Code Voting With Linkable Group Signatures

Jörg Helbach Jörg Schwenk Sven Schäge

EVOTE08, Bregenz
09.08.2008

Outline

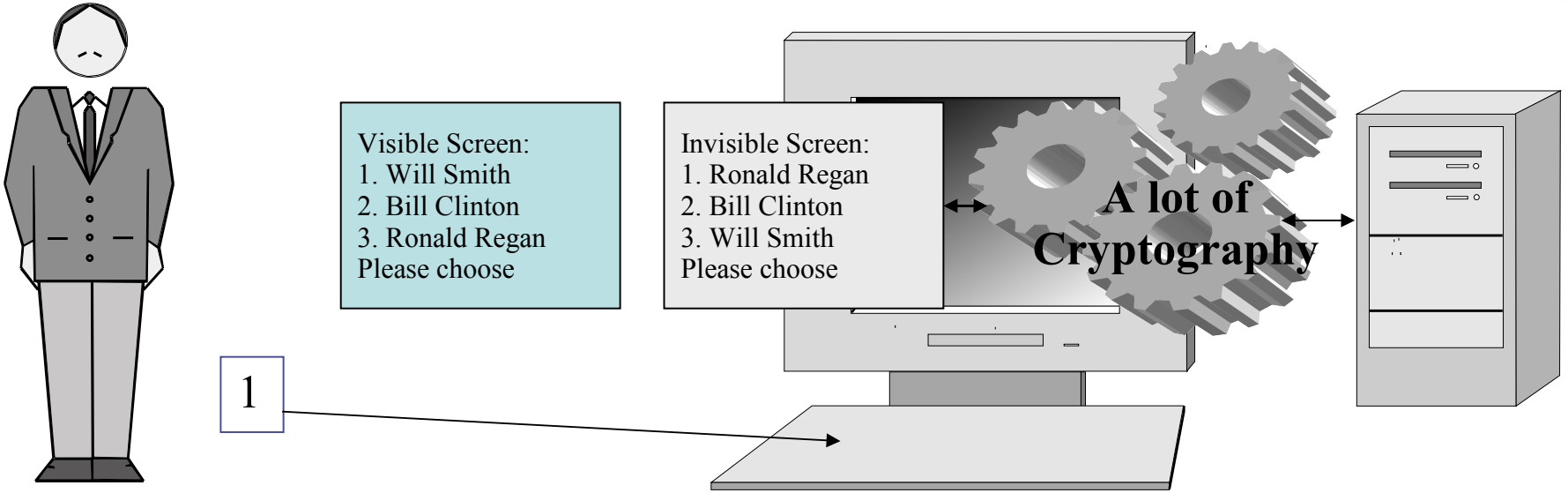
- Secure Platform Problem
- (Enhanced) Code Voting
- Group Signatures
- Vote Updating
- Voting scheme
- Security Properties
- Outlook

The Secure Platform Problem (SPP)

- Trojan Horse
  - controls PC system
  - input/output can be manipulated
  - especially: controls GUI
- Bot/Botnet
  - from 10.000 to 400.000 PCs controlled by a single person
  - used to make profit
  - difficult to control (tendency towards P2P networks)
  - Vint Cerf (2007): 25% of all PCs on the Internet have been compromised
- Rootkit
  - can remain undetected

eVoting and the SPP

- Browser based eVoting: Trojan Horses can be used to alter the vote
- GUI-based attack: Voting alternatives are displayed in different order



