1. Preuves Zero-Knowledge

1.1 Preuve de connaissance d'un logarithme discret

On considère un groupe cyclique G, engendré par g, d'ordre premier q.

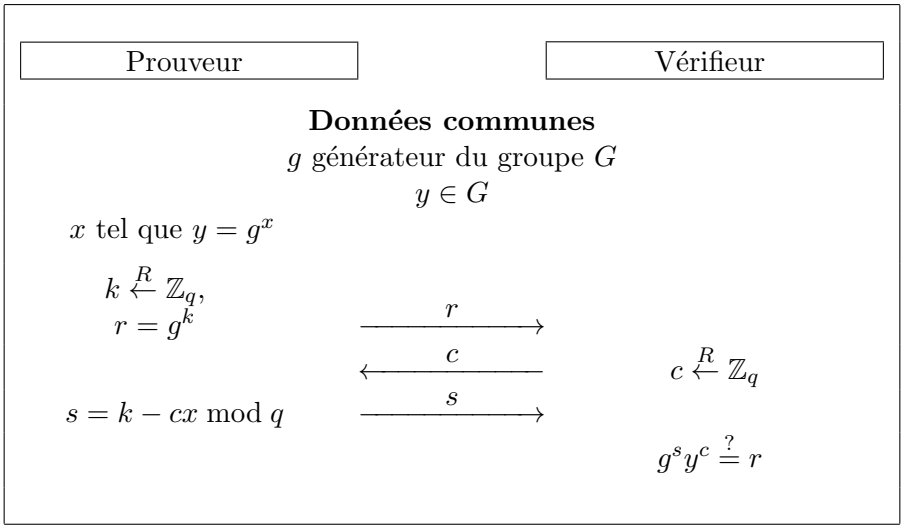


FIGURE 1 – Schéma de Schnorr

Q-1. Montrer la complétude et la correction.

Dans une approche exacte, on considère un adversaire probabiliste qui est accepté avec probabilité $\varepsilon > 1/q$ et dont le temps de calcul est T.

Q-2. Evaluer le temps de calcul moyen T' et la probabilité de succès ε' de l'extracteur en fonction de T, ε, q et τ, où τ est le temps d'exécution du schéma élémentaire.

Q-3. Montrer que ce protocole est ZK face à un vérifieur honnête, et évaluer la complexité de la simulation. Qu'en est-il face à un vérifieur malhonnête ?

1.2 Preuve de connaissance d'une information parmi plusieurs

On considère m utilisateurs d'un système d'identification du type Guillou-Quisquater. On veut proposer un schéma par lequel l'un quelconque de ces utilisateurs peut s'identifier sans qu'il soit possible de déterminer lequel. On note n l'entier RSA sur lequel le Guillou-Quisquater est basé, e l'exposant premier, ℓ un paramètre de sécurité tel que $2^\ell < e$, et t le nombre d'itérations du protocole. On propose le schéma présenté figure 2, répété t fois, dans lequel le j-ième utilisateur joue le rôle du prouveur.

Q-4. Montrer la complétude et la correction. (On se bornera à expliciter ce qu'est une situation favorable, et comment on en extrait la racine e-ième de l'un des x_i.)

Q-5. Expliquer pourquoi le protocole est ZK et pourquoi un adversaire ne peut déterminer lequel des utilisateurs s'est identifié.

Q-6. Dans une approche asymptotique, quel type d'hypothèse doit-on faire sur ℓ, t et m ? On notera k le paramètre de sécurité $k = \log n$.

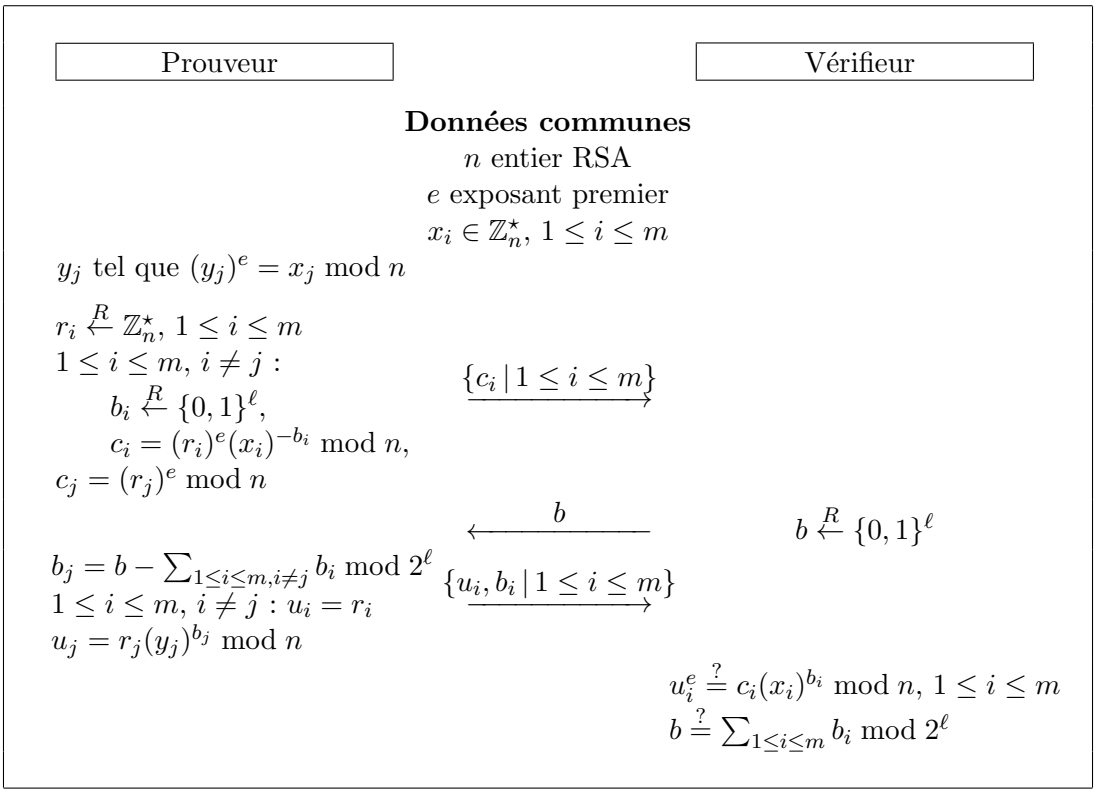


FIGURE 2 – Schéma de un-parmi-m GQ

2. Quelques réductions

2.1 Problèmes aléatoirement auto-réductibles

Considérons le problème RSA, pour un module $n = pq$ et un exposant e premier avec $\varphi(n)$. On dit qu'un algorithme $\mathcal{A}$ est un (ε, t)-adversaire contre le problème RSA (n, e) si, en temps t, sa probabilité de résoudre une instance aléatoire est supérieure à ε :

$$\mathsf{Succ}^{\mathsf{rsa}} n, e(\mathcal{A}) = \Pr_{x \in \mathbb{Z}_n^\star}[\mathcal{A}(n, e, x^e \bmod n) = x] \geq \varepsilon.$$

Q-1. Montrer que s'il existe un (ε, t)-adversaire contre le problème RSA (n, e), alors il existe un algorithme qui résoud toute instance (avec probabilité proche de 1) en temps "raisonnable".

