Projet de Cryptographie

Vote Electronique

David Pointcheval

A rendre: le 8 janvier 2018

david.pointcheval@ens.fr

1 Description

1.1 Chiffrement ElGamal

On considère un groupe $\mathbb{G}$ d'ordre q, engendré par un élément g, dans lequel le logarithme discret et le problème Diffie-Hellman peuvent être admis difficiles. Pour cela, plusieurs possibilités, en fonction de votre langage préféré :

- la librairie OpenSSL en C ;
- la librairie BouncyCastle en java ;
- la librairie pycrypto en python

Il sera important de comprendre certaines API bas-niveau pour utiliser le chiffrement ElGamal en tant que chiffrement additivement homomorphe :

- Génération KeyGen d'une paire de clés : $x \xleftarrow{\$} \mathbb{Z}_q$ et $y \leftarrow g^x$;
- Chiffrement $\mathsf{Enc}(m)$ d'un message $m \in \mathbb{Z}_q$: $r \xleftarrow{\$} \mathbb{Z}_q$, $c \leftarrow g^r$ and $c' \leftarrow y^r g^m$
- Déchiffrement $\mathsf{Dec}(C)$ d'un chiffré $C = (c, c')$: $M \leftarrow c'/c^x$, puis calcul du logarithme discret m de M en base g, soit donc m tel quel $g^m = M$

On peut constater que $\mathsf{Dec}(\mathsf{Enc}(m_0) \times \mathsf{Enc}(m_1)) = m_0 + m_1 \bmod q$.

1.2 Procédure de vote

Pour exprimer son vote lors d'un référendum, le vote $v \in \{0, 1\}$ est chiffré en tant que scalaire : $C = \mathsf{Enc}(v)$. La propriété additive du chiffrement ElGamal permet de déterminer le résultat de façon chiffrée en multipliant tous les chiffrés (composante par composante) : $R = \prod C_i$; puis il suffit alors de déchiffrer R pour obtenir le résultat.

2 Développement d'un système de vote électronique

2.1 Système de base

Il conviendra alors de permettre l'exécution des différentes étapes suivantes, en appelant le programme `vote` avec différents arguments :

- `vote -keygen <key>` qui génère une paire de clés, et stocke la clé secrète dans le fichier `<key>` puis la clé publique dans le fichier `<key>.pub` ;
- `vote -key <key> -vote <v> <file>` qui génère un bulletin chiffré (sous la clé publique `<key>.pub`) contenant le vote `<v>` (0 ou 1), et le concatène au fichier `<file>` ;
- `vote -key <key> -decrypt <file> <result>` qui déchiffre tous les chiffrés contenus dans le fichier `<file>` et stocke les clairs dans le ficher `<result>` ;
- `vote -key <key> -randvote <n> <file>` qui génère `<n>` bulletins chiffrés (sous la clé publique `<key>.pub`) contenant des votes aléatoires (parmi 0 ou 1), et les concatène au fichier `<file>` ;
- `vote -combine <file> <result>` qui combine tous les chiffrés du fichier `<file>` en le chiffré de la somme des clairs et stocke le résultat dans le fichier `result` ;

L'utilisation réelle consistera donc à

- générer les clés ;
- générer de nombreux votes aléatoires ;
- combiner ces votes chiffrés en le chiffré du résultat ;
- effectuer le déchiffrement final.

2.2 Améliorations

Plusieurs améliorations peuvent être apportées, indépendamment et/ou combinées :

- partager la clé de déchiffrement et effectuer le déchiffrement de façon distribuée ;
- générer les clés et effectuer le déchiffrement de façon distribuée ;
- prouver que le déchiffrement est correct, sans révéler la clé de déchiffrement ;
- prouver que le déchiffrement distribué est correct, sans révéler les clés partielles de déchiffrement.

3 Livrable

Le livrable attendu est une archive, envoyée par mail, qui contient

- les sources et les librairies nécessaires ;
- un fichier `readme.txt` qui explique ce qui est fourni (langage, système de base, améliorations) ;
- un fichier `Makefile` qui par défaut effectue la séquence ci-dessus de vote aléatoire et de dépouillement.