Secure Platform Problem

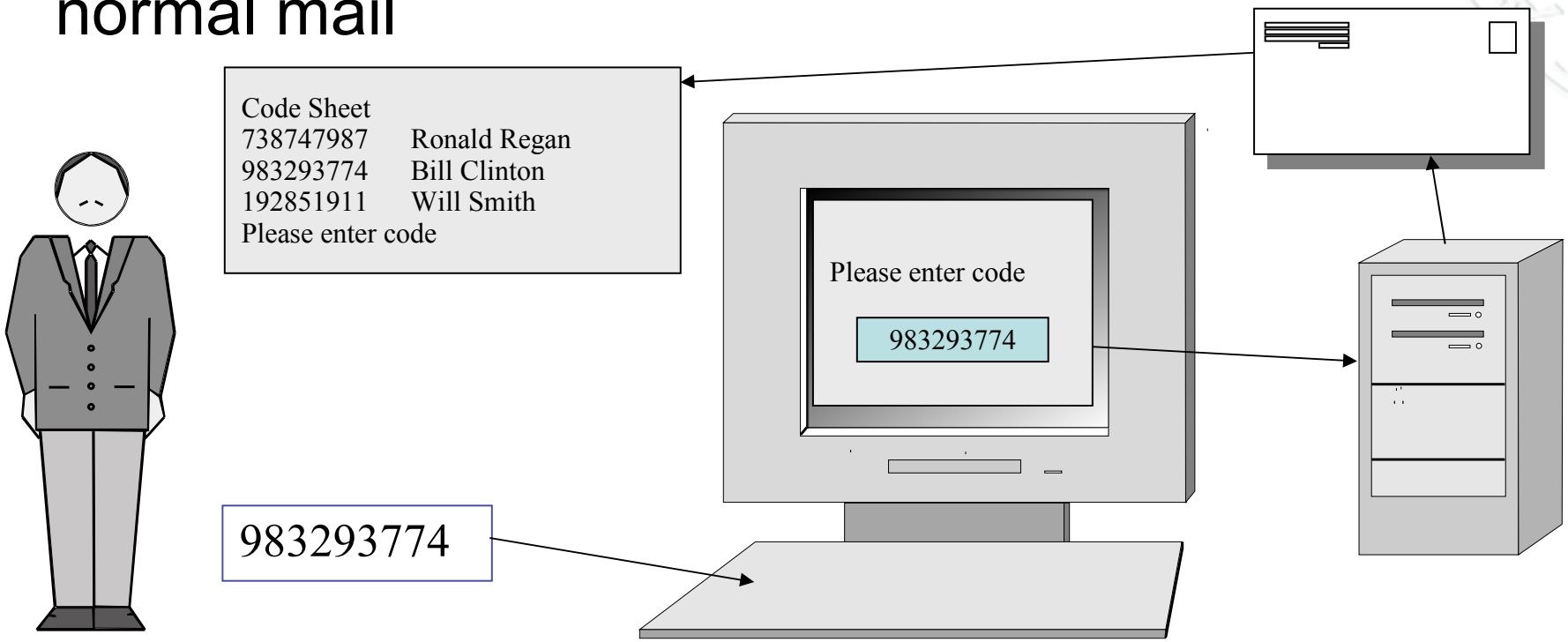
- If the attacker is able to control the communication channel between voter and PC, he can control the vote

- Two major options
  - Trusted Computing
  - Seperate Communication Channels

Code Voting: Standard Solution

Basic Idea:

- PC is used only for vote collection
- Information about the election is distributed over a secure (i.e. attacks do not scale well) channel, e.g. normal mail



Code Voting

- Vote Selling
- Assumptions
  - Trustworthy voting authority
  - Voting servers are reliable and secure
  - Voting codes are random
  - Code sheets are not distributed by electronical means
- Code Voting is secure against passive and partially secure against active attack (e.g. DoS)

Code Voting: Standard Solution

Security properties:

- Vote Selling — Is possible
- Receipt freeness — Yes
- Anonymity — Yes (random shuffling of code sheets)
- Security against malware attacks — Partially

Enhanced Code Voting

- Adding a confirmation TAN
- Receipt-freeness
- Vote-for-the-outsider-attack
  - Attacker simulates a “code error” message from the voting server, as long as the voter enters the same code
  - If the voter tries a second code from the code sheet (e.g. for an outsider candidate), this code is passed on to the server