Décrire cet algorithme, et estimer ce temps "raisonnable".

2.2 Algorithme avec erreur contre le problème Diffie-Hellman

Dans la réduction ci-dessus, l'algorithme de départ résoud une instance RSA avec probabilité ε. Dans les autres cas, une réponse erronée ou un constat d'échec de la part de l'algorithme, sont équivalents puisque l'erreur est facilement détectée.

Considérons désormais le problème Diffie-Hellman sur un groupe cyclique $\mathcal{G}$ (noté multiplicativement) d'ordre premier q (dont on désignera un générateur g). Le problème Diffie-Hellman dans $\mathcal{G}$, en la base g, consiste à calculer, sur les entrées $X = g^x$ et $Y = g^y$, la valeur $Z = g^{xy}$. On dit qu'un algorithme $\mathcal{A}$ est un (ε, t)-adversaire contre le problème Diffie-Hellman $(\mathcal{G}, g)$ si, en temps t, sa probabilité de résoudre une instance aléatoire est supérieure à ε :

$$\mathsf{Succ}^{\mathsf{cdh}} \mathcal{G}, g(\mathcal{A}) = \Pr_{x,y \in \mathbb{Z}_q^\star}[\mathcal{A}(\mathcal{G}, g, g^x, g^y) = g^{xy}] \geq \varepsilon.$$

Q-2. Peut-on utiliser la même méthode que pour RSA ?

On veut cependant utiliser $\mathcal{A}$ pour résoudre l'instance fixée $(A = g^a, B = g^b)$, avec une bonne garantie du résultat. On dérive alors de cette instance une série d'instances $(X_i = A^{\alpha_i} g^{u_i}, Y_i = B^{\beta_i} g^{v_i})$, avec des exposants $\alpha_i, \beta_i, u_i, v_i$ aléatoires dans $\mathbb{Z}_q^\star$.

Q-3. Montrer qu'en cas de succès sur une instance (X_i, Y_i), on en déduit la solution pour (A, B). Montrer qu'en cas d'échec, cette même opération conduit à un élément parfaitement aléatoire dans $\mathcal{G}$.

Une méthode naturelle est donc de retourner la première collision, comme étant la solution. Il est en effet clair que deux succès conduisent tous deux à la solution, donc constituent une collision.

Q-4. Estimer la probabilité d'erreur, ainsi que le temps moyen d'exécution.

Il aurait été plus simple et plus rapide d'utiliser une séquence $(X_i = A^{\alpha_i}, Y_i = B^{\beta_i})$, avec des exposants α_i, β_i aléatoires dans $\mathbb{Z}_q^\star$.

Q-5. Montrer qu'un attaquant (tout puissant) pourrait nous induire en erreur, alors qu'avec le premier type de séquences, même un attaquant tout-puissant ne peut que nous aider à atteindre notre objectif.

2.3 Méthode des hybrides

Considérons deux distributions Δ_0 et Δ_1 sur un même ensemble $\mathcal{S}$. Un distingueur entre ces deux distributions est un algorithme capable d'avoir un comportement différent selon qu'il travaille sur une instance provenant de Δ_0 ou de Δ_1. Plus précisément, un (ε, t)-distingueur des distributions Δ_0 et Δ_1 est un algorithme $\mathcal{D}$ qui fonctionne en temps t, et possède un avantage $\mathsf{Adv}^{\Delta}(\mathcal{D})$ supérieur à ε où

$$\mathsf{Adv}^{\Delta}(\mathcal{D}) = \Pr_{\delta\in\Delta_0}[\mathcal{D}(\delta)=1] - \Pr_{\delta\in\Delta_1}[\mathcal{D}(\delta)=1].$$

Q-6. Montrer que s'il existe un distingueur $\mathcal{D}'$ entre Δ_0^n et Δ_1^n en temps t avec un avantage ε, alors il existe un distingueur $\mathcal{D}$ entre les distributions Δ_0 et Δ_1 avec un avantage supérieur à ε/n.

En déduire l'avantage maximal en temps t entre Δ_0^n et Δ_1^n (noté $\mathsf{Adv}^{\Delta^n}(t)$) en fonction de l'avantage maximal en temps t entre Δ_0 et Δ_1 (noté $\mathsf{Adv}^{\Delta}(t)$).

Considérons à nouveau le problème Diffie-Hellman. Comme on vient de le voir, le problème Diffie-Hellman (calculatoire) dans $\mathcal{G}$, en la base g, consiste à calculer, sur les entrées $X = g^x$ et $Y = g^y$, la valeur $Z = g^{xy}$. Il en existe une variante, appelée problème Diffie-Hellman décisionnel, qui consiste pour un algorithme (distingueur) de décider si, pour un triplet (X, Y, Z) dans $\mathcal{G}$, Z est effectivement la solution au problème Diffie-Hellman calculatoire ou non. On dit que $\mathcal{D}$ est un (ε, t)-distingueur contre le problème Diffie-Hellman décisionnel dans $\mathcal{G}$, en la base g, s'il fonctionne en temps t, et possède un avantage $\mathsf{Adv}^{\mathsf{ddh}}\mathcal{G}, g(\mathcal{D})$ supérieur à ε où

$$\mathsf{Adv}^{\mathsf{ddh}}\mathcal{G}, g(\mathcal{D}) = \Pr_{x,y\in\mathbb{Z}_q^\star}[\mathcal{D}(\mathcal{G}, g, g^x, g^y, g^{xy})=1] - \Pr_{x,y,z\in\mathbb{Z}_q^\star}[\mathcal{D}(\mathcal{G}, g, g^x, g^y, g^z)=1].$$

