ ,  
or modifies at least one  $K_{aj}^t \Rightarrow P_j$  detects  $K_{a,j}^t \neq K_{j,a}^t$ .

Problem:

$P_a$  could announce trap for slot  $s_x$  without reserving slot  $s_x$

=>

$P_a$  could deanonymise legitimate user of slot  $s_x$

Enhancement:

Trap announcement and traps are unambiguously linked by their slot number

(Further protocols for computationally secure servability:  
[Waidner / Pfitzmann 1989])