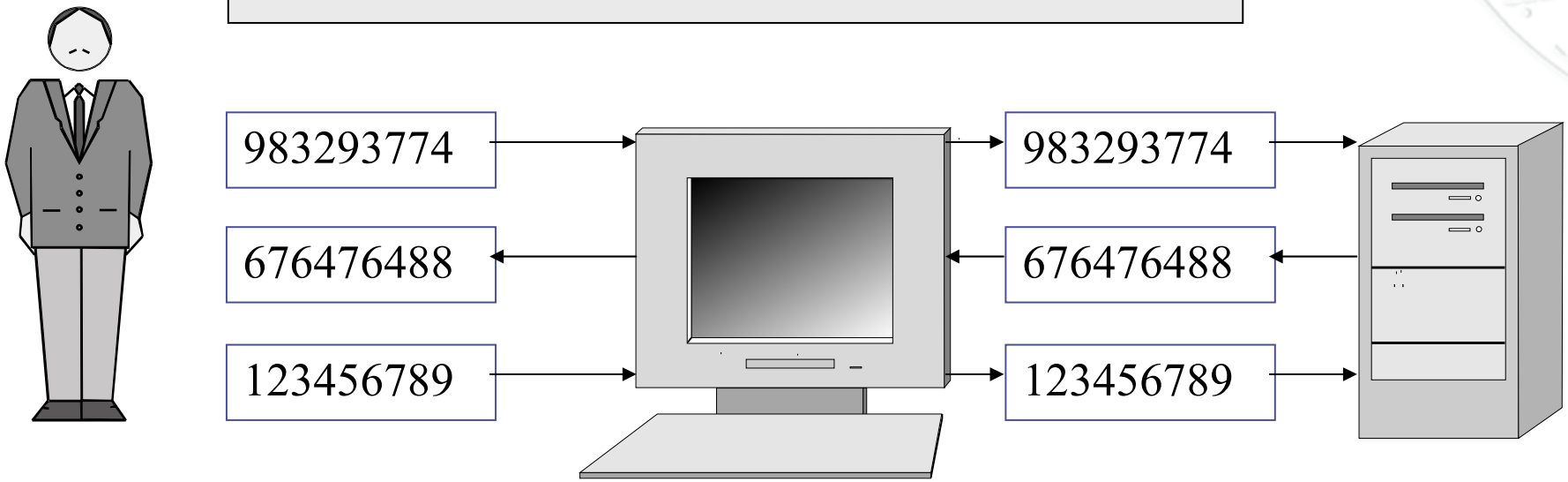
Enhanced Code Voting

- Neither the sender nor the receiver of a TAN could know, if his message was delivered sucessfull
- 2-Army-Problem
- 3-step-scheme
- Finalization TAN

Code Voting: Advanced Solution

- PC is used only for vote collection (vote updating allowed), vote confirmation and vote finalization

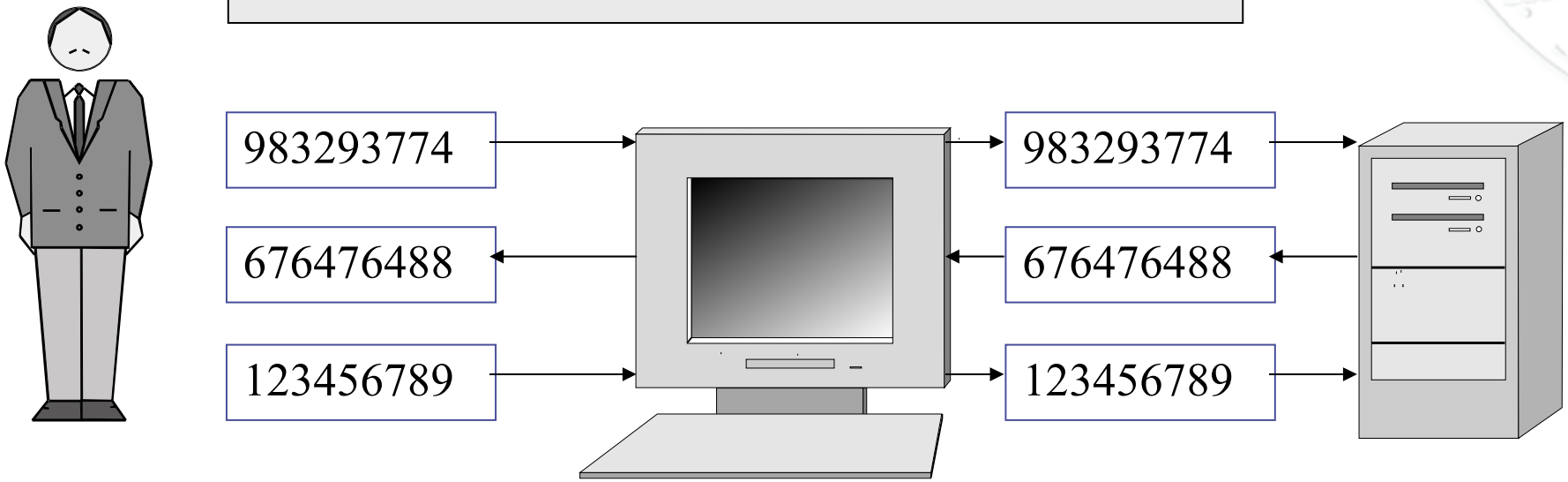
Voting TAN	**Candidate**	Confirmation TAN	Finalization TAN
738747987	**Ronald Reagan**	332676873	442367810
983293774	**Bill Clinton**	676476488	123456789
192851911	**Will Smith**	301287123	520172861



