LELEC2770

Project: Hands on remote voting

1 Goal

The goal of the project is to write all the code of the remote voting protocol seen in class. More precisely, you have to complete the code given for a voter, a tallier and the bulletin board participating in an election.

2 Description of the election

At the beginning of the election, an empty bulletin board is created with the same structure as in Figure 1. The status is set to "`init`". During the initialization phase, every tallier can generate a partial ElGamal key-pair and append the public key (with a proof of correctness) to the bulletin board. When every tallier is done, the "`start_voting_phase`" method is called and the status of the election is set to "`vote`".

During the voting phase, every voter can generate a ballot and append it to the bulletin board. A ballot can be seen as a list of `n_choices` candidates for which the voter has to choose "yes" (an encryption of 1) or "no" (an encryption of 0). For each of those choices, the ballot also contains a disjunctive proof that the ciphertext contains either 0 or 1. The code for generating disjunctive proofs is already given. Finally, the ballot may have another constraint depending on the value of `n_selections`. If this value is between 1 and `n_choices`-1, then the voter has to select exactly `n_selections` times the "yes" option in their ballot. In that case, it is called a "complex ballot" and it contains an additional proof of correctness. Otherwise, `n_selections` = -1. The ballot is called a "simple ballot" and the proof is omitted. `n_choices` and `n_selections` are parameters of the elections published on the bulletin board. When every voter is done, the "`start_tally_phase`" method is called and the status of the election is set to "`tally`".

During the tallying phase, every tallier discards the invalid ballots, computes their decryption factor and appends it to the bulletin board with a proof of correctness. Finally, the validity of the bulletin board can be checked, and if everything is alright, the result can be computed by anyone.

```
<status> ::= "init" | "vote" | "tally"
<group> ::= {"p": <p>, "g": <g>}

<tallier_key> ::= {
    "pk": <y>,
    "proof": <key_correctness_proof>,
}

<key_correctness_proof>::= {
    "commit": <commit>,
    "response": <response>
}

<ballot> ::= {
    "vote_encryptions": [<selection_ct_0>, <selection_ct_1>, ... ],
    "zero_one_proofs": [<zero_one_proof_0>, <zero_one_proof_1>, ... ],
    ("k_selection_proof": <k_selection_proof>,)
}
<selection_ct> ::= {
    "c1": <c1>,
    "c2": <c2>,
}
<zero_one_proof> ::= {
    "commit": [[<commit_0_0>, <commit_0_1>], [<commit_1_0>, <commit_1_1>]],
    "response": [<response_0>, <response_1>],
    "challenge": [<challenge_0>, <challenge_1>],
}
<k_selection_proof> ::= {
    "commit": [<commit_0>, <commit_1>],
    "response": <response>,
}

<dec_factor> ::= {
    "pk": <y>,
    "df": [<df_0>, <df_1>, ... ],
    "proof": [<tally_correctness_proof_0>, <tally_correctness_proof_1>, ... ],
}
<tally_correctness_proof> ::= {
    "commit": [<commit_0>, <commit_1>],
    "response": <response>,
}

<bulletin_board> ::= {
    "status": <status>,
    "group": <group>,
    "n_choices": <n_choices>,
    "n_selections": <n_selections>,
    "tallier_keys": [ <tallier_key_0>, <tallier_key_1>, ... ],
    "ballots": [<ballot_0>, <ballot_1>, ... ],
    "decryption_factors": [ <dec_factor_0>, <dec_factor_1>, ... ],
}
```

Figure 1: Structure of the bulletin board. All elements which are not defined (such as `<g>`, `<c1>`, ...) are integers between 0 and `<p>`.

3 Organization

The project has to be done by group of 1 or 2 students (use the moodle forum to find a project partner if needed). Discussion with other students about the project is welcome (including through the moodle forum), but sharing of code is not allowed. We will check for plagiarism.

4 Deadline

The deadline is 23:59 on December 5th.

5 Deliverable

A zip file only containing (at its root) the files `bb.py`, `voter.py` and `tallier.py` and a file `README.txt` that should contain your names and any information of interest. Your code should be **interoperable**. That is, it should be possible to interact with your bulletin board with the code of the voter of another group and the code of the tallier of yet another group.

Beware, do not modify the files `utils.py`, `group.py` and `number.py` as they cannot be part of your submission.

Make you code as readable as possible (see `https://www.python.org/dev/peps/pep-0008/` for guidance). Excellent tools for this are docstrings and comments.