En d'autres termes, si l'on considère les distributions suivantes :

$$\begin{aligned}\mathcal{DH}(\mathcal{G}, g) &= \{\mathcal{I} = (g^x, g^y, g^{xy}) \,|\, x, y \in \mathbb{Z}_q^\star\}\\ \mathcal{R}and(\mathcal{G}, g) &= \{\mathcal{I} = (g^x, g^y, g^z) \,|\, x, y, z \in \mathbb{Z}_q^\star\}\end{aligned}$$

$$\mathsf{Adv}^{\mathsf{ddh}}\mathcal{G}, g(\mathcal{D}) = \Pr_{\mathcal{I}\in\mathcal{DH}(\mathcal{G},g)}[\mathcal{D}(\mathcal{G}, g, \mathcal{I})=1] - \Pr_{\mathcal{I}\in\mathcal{R}and(\mathcal{G},g)}[\mathcal{D}(\mathcal{G}, g, \mathcal{I})=1].$$

Q-7. Montrer que s'il existe un distingueur $\mathcal{D}'$ entre $\mathcal{DH}(\mathcal{G}, g)^n$ et $\mathcal{R}and(\mathcal{G}, g)^n$ en temps t avec un avantage ε, alors il existe un distingueur $\mathcal{D}$ contre le problème Diffie-Hellman décisionnel avec un avantage supérieur à ε, en un temps $t' \leq t + \mathcal{O}(n)$.

3. Schémas de chiffrement

3.1 Sécurité sémantique et indistingabilité des chiffrés

En 1984, Goldwasser et Micali ont défini la notion de sécurité sémantique par " tout ce qui peut être calculé efficacement sur le clair avec le chiffré, peut être calculé sans le chiffré. " Ceci peut alors être formalisé de la façon suivante : on considère un attaquant $\mathcal{A} = (A_1, A_2)$ en deux étapes. Dans un premier temps, l'algorithme A_1, à la vue de la clé publique pk, retourne une distribution sur l'ensemble des messages, caractérisée par un algorithme d'échantillonnage M. Un tel algorithme ne doit retourner avec une probabilité non nulle que des messages de même taille. Dans un deuxième temps, l'algorithme A_2 reçoit le chiffré y d'un message aléatoire x (selon la distribution M). Cet adversaire retourne une fonction f, et une valeur α.

Un tel attaquant réussit dans son attaque si $f(x) = \alpha$ avec une meilleure probabilité que sur un message aléatoire inconnu $f(x^\star) = \alpha$.

$$\mathsf{Adv}^{\mathsf{sem}}(\mathcal{A}) = \left|\mathsf{Succ}^{M}(\mathcal{A}) - \mathsf{Succ}^{\$}(\mathcal{A})\right|, \text{ avec}$$

$$\left.\begin{array}{rcl} \mathsf{Succ}^{M}(\mathcal{A}) & = & \Pr\left[\alpha = f(x)\right] \\ \mathsf{Succ}^{\$}(\mathcal{A}) & = & \Pr\left[\alpha = f(x^\star)\right] \end{array}\right\} \begin{array}{l} \text{sur l'espace de probabilités défini par} \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (M, s) \leftarrow A_1(\mathsf{pk}), \\ x, x^\star \leftarrow M, y = \mathcal{E}_{\mathsf{pk}}(x; r), (f, \alpha) \leftarrow A_2(y, M, s). \end{array}$$

Q-1.

- Montrer que la sécurité sémantique est équivalente à l'indistingabilité des chiffrés.
- Montrer que cette équivalence subsiste même si f est restreinte à retourner une sortie binaire.

3.2 Chiffrement de Goldwasser et Micali

En 1984, suite à la définition de la sécurité sémantique, Goldwasser et Micali ont proposé le premier schéma de chiffrement sémantiquement sûr.

- Génération des clés : étant donné un paramètre de sécurité k, on choisit un module RSA N sur k bits, ainsi qu'un non-résidu quadratique $\alpha \in \mathbb{Z}_N^\star$ (comme cas particulier, on peut considérer $N = pq$, où $p, q = 3 \bmod 4$ et $\alpha = -1$.
- Chiffrement d'un bit b : $c = x^2\alpha^b \bmod N$, pour $x \xleftarrow{R} \mathbb{Z}_N^\star$.

Q-2.

- Décrire l'algorithme de déchiffrement.
- Montrer que ce schéma est bien sémantiquement sûr. Préciser l'hypothèse algorithmique nécessaire et suffisante.

3.3 Chiffrement de Paillier

En 1999, Pascal Paillier a proposé une extension du schéma de Goldwasser et Micali, mais dans un contexte particulier : $N = pq$ est la clé publique, puis $\mathcal{E}(m; r) = (1 + N)^m r^N \bmod N^2$.

Q-3.

— Montrer que la fonction

$$\begin{array}{rcccl} f & : & \mathbb{Z}_N \times \mathbb{Z}_N^\star & \to & \mathbb{Z}_{N^2}^\star \\ & & (m, r) & \mapsto & (1+N)^m r^N \bmod N^2 \end{array}$$

est un isomorphisme de groupes.

— Montrer que la factorisation de N permet de l'inverser.

— Montrer que le calcul de racines N-ièmes modulo N permet de l'inverser.

— En déduire l'algorithme de déchiffrement.

— Préciser les hypothèses algorithmiques sur lesquelles reposent les notions OW-CPA et IND-CPA.

3.4 Chiffrement de Rabin

En 1978, Rabin propose une alternative à RSA, basée sur le problème de la racine carrée modulaire.

— Génération des clés : étant donné un paramètre de sécurité k, on choisit un module RSA $N = pq$ sur $2k + 1$ bits.

— Chiffrement d'un message $x \in \mathbb{Z}_N$: $c = x^2 \bmod N$.

Q-4. Décrire l'algorithme de déchiffrement. Montrer qu'une redondance dans le message clair est nécessaire.

On propose la redondance : $x = m \| 0^{k_1}$, où m est le message à chiffrer sur $2k - k_1$ bits, et les k_1 bits de poids faible de x sont imposés à 0.

Q-5. Décrire l'algorithme de déchiffrement associé. La OW-CPA est-elle équivalente à la factorisation ? Exprimer le coût de la réduction en fonction de k_1.

4. Chiffrement asymétrique IND-CCA2

4.1 Conversion générique

En 1999, Fujisaki et Okamoto ont proposé une construction générique, conduisant à du chiffrement asymétrique IND-CCA2, à partir d'un chiffrement faible.