Code Voting: Advanced Solution

- PC is used only for vote collection (vote updating allowed), vote confirmation and vote finalization

Voting TAN	**Candidate**	Confirmation TAN	Finalization TAN
738747987	**Ronald Reagan**	332676873	442367810
983293774	**Bill Clinton**	676476488	123456789
192851911	**Will Smith**	301287123	520172861

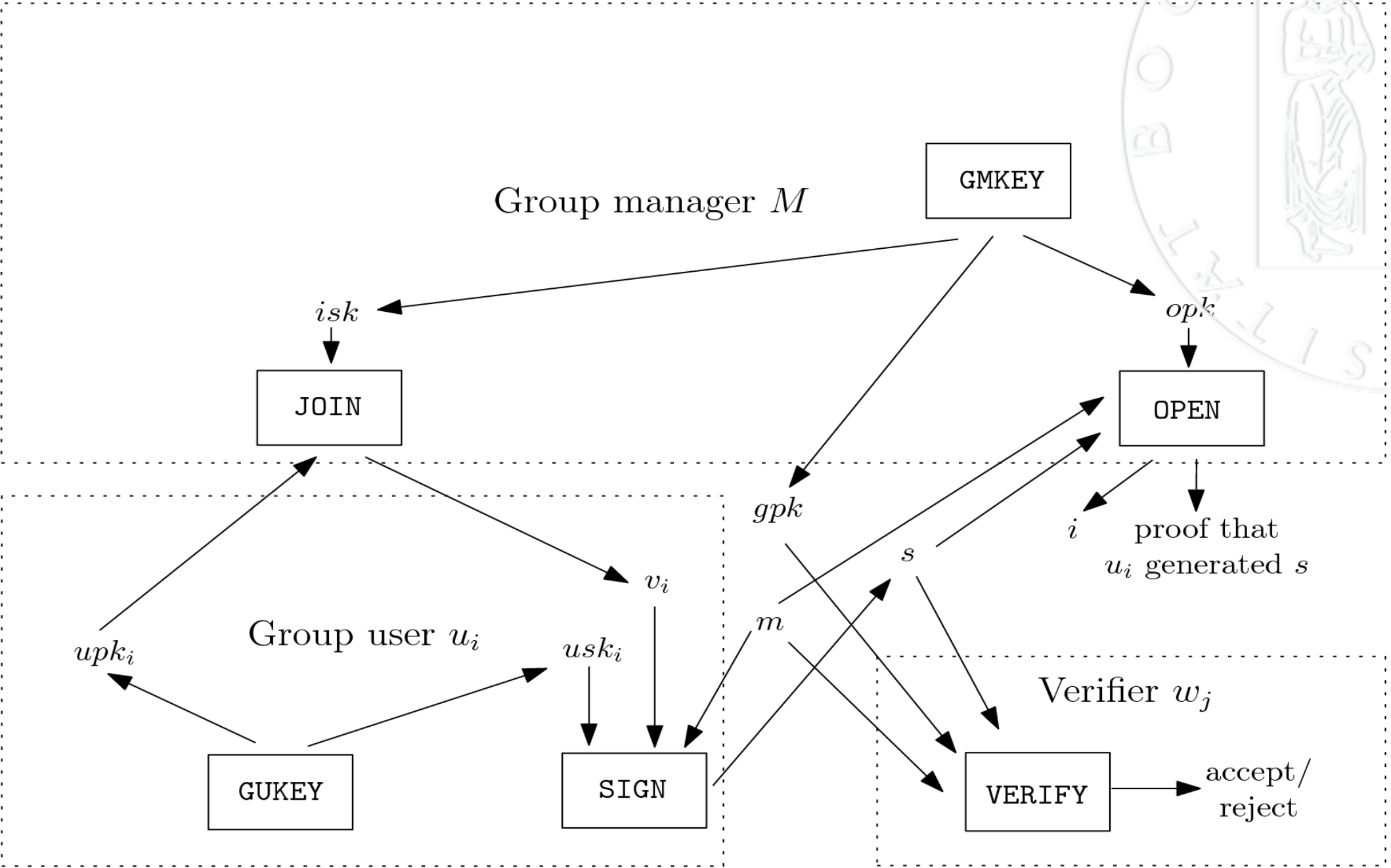


- BUT: Vote Selling Problem is still not solved!

Group Signatures

- Allow every member of a group to sign messages on behalf of it
- Signatures are unlinkable
  - It is not possible to identify which member has signed a particular message
  - It is not possible to verify if two signed messages were signed by the same group member

Group Signatures



Group Signatures

- Camenish and Stadler introduced the first efficient gs scheme in 1997
  - Group public key doesn't change even if a new member joins the group
  - The two different roles of a group manager (issuer and opener) is divideable

Group Signatures

- Signatures of knowledge are based on zero-knowledge protocols and used to convince a verifier that a given message was signed by a group member
- Used for the SIGN algorithm
- Group member proofs that he posesses a group certificate and knows the corresponding private key without revealing it
- Unlinkable because of a random commitment

Linkable group signatures

- Each group user is forced not to randomize his signature of knowledge

- Group manager *M* computes a digital signature scheme (sig_M, ver_M) and a public key encryption key pair (enc_M, dec_M)
- A voter builds $z=f(x)$ with x a random number and f a hash function and sends z to M
- M signs z building $v=sig_M(z)$

Linkable group signatures

