1. In many scenarios it is necessary that ciphertexts can be decrypted only if at least \( t \) out of \( n \) parties agree to do it. For example in an e-voting system we would like to distribute the decrypting power between many independent authorities.

In the following we look at \((t,n)\)-threshold Elgamal cryptosystem. Let \( G \) be a size \( q \) Schnorr’s group and \( g \) a generator of \( G \).

**Key Generation.** We assume a trusted dealer\(^1\) that sets up keys of each party as follows:

- It picks \( a_i \leftarrow r \mathbb{Z}_q \) for \( i = 0, \ldots, t - 1 \) which defines a polynomial \( f(X) := \sum_{i=0}^{t-1} a_i X^i \).
- Secret key is \( s := f(0) = a_0 \) and public key is \( h := g^s \). Secret key \( s \) is erased/ignored.
- Each party \( P_i \) for \( i = 1, \ldots, n \) gets a private value \( s_i := f(i) \) and \( h_i := g^{s_i} \) is published.

**Encryption.** Encryption works as usual, that is, to encrypt a message \( m \in G \) we pick \( r \leftarrow r \mathbb{Z}_q \) and compute \( c = (c_0, c_1) = (m h^r, g^r) \).

**Decryption.** Decryption requires participation of at least \( t \) distinct parties \( P_{\alpha_1}, \ldots, P_{\alpha_t} \). Remember that a degree \( t - 1 \) polynomial is uniquely determined by \( t \) points (line is determined by 2 points, parabola is determined by 3 points, etc). Let us define \( \ell_{\alpha_i}(X) := \prod_{j=1}^{t} \frac{X - \alpha_j}{\alpha_i - \alpha_j} \); this is a unique degree \( t - 1 \) polynomial such that \( \ell_{\alpha_i}(\alpha_i) = 1 \) and \( \ell_{\alpha_i}(\alpha_j) = 0 \) for all \( \alpha_j \neq \alpha_i \). Above allows us to define \( F(X) := \sum_{i=1}^{t} s_{\alpha_i} \cdot \ell_{\alpha_i}(X) \). Since \( F(\alpha_i) = f(\alpha_i) \) for \( i = 1, \ldots, t \), then \( F(X) = f(X) \).

Hence parties can collectively compute \( F(0) = f(0) = s \) in the exponent. More precisely, each party \( P_{\alpha_i} \) can output a partial decryption \( d_i = c^{s_{\alpha_i} \ell_{\alpha_i}(0)} = (g^r)^{s_{\alpha_i} \ell_{\alpha_i}(0)} \) which allows to compute \( \prod_{i=1}^{t} d_i = (g^r)^{\sum_{i=1}^{t} s_{\alpha_i} \ell_{\alpha_i}(0)} = g^{rs} = h^r \). Finally the message can be recovered by computing \( m h^r / h^r = m \).

Implement the above protocol and use [https://courses.cs.ut.ee/MTAT.07.014/2018_fall/uploads/Main/threshold_elgamal_template.txt](https://courses.cs.ut.ee/MTAT.07.014/2018_fall/uploads/Main/threshold_elgamal_template.txt) as a template. As usual, feel free to modify the template.

(5pt)

2. Suppose that parties \( P_1, \ldots, P_t \) wish to decrypt a ciphertext \( c \), but party \( P_t \) is malicious. Show that \( P_t \) can trick the decryption process into outputting \( m^* \) where \( m^* \) can be any message of \( P_t \)’s choosing. Propose a fix to the issue.

**Hint:** one possible fix uses zero-knowledge proofs.

(4pt)

---

\(^1\)For example keys could be generated by a hardware security module (HSM) – a special-purpose tamper-resistant hardware device.