Soit un schéma probabiliste $\mathcal{S} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$:

- $\mathcal{K}(1^k)$: retourne un couple de clés $(\mathsf{sk}, \mathsf{pk}) \in \mathcal{SK} \times \mathcal{PK}$.
- $\mathcal{E}_{\mathsf{pk}}(m; r)$: pour un message $m \in \mathcal{M}$ et l'aléa $r \in \mathcal{R}$, retourne le chiffré $c \in \mathcal{C}$ résultant.
- $\mathcal{D}_{\mathsf{sk}}(c)$: pour un chiffré $c \in \mathcal{C}$, retourne le message clair $m \in \mathcal{M}$, ou $\perp$ si le chiffré n'est pas valide.

On dérive le schéma $\mathcal{S}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ suivant, avec deux fonctions de hachage

$$G : \{0,1\}^\star \to \{0,1\}^{k_1} \qquad H : \{0,1\}^\star \to \mathcal{R}$$

- $\mathcal{K}'(1^k)$: exécute $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{K}(1^k)$, retourne les clés $(\mathsf{sk}, \mathsf{pk})$.
- $\mathcal{E}'_{\mathsf{pk}}(m; r)$: pour un message m de k_1 bits, et un aléa $r \in \mathcal{M}$, on calcule $s = H(m, r)$, puis $c_1 \leftarrow \mathcal{E}_{\mathsf{pk}}(r; s)$. Ensuite, on calcule $c_2 = m \oplus G(r)$. Le chiffré consiste en le couple $C = (c_1, c_2)$.

Q-1. Décrire l'algorithme de déchiffrement $\mathcal{D}'_{\mathsf{sk}}$. Préciser les espaces de départ et d'arrivée.

4.2 Sécurité

On définit, pour tout algorithme $\mathcal{A}$, son succès contre la non-inversibilité :

$$\mathsf{Succ}^{\mathsf{ow-cpa}}_{\mathcal{S}}(\mathcal{A}) = \Pr_{\substack{m \in \mathcal{M} \\ r \in \mathcal{R}}} [(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{K}(1^k) : \mathcal{A}(\mathcal{E}_{\mathsf{pk}}(m; r)) \stackrel{?}{=} m].$$

De même, pour tout algorithme $\mathcal{A}^{\mathcal{D}} = (A_1^{\mathcal{D}}, A_2^{\mathcal{D}})$, qui a accès à l'oracle de déchiffrement $\mathcal{D}$, on définit son avantage contre la sécurité sémantique :

$$\mathsf{Adv}^{\mathsf{ind-cca2}}_{\mathcal{S}}(\mathcal{A}) = \left| 2 \times \Pr_{\substack{b \in \{0,1\} \\ r \in \mathcal{R}}} \left[\begin{array}{l} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{K}(1^k), (m_0, m_1, \sigma) \leftarrow A_1^{\mathcal{D}}(\mathsf{pk}) \\ c \leftarrow \mathcal{E}_{\mathsf{pk}}(m_b; r) : A_2^{\mathcal{D}}(c, \sigma) \stackrel{?}{=} b \end{array} \right] - 1 \right|.$$

Enfin, $\mathsf{Succ}^{\mathsf{ow-cpa}}_{\mathcal{S}}(t)$ (resp. $\mathsf{Adv}^{\mathsf{ind-cca2}}_{\mathcal{S}}(t)$) désigne le succès (resp. l'avantage) maximal qu'un attaquant peut obtenir en temps t.

Q-2. Montrer le lien qui existe entre $\mathsf{Succ}^{\mathsf{ow-cpa}}_{\mathcal{S}}(t)$ et $\mathsf{Adv}^{\mathsf{ind-cca2}}_{\mathcal{S}'}(t')$, dans le modèle de l'oracle aléatoire (les fonctions G et H sont considérées parfaitement aléatoires).

Pour cela, on fera évoluer la distribution de probabilités qui définit les réponses des oracles aléatoires G et H, ainsi que r et b impliqués pour la sécurité sémantique ($\mathsf{Adv}^{\mathsf{ind-cca2}}$). Puis on étudiera, en fonction des différentes distributions de probabilités, les événements S (la réponse b' de l'attaquant est égale au bit b ci-dessus), AskG (la question $G(r)$ est posée, où r est l'aléa utilisé dans le challenge c), AskH (la question $H(m_b, r)$ est posée) et AskR (r est posé à G ou H, soit $\mathsf{AskR} = \mathsf{AskG} \vee \mathsf{AskH}$).

Dans un premier temps, on rend la vue de l'attaquant indépendante de b, ainsi la probabilité de l'événement S est exactement $1/2$. Ensuite, on simule l'oracle de déchiffrement : On peut, dès le début, transmettre toute question $H(m, r)$ à $G(r)$, et stocker dans les listes Λ_H et Λ_G toutes les questions-réponses obtenues par l'attaquant auprès de H et G. Puis,

1. r^+ et g^+ sont choisis, pour définir respectivement r et $G(r)$, dans le challenge $C = (c_1, c_2)$ (avec $c_2 \leftarrow m_b \oplus g^+$), et pour les réponses de G (avec $G(r) \leftarrow g^+$) ;
2. On conserve $c_2 = m_b \oplus g^+$, mais $G(r)$ est défini indépendamment de g^+ ;
3. s^+ est choisi, pour définir s et donc $H(m_b, r)$, dans le challenge $C = (c_1, c_2)$ (avec $c_1 \leftarrow \mathcal{E}_{\mathsf{pk}}(r, s^+)$), et pour les réponses de H (avec $H(m_b, r) \leftarrow s^+$) ;
4. On conserve $c_1 = \mathcal{E}_{\mathsf{pk}}(r, s^+)$, mais $H(m_b, r)$ est défini indépendamment de s^+ ;
   Quelle est la probabilité de S_4 ?
5. On simule l'oracle de déchiffrement, grâce à la liste Λ_H ;
6. c_1^+ est choisi aléatoirement, pour définir c_1 dans le challenge C. Cela définit alors implicitement r^+ et s^+. On remarquera que AskR_6 est relié à $\mathsf{Succ}^{\mathsf{ow-cpa}}$.

Remarques :

— On pourra utiliser la valeur $\lambda(\mathcal{S})$, définie comme étant le maximum, pour $\mathsf{pk} \in \mathcal{PK}$, $m \in \mathcal{M}$ et $c \in \mathcal{C}$, de $\Pr_{r \in \mathcal{R}}[\mathcal{E}_{\mathsf{pk}}(m; r) = c]$.