- The voter can sign his message *m* using $d=enc_M(m,z)$
- The voter can proof, using a signature of knowledge, that he knows *x* and *v* satisfying $enc_M(m,f(x))$ and $ver_M(v,f(x))=true$

Vote updating

- A good method to prevent vote selling, but cannot be the only measure against vote selling

- E.g. additional protection measures could be
  - Multiple code sheets
  - Powerful voting credentials, e.g. ID card

Voting scheme

- Voting authority is divided into different groups:
  (1) Print and issue code sheets
  (2) Operate the voting servers and databases
  (3) Manage the group signature scheme
  (4) Verify that every eligible voter only casts one ballot

- Private group signature key is issued to the voter, e.g. bound to an electronic passport (3)
- Voting Authority prints and shuffles code sheets (1)
- Voting Authority issues a code sheet to every (1) eligible voter

Voting scheme

- Voter transfers his voting TAN and signs it with his private group signature key
- After receiving the correct confirmation TAN (2), the voter transfers his signed finalization TAN
  - If the voters gets no or a faulty confirmation TAN the client may be infected with malware. Assuming a transferable gs scheme, he may vote again using another client
- For every transfered finalization TAN the voting authority verifies that the voter only cast one ballot (4)

Security Properties

- Vote Selling — Is not possible, if the voter cannot sell his private gs key
- Receipt freeness — Yes
- Anonymity — Yes (random shuffling of code sheets)
- Security against malware attacks — Yes

Outlook

- Usability issues
- Verifiability
- Treshold scheme in conjunction with group signatures

Thank you for your attention.