Paillier-Evoting

Roles

Connor: Paillier encryption / decryption & tabulation Jacob: Client stuff Kiana: BB and ZKP

Scheme:

EB has a known RSA public key & hidden private EB has a known Paillier public key & hidden private

Voter has private blind signature keys

1. Voter:
    i. Requests candidates
    ii. Generates vote V & encrypts using EB's public Paillier
    iii. Creates authorization token A using known string and hash of encrypted vote
    iv. Generates ZKP of correctness & validity of votes
    v. Blinds A & V using private keys
2. Voter submits blinded A to EB
    i. EB verifies registration
    ii. EB signs blinded A with private RSA
    iii. Returns signed, blinded A to voter
    iv. Voter un-blinds A
3. Voter:
    i. Submits signed A & V, and ZKP
4. BB:

    i. Verifies validity of A & signature using EB's public RSA & checks for string
    ii. Verifies validity of V using using EB's public RSA, hash in A, & ZKP
    iii. Saves V
5. Upon election close:
    i. BB sends all V to CA
    ii. CA sums all votes
    iii. CA sends sums to EB
    iv. EB decrypts using private Paillier & releases results

Requirements:

1. python2.7
2. Update Pailler Evoting/paillier submodule (https://github.com/kcmcnellis/paillier.git)
3. `sudo pip install -r requirements.txt`

To run:

1. `cd Pailler\ Evoting`
2. `python evoting_main.py`
3. `python evoting_client.py`

To run using gui:

`python evoting_gui.py`

Assumptions

1. Everything in the `database` file is securely stored and only the server has access to it.
2. Under `keyserver`, the `Public` folder is public knowledge (on a trusted keyserver). The `Private` folder is securly stored and only the server has access to it
3. The client can access the server only when an election is running, and does so from a unique, private terminal or computer
4. Each voter knows his/her voter_id and no others'. It was previously distributed offline.

Security

The ZNPs used (knowledge of plaintext and plaintext is in a given set) are from Practical Multi-Candidate Election System (O. Baudron, P.-A. Fouque, D. Pointcheval, G. Poupard, and J. Stern)

Libraries

1. Paillier encryption : https://github.com/mikeivanov/paillier (forked, a copy is provided in `Pailler Evoting/paillier`)
2. RSA encryption : pycrypto
3. Sockets & threading : eventlet