— On notera q_G et q_H le nombre de questions posées aux oracles G et H respectivement, ainsi que q_D le nombre de questions posées à l'oracle de déchiffrement $\mathcal{D}'$.

4.3 Chiffrement RSA

On a vu que la primitive RSA, $f_{e,N}(x) = x^e \bmod N$, fournit un schéma de chiffrement OW-CPA. On envisage d'utiliser une version probabiliste de la forme

$$\mathcal{E}_{e,N}(m; r) = f_{e,N}(m) \| r.$$

Q-3. Détailler le schéma de chiffrement $\mathcal{S}$ ainsi obtenu, en précisant les tailles de chaque élément, ainsi que les espaces de départ et d'arrivée.

Q-4. Calculer le niveau de sécurité garanti par ce schéma, soit $\mathsf{Succ}_{\mathcal{S}}^{\mathsf{ow-cpa}}(t)$ en fonction de $\mathsf{Succ}^{\mathsf{rsa}(k)}(t)$, la probabilité d'inverser la fonction RSA sur des modules RSA de k bits en temps t. Puis donner un majorant de $\lambda(\mathcal{S})$. On expliquera alors l'intérêt de « r ».

Q-5. Préciser le schéma $\mathcal{S}'$ qui résulte de l'application de la conversion précédente (présentée section 4.1) à ce schéma $\mathcal{S}$.

Q-6. Évaluer le niveau de sécurité garanti par $\mathcal{S}'$: une borne supérieure de $\mathsf{Adv}_{\mathcal{S}'}^{\mathsf{ind-cca2}}(t)$ en fonction de $\mathsf{Succ}^{\mathsf{rsa}(k)}(t)$.

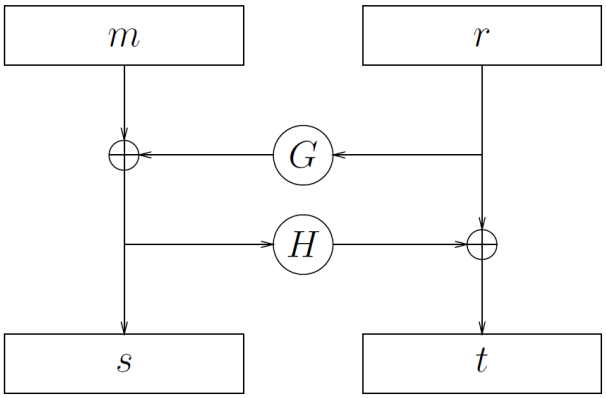
5. OAEP : Optimal Asymmetric Encryption Padding

5.1 Description

En 1994, Bellare et Rogaway ont proposé le padding suivant à appliquer avant l'utilisation d'une permutation à sens-unique à trappe f_{pk} (telle RSA) de $\{0,1\}^k$, où

$$G : \{0,1\}^{k_0} \to \{0,1\}^n \text{ et } H : \{0,1\}^n \to \{0,1\}^{k_0},$$

avec $k = k_0 + n$.



C'est-à-dire que pour chiffrer un message $m \in \{0,1\}^n$, on applique la construction ci-dessus avec un aléa $r \xleftarrow{R} \{0,1\}^{k_0}$, pour obtenir le couple $(s,t) = \mathsf{OAEP}(m,r)$. On applique alors la permutation f_{pk} sur l'objet $s\|t$ pour obtenir le chiffré $c = \mathcal{E}_{\mathsf{pk}}(m;r)$.

> **Q-1.** Décrire l'algorithme de déchiffrement en fonction de g_{sk}, la permutation réciproque de f_{pk}, accessible à qui connaît la trappe sk.

5.2 Sécurité sémantique

Soit $y = f_{\mathsf{pk}}(s\|t)$, le chiffré du message $m_b = m \in \{m_0, m_1\}$, avec l'aléa r (donc $(s,t) = \mathsf{OAEP}(m_b, r)$ pour $b \in \{0,1\}$).

> **Q-2.** Montrer que, dans le modèle de l'oracle aléatoire où les fonctions G et H sont modélisées par des fonctions parfaitement aléatoires, sans avoir posé la question $G(r)$, un attaquant a un avantage nul pour deviner b (soit si $m = m_0$ ou si $m = m_1$).

Rappel : L'avantage d'un attaquant contre la sécurité sémantique d'un schéma de chiffrement $\pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ est

$$\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) = 2 \times \Pr_{\substack{b\leftarrow\{0,1\} \\ r\leftarrow\{0,1\}^{k_0}}} \left[\begin{array}{l} (\mathsf{pk},\mathsf{sk}) \leftarrow \mathcal{G}(1^k); (m_0,m_1,s) \leftarrow A_1(\mathsf{pk}) \\ c = \mathcal{E}_{\mathsf{pk}}(m_b;r) : A_2(m_0,m_1,s,c) = b \end{array} \right] - 1.$$

> **Q-3.** Préciser l'avantage d'un attaquant qui n'aurait pas posé la question $H(s)$.

Q-4. En déduire que cette construction conduit à un schéma IND-CPA, dans le modèle de l'oracle aléatoire, sous réserve que f_{pk} soit une permutation à sens-unique à trappe sur $\{0,1\}^k$.

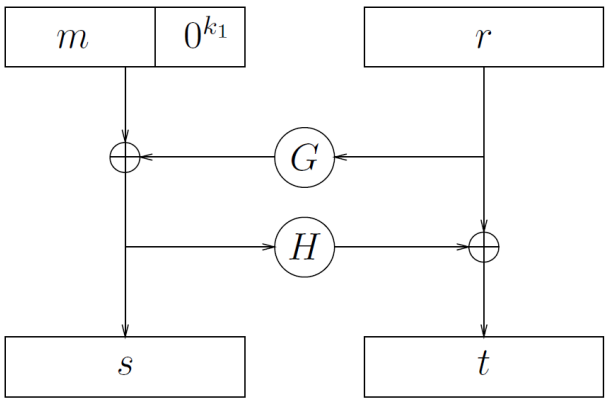
Note : Pour cela, on précisera l'avantage maximal, en temps t, d'un attaquant qui peut poser au plus q_g et q_h questions aux oracles G et H respectivement. On pourra utiliser $\mathsf{Succ}^{\mathsf{ow}}_f(t)$, le maximum pour tout algorithme $\mathcal{A}$ en temps t de

$$\mathsf{Succ}^{\mathsf{ow}}_f(t) = \max_{\mathcal{A}} \left\{ \Pr_{\mathsf{pk},x}[\mathcal{A}(f_{\mathsf{pk}}(x)) = x] \right\},$$

où pk et x sont choisis selon les distributions convenables : $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathcal{G}(1^k)$ et $x \leftarrow \{0,1\}^k$.

