DEMTECH
DEMOCRATIC TECHNOLOGY

Preserving voter's privacy with homomorphic encryption

Lorena Ronquillo – IT Universitetet i København
lron@demtech.dk

Summary

E-voting schemes based on homomorphic encryption provide universal verifiability. With homomorphic encryption, encrypted ballots can be added to obtain the encryption of the tally without having to be individually decrypted first. However, nothing prevents the election authority from decrypting individual balllots instead of the election result, thus violating voter's privacy. To avoid this, a secret sharing scheme is often used to split the decryption key into a set of shares that will be given to several authorities, assuming that it is unlikely for a certain subset of authorities to be all dishonest and willing to know individual votes.

In this work we study a different decryption mechanism that lets the authority decrypt any ciphertext obtained as a result of performing some operation on a set of encrypted ballots, but prevents the decryption of a ciphertext that encrypts one single ballot.

Participants

Let V be the number of voters, N the number of candidates, and an election authority. All participants are seen as probabilistic Turing machines, which can perform polynomial-time computations.
Participants can communicate with each other through a public channel with memory: the Bulletin Board (BB).

Homomorphic encryption

Let $E(r, m)$ denote a probabilistic encryption of message m using the random parameter r. Consider a plaintext space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$ such that $\mathcal{M}$ is a group under the binary operation $\oplus$, and $\mathcal{C}$ is a group under the operation $\otimes$.
A probabilistic encryption scheme is said to be $(\otimes,\oplus)$-homomorphic, if given $c_1 = E(r_1, m_1)$ and $c_2 = E(r_2, m_2)$, there exists an r such that

$$c_1 \otimes c_2 = E(r, m_1 \oplus m_2).$$

Exponential ElGamal

Exponential ElGamal is a $(\cdot,+)$-homomorphic (additive) variant of the well-known ElGamal public-key cryptosystem [2].

Let p and q be large primes such that $q \mid (p-1)$, and let g be a generator of the order-q subgroup of $\mathbb{Z}_p^\star$, for which the discrete logarithm problem is considered to be intractable.

- private key: s, drawn uniformly at random from $\mathbb{Z}_q$.
- public key: $h = g^s \bmod p$. The values p, q, and g are also public parameters.

We assume that each vote is a yes or no for each candidate, encoded by 1 and 0, respectively.

Casting a vote

Each voter $1 \le i \le V$ posts to the BB an encrypted ballot c_{ij} of the form

$$c_{ij} = (R_{ij}, S_{ij}) = (g^{r_{ij}}, g^{m_{ij}} h^{r_{ij}})$$

for every candidate $1 \le j \le N$, where $m_{ij} \in \{0, 1\}$ is her vote to candidate j, and r_{ij} is a random value. Together with the encrypted ballot, a non-interactive proof of knowledge that proves in zero-knowledge the validity of the ballot is also published.

Result of the election

Considering only valid ballots, the authority obtains an encryption of the sum of votes for every candidate $1 \le j \le N$ as follows

$$\prod_{i=1}^{V} c_{ij} = (X_j, Y_j) = \left(\prod_{i=1}^{V} R_{ij} \bmod p, \prod_{i=1}^{V} S_{ij} \bmod p\right) = (g^{r'_j}, g^{m'_j} h^{r'_j}), \quad (1)$$

where $r'_j = \sum_{i=1}^{V} r_{ij}$, and $m'_j = \sum_{i=1}^{V} m_{ij}$.

Table 1 gives an overview of the tallying process from the information published at the bulletin board.

Finally, in order to decrypt the result of the election, the authority computes for every candidate $1 \le j \le N$

$$m'_j = \log_g \frac{Y_j}{(X_j)^s} = \log_g g^{m'_j}. \quad (2)$$

The total number of votes going for m'_j for candidate j can be efficiently computed by using methods like Baby step giant step [1], or the Pollard rho method [3].

Problem:

- the whole responsibility of the election relies on one authority.
- performing the same operation as in (2) the authority could equally decrypt individual encrypted ballots (R_{ij}, S_{ij}) instead of the encryption (X_j, Y_j) of the result, thus violating voter's privacy.

BB	cand. 1	...	cand. N
voter 1	$c_{11} = (g^{r_{11}}, g^{m_{11}} h^{r_{11}})$	⋮	$c_{1N} = (g^{r_{1N}}, g^{m_{1N}} h^{r_{1N}})$
⋮	⋮	⋮	⋮
voter V	$c_{V1} = (g^{r_{V1}}, g^{m_{V1}} h^{r_{V1}})$	⋯	$c_{VN} = (g^{r_{VN}}, g^{m_{VN}} h^{r_{VN}})$
	↓		
	$\prod_{i=1}^{V} c_{i1}$	⋯	$\prod_{i=1}^{V} c_{iN}$

Table 1 : Computation of the tally.

Secret sharing

Both in its versions with or without dealer, secret sharing schemes are used to split a secret key s and distribute the corresponding shares among several authorities in such a way that only some predefined coalitions of authorities can later reconstruct the secret. One of the most common schemes is Shamir's (t, T) secret-sharing scheme [4], which only allows to any coalition of at least t from T authorities to get the secret.

Problem:

- A coalition of t dishonest authorities can recover the secret key before having the election result.

Decrypting only the tally

The tally can be decrypted by using the random value of the encryption of the sum of votes r'_j.
Once we have the encryption of the sum of votes for a certain candidate j, as in (1), and knowing the sum of the randomness, r'_j, one can compute $h^{r'_j}$ and decrypt the election result by doing as follows

$$m'_j = \log_g \frac{Y_j}{h^{r'_j}} = \log_g g^{m'_j}.$$

Advantages of decrypting with the randomness:

- The secret key is only used to generate the corresponding public key, after that it can be permanently destroyed/forgotten.
- Privacy: given the randomness r'_j, only the ciphertext containing the tally can be decrypted, and not the individual encrypted ballots.
- Verifiability: Unlike the private key, the value r'_j can be made public for anyone willing to check the decryption.

Ongoing work

Analyze and evaluate the pros and cons of the possible ways in which voters can secretly communicate the sum of the random values they used to encrypt their ballots.

Some of the options under consideration:

- Voters are given a pre-computed random value to encrypt their ballot. Problem: authority will again have all responsibility of the election. Who would generate this value and how will it be communicated to the voters?
- Voters use (Pedersen) commitment schemes to commit to the random value they used to encrypt their ballot. Problem: secure channel between the voter and the server. Assuming untappable, or anonymous and untappable channels is impractical. Assuming only anonymous channel is more practical.
- Voters securely add their random values through a multiparty computation and send the final result to the election authority.

References

[1] Cohen, H., "A course in Computational Algebraic Number Theory", Springer, Berlin, 1993.

[2] ElGamal, T. "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". IEEE Transactions on Information Theory 31 (4): 469–472, 1985.

[3] Pollard, J.M. "A Monte Carlo method for factorization", BIT Numerical Mathematics 15 (3): 331–334, 1975.

[4] Shamir, A. "How to share a secret", Communications of the ACM 22 (11): 612–613, 1979.