5.3 Non-malléabilité

L'objectif initial de OAEP était de conduire à un schéma de chiffrement sûr contre les attaques à chiffrés choisis à partir de toute permutation à sens-unique à trappe. Pour cela, une redondance était nécessaire : $(s,t) = \mathsf{OAEP}'(m,r) = \mathsf{OAEP}(m\|0^{k_1},r)$, avec $m \in \{0,1\}^{n-k_1}$ et $r \in \{0,1\}^{k_0}$.



Un chiffré est considéré valide si l'inversion de f_{pk} fait bien apparaître cette redondance 0^{k_1}. Et alors, ce qui précède cette série de 0 est retourné en tant que message clair. Dans le cas contraire, le chiffré est refusé.

Considérons une permutation à sens-unique à trappe F_{pk} de $\{0,1\}^{k_0}$, pour laquelle il existe un algorithme $\mathcal{A}$ qui calcule $F_{\mathsf{pk}}(x \oplus a)$ à partir de pk, a et $y = F_{\mathsf{pk}}(x)$:

$$\mathcal{A}(\mathsf{pk}, a, y = F_{\mathsf{pk}}(x)) = F_{\mathsf{pk}}(x \oplus a).$$

Définissons la fonction de $f_{\mathsf{pk}}(s\|t) = s\|F_{\mathsf{pk}}(t)$, pour $s \in \{0,1\}^n$ et $t \in \{0,1\}^{k_0}$.

Q-5. Montrer que f_{pk} est une permutation de $\{0,1\}^k$ à sens-unique à trappe. Pour cela, on évaluera $\mathsf{Succ}^{\mathsf{ow}}_f(t)$ en fonction de $\mathsf{Succ}^{\mathsf{ow}}_F(t)$.

Q-6. Exprimer $(s',t') = \mathsf{OAEP}'(m \oplus \delta, r)$, pour $\delta \in \{0,1\}^{n-k_1}$, en fonction de

$$(s,t) = \mathsf{OAEP}'(m,r), \quad \Delta = \delta\|0^{k_1}, \quad \Gamma = H(s) \oplus H(s').$$

Q-7. En déduite que la construction OAEP' sur la fonction f_{pk} est malléable.

Q-8. Que dire de son niveau de sécurité IND-CCA2 souhaité ?

6. Dependent-RSA

6.1 Chiffrement RSA de base

Considérons le schéma de chiffrement suivant :
— la clé publique de chiffrement est définie par un module RSA $n = pq$, ainsi qu'un exposant e, premier avec $\varphi(n)$;
— la clé de déchiffrement consiste en l'exposant $d = e^{-1} \bmod \varphi(n)$.
— le chiffrement d'un message $m \in \mathbb{Z}_n^\star$ est $c = m^e \bmod n$;
— le déchiffrement d'un chiffré $c \in \mathbb{Z}_n^\star$ est $m = c^d \bmod n$.

Q-1. Décrire une attaque contre la sécurité sémantique de ce schéma, en précisant sa complexité algorithmique et son avantage.

Rappel : L'avantage $\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A})$ d'un attaquant $\mathcal{A} = (A_1, A_2)$ contre la sécurité sémantique d'un schéma de chiffrement $\pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ est

$$\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr_{b \xleftarrow{R} \{0,1\}} \left[\begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{G}(1^k); (m_0, m_1, s) \leftarrow A_1(\mathsf{pk}) \\ c = \mathcal{E}_{\mathsf{pk}}(m_b) : A_2(m_0, m_1, s, c) = b \end{array} \right] - 1.$$

6.2 Chiffrement D-RSA

On a alors proposé la variante suivante :
— la clé publique de chiffrement est définie par un module RSA $n = pq$, ainsi qu'un exposant e, premier avec $\varphi(n)$;
— la clé de déchiffrement consiste en l'exposant $d = e^{-1} \bmod \varphi(n)$.
— pour le chiffrement d'un message $m \in \mathbb{Z}_n$, on choisit un élément aléatoire r dans $\mathbb{Z}_n$, puis on calcule $a = r^e \bmod n$ ainsi que $b = (r+1)^e \times m \bmod n$. Le couple (a, b) constitue le chiffré.

Q-2. Décrire l'algorithme de déchiffrement, à partir du chiffré (a, b), du module public n et de la clé de déchiffrement d.

6.3 Sécurité prouvée

On définit le problème du *(Computational) Dependent-RSA* – C-DRSA :
étant donné (n, e) et $a = r^e \bmod n$, calculer $(r+1)^e \bmod n$.

Q-3. Montrer que, sous réserve que le problème C-DRSA soit difficile à résoudre, le schéma de chiffrement décrit ci-dessus est non-inversible (soit $\mathsf{OW} - \mathsf{CPA}$).

On définit le problème du *Decisional Dependent-RSA* – D-DRSA :
étant donné (n, e), $a = r^e \bmod n$ et $b = s^e \bmod n$, décider si $s = r + 1 \bmod n$.

Rappel : Un attaquant $\mathcal{A}$ est capable de décider (résoudre le problème D-DRSA) si $\mathsf{Adv}_{n,e}^{\mathsf{ddrsa}}(\mathcal{A})$ est non-négligeable, où

$$\begin{aligned} \mathsf{Adv}_{n,e}^{\mathsf{ddrsa}}(\mathcal{A}) \;=\; & \Pr_{r \in \mathbb{Z}_n} [\mathcal{A}(n, e, r^e \bmod n, (r+1)^e \bmod n) \to 1] \\ & - \Pr_{r,s \in \mathbb{Z}_n} [\mathcal{A}(n, e, r^e \bmod n, s^e \bmod n) \to 1]. \end{aligned}$$

Q-4. Montrer que, sous réserve que ce problème D-DRSA soit difficile à résoudre, le schéma de chiffrement décrit ci-dessus est sémantiquement sûr (soit $\mathsf{IND} - \mathsf{CPA}$).

Q-5. Préciser l'avantage maximum d'un attaquant contre la sécurité sémantique de ce schéma, à clé (n, e) fixée, par rapport à l'avantage maximal contre le problème D-DRSA (soit $\mathsf{Adv}_{n,e}^{\mathsf{ddrsa}}(t)$, l'avantage maximal qu'un attaquant peut obtenir en temps t).

6.4 Non-malléabilité

Malheureusement, on ne peut espérer prouver mieux en terme de sécurité :

Q-6. Décrire un attaquant contre la non-malléabilité de ce schéma (sans entrer dans les détails de la définition de la non-malléabilité, montrer comment de simples relations entre des clairs se transposent en des relations simples entre les chiffrés).

Q-7. Que dire de son niveau de sécurité IND-CCA2 ?

6.5 Amélioration

Pour renforcer le niveau de sécurité, on a alors ajouté un troisième champ $c = H(m, r)$ dans le chiffré : pour le chiffrement d'un message $m \in \mathbb{Z}_n$, on choisit un élément aléatoire r dans $\mathbb{Z}_n$, puis on calcule $a = r^e \bmod n$ ainsi que $b = (r+1)^e \times m \bmod n$, et $c = H(m, r)$. Le triplet (a, b, c) constitue le chiffré.

Q-8. Montrer que si on modélise H par un oracle aléatoire, il est aisé de simuler l'oracle de déchiffrement à partir des questions-réponses de cette fonction H.

Q-9. Que dire de son niveau de sécurité IND-CCA2 ?

7. Etude d'un schéma de signature

7.1 Signature RSA de base

Considérons le schéma de signature suivant :
- la clé publique pk de vérification est définie par un module RSA $n = pq$, ainsi qu'un exposant e, premier avec $\varphi(n)$;
- la clé de signature sk consiste en l'exposant $d = e^{-1} \bmod \varphi(n)$.
- la signature d'un message $m \in \mathbb{Z}_n^\star$ est $\sigma = m^d \bmod n$;
- la vérification d'un couple $(m, \sigma) \in \mathbb{Z}_n^\star \times \mathbb{Z}_n^\star$ consiste en le test $\sigma^e \stackrel{?}{=} m \bmod n$.

Q-1. Décrire une falsification existentielle (sans message connu) contre ce schéma, en précisant sa complexité algorithmique et son succès.

Rappel : Le succès $\mathsf{Succ}^{\mathsf{ef}}_{\Sigma}(\mathcal{A})$ d'un attaquant $\mathcal{A}$ pour produire une falsification existentielle contre un schéma de signature $\Sigma = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ est

$$\mathsf{Succ}^{\mathsf{ef}}_{\Sigma}(\mathcal{A}) = \Pr\left[\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m, \sigma) \leftarrow \mathcal{A}(\mathsf{pk}) : \mathcal{V}(\mathsf{pk}, m, \sigma) = 1\ \right].$$

7.2 Signature « Hash-and-Invert »

Une méthode classique, qui permet à la fois de contrer cette attaque et de signer efficacement de longs messages, consiste à effectuer l'opération ci-dessus, non plus sur le message m, mais sur un haché $H(m)$ de ce dernier. Cette construction admet une preuve de sécurité, mais en faisant l'hypothèse que H ressemble à une fonction parfaitement aléatoire (*i.e.* dans le modèle de l'oracle aléatoire).

Gennaro, Halevi et Rabin ont proposé une construction différente, dont la sécurité peut être prouvée dans le modèle standard, mais sur une variante plus faible du problème RSA :
- pour le paramètre de sécurité k, on utilisera la fonction commune à toutes les clés publiques f_k. Sa définition et son utilisation seront précisées ultérieurement. L'algorithme de génération de clés produit deux grands nombres premiers p et q, sur k bits ($2^{k-1} < p, q < 2^k$), et calcule $n = pq$. La clé publique pk de vérification est définie par ce module RSA $n = pq$, ainsi que par un élément aléatoire $y \in \mathbb{Z}_n^\star$; La clé de signature sk consiste en la factorisation de n ou, de façon équivalente, $\varphi(n)$.
- la signature d'un message m est la racine e-ième de y, pour e dépendant de m, en fonction de f_k : $e = f_k(m)$, alors on calcule $d = e^{-1} = (f_k(m))^{-1} \bmod \varphi(n)$, puis $\sigma = y^d \bmod n$;
- la vérification d'un couple $(m, \sigma) \in \{0,1\}^\star \times \mathbb{Z}_n^\star$ consiste en le test $\sigma^{f_k(m)} \stackrel{?}{=} y \bmod n$.

Q-2. Montrer que f_k ne peut pas être une fonction quelconque (il suffira d'exhiber des exemples de fonctions avec les attaques associées).

7.3 Sécurité prouvée

On restreint f_k à retourner des valeurs entières strictement supérieures à 1. Puis on définit le problème du *RSA flexible* – $\mathsf{Fl\text{-}RSA}$: étant donnés un module RSA n, ainsi qu'un élément $y \in \mathbb{Z}_n^\star$, fournir un entier $e > 1$ et une racine e-ième de y modulo n.

$$\mathsf{Succ}^{\mathsf{fl-rsa}}(\mathcal{A}) = \Pr\left[\ (n, y) \leftarrow \mathcal{K}(1^k), (e, x) \leftarrow \mathcal{A}(n, y) : (x^e = y \bmod n) \wedge (e > 1)\ \right].$$

Q-3. Exhiber une réduction du cassage de ce problème à une falsification existentielle (sans message connu), et ce, indépendamment du choix de la fonction f_k à valeurs dans $\mathbb{N}^\star$.

7.4 Simulation des signatures

Cette sécurité est malheureusement insuffisante. Elle garantit l'infalsifiabilité, mais à condition de ne révéler aucune signature. Dans un deuxième temps, nous allons étudier les attaques à messages connus : on fournit à l'attaquant la clé publique de vérification, ainsi qu'une liste de ℓ couples message-signature (m_i, σ_i).

Q-4. Supposons des exposants impairs fixés $e_1, \ldots, e_\ell < 2^{k-2}$. Montrer que, à partir d'une instance (n, y) du *problème du RSA Flexible*, on peut générer une nouvelle instance (n, z) telle que
- z soit uniformément distribué dans $\mathbb{Z}_n^\star$, si y est initialement uniformément distribué dans $\mathbb{Z}_n^\star$;
- on connaisse les ℓ racines e_i-ièmes de z modulo n ;
- une nouvelle racine e-ième de z, pour e premier avec le produit E des e_i, nous permette de résoudre l'instance (n, y).

Remarque : On supposera que $n = pq$ est un premier « fort », c'est-à-dire que $p = 2p' + 1$ et $q = 2q' + 1$, avec p' et q' tous les deux premiers, sur $k - 1$ bits. Ainsi, $\varphi(n) = 4p'q' : 2$ est le seul diviseur premier plus petit que 2^{k-2}.

7.5 Hypothèse d'indivisibilité

On restreint désormais f_k à retourner des entiers impairs entre 2 et 2^{k-2} (où k est la taille en bits des facteurs premiers p et q, supposés « forts »), puis à satisfaire la propriété calculatoire suivante de **ℓ-indivisibilité** : étant donnés $x_1, \ldots, x_\ell$, il est difficile de trouver x tel que $f_k(x)$ divise le produit des $f_k(x_i)$.

Q-5. Montrer que l'implémentation de cette signature avec une telle fonction *ℓ-indivisible* résiste aux falsifications existentielles, même selon des attaques à ℓ messages connus, sous l'hypothèse que le problème du RSA flexible est difficile.

7.6 Attaques à messages choisis : fonctions caméléons

Il existe ensuite une conversion générique, pour faire passer le niveau de sécurité de la résistance aux attaques à messages connus à la résistance aux attaques à messages choisis adaptatives (qui donnent accès à l'attaquant à un oracle de signature), en utilisant une fonction caméléon : une fonction $h(m, r)$ qui résiste aux collisions, mais dont une trappe permet de trouver une seconde pré-image (m', r') avec la valeur partielle m' choisie.

Soit $h_N : \{0,1\}^\kappa \times \mathbb{Z}_N^\star \to \mathbb{Z}_N^\star$, $h_N(m, r) = y^m r^E \bmod N$, pour un module RSA N, et E un entier premier supérieur à 2^κ, la trappe étant $D = E^{-1} \bmod \varphi(N)$.

Q-6. Montrer que cette fonction h_N est une fonction caméléon :
- trouver une collision (sans la trappe) permet de casser un problème difficile, que l'on précisera ;
- la trappe permet de trouver, pour (m, r) et m' fixés, r' tel que $h_N(m, r) = h_N(m', r')$.

Q-7. Montrer comment on peut en déduire un schéma de signature (probabiliste) sûr contre les attaques à messages choisis adaptatives. Préciser les hypothèses algorithmiques nécessaires pour ce niveau de sécurité.

8. Etude d'un schéma de chiffrement

8.1 Chiffrement Multiple

Soit $\mathcal{S} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ un schéma de chiffrement sémantiquement sûr contre les attaques à messages choisis (mais pas forcément contre les attaques à chiffrés choisis). On définit alors le schéma de chiffrement multiple $\mathcal{S}' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ de la façon suivante :

- l'algorithme de génération de clés $\mathcal{K}'$ exécute deux fois l'algorithme $\mathcal{K}$, et fournit ainsi deux paires de clés $(\mathsf{sk}_0, \mathsf{pk}_0)$ et $(\mathsf{sk}_1, \mathsf{pk}_1)$. La clé secrète est alors définie par $\mathsf{sk}' = (\mathsf{sk}_0, \mathsf{sk}_1)$, tandis que la clé publique est définie par $\mathsf{pk}' = (\mathsf{pk}_0, \mathsf{pk}_1)$.
- l'algorithme de chiffrement consiste à chiffrer deux fois le même message sous les deux clés pk_0 et pk_1. Ainsi, pour chiffrer le message m, on choisit deux aléas r_0 et r_1, puis on calcule $c_0 = \mathcal{E}_{\mathsf{pk}_0}(m, r_0)$ et $c_1 = \mathcal{E}_{\mathsf{pk}_1}(m, r_1)$, enfin on retourne $c = (c_0, c_1)$.
- l'algorithme de déchiffrement, sur $c = (c_0, c_1)$, déchiffre les deux sous-chiffrés c_0 et c_1, puis compare les clairs. Plus concrètement, on obtient $m_0 = \mathcal{D}_{\mathsf{sk}_0}(c_0)$ et $m_1 = \mathcal{D}_{\mathsf{sk}_1}(c_1)$ (l'un d'eux peut être égal à $\perp$ en cas d'invalidité). Si $m_0 = m_1$, alors $m = m_0$, sinon $m = \perp$ (qui signifie « chiffré invalide ». Puis on retourne m.

Jeu 0. Le jeu d'attaque contre la **sécurité sémantique du schéma** $\mathcal{S}'$ est le suivant :

- on génère une clé publique aléatoire $\mathsf{pk}' = (\mathsf{pk}_0, \mathsf{pk}_1)$, que l'on transmet à l'attaquant $\mathcal{A}$;
- l'attaquant $\mathcal{A}$ choisit deux messages m^0 et m^1 ;
- on choisit un bit b aléatoire, on chiffre m^b en

$$c = \mathcal{E}'_{\mathsf{pk}'}(m^b, r' = (r_0, r_1)) = (c_0 = \mathcal{E}_{\mathsf{pk}_0}(m^b, r_0), c_1 = \mathcal{E}_{\mathsf{pk}_1}(m^b, r_1)) \ ;$$

- l'attaquant $\mathcal{A}$ retourne son choix b' pour b.

Son avantage utilise l'événement $\mathsf{S} = (b' = b)$, et est défini par

$$\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}'}(\mathcal{A}) = 2\Pr[\mathsf{S}_0] - 1.$$

Jeu 1 On considère désormais le jeu modifié suivant, avec un attaquant $\mathcal{A}$ contre $\mathcal{S}'$, où intervient un bit aléatoire u qui définit sur quelle sous-clé on applique le jeu de la **sécurité sémantique du schéma de base** $\mathcal{S}$:

- on fait générer pk_u, puis on génère le couple $(\mathsf{pk}_{1-u}, \mathsf{sk}_{1-u})$. On définit alors $\mathsf{pk}' = (\mathsf{pk}_0, \mathsf{pk}_1)$, que l'on transmet à l'attaquant $\mathcal{A}$;
- l'attaquant $\mathcal{A}$ choisit deux messages m^0 et m^1 ;
- on demande le chiffrement c_u de m_b par pk_u (sans connaître b), on choisit d aléatoire (en espérant que $d = b$), et on génère le *chiffré*

$$c = (c_0, c_1) \qquad \text{où} \qquad c_{1-u} = \mathcal{E}_{\mathsf{pk}_{1-u}}(m^d, r_{1-u}) \ ;$$

- l'attaquant $\mathcal{A}$ retourne son choix b' pour b, que l'on transmet.

Q-1. Donner une relation entre $\Pr[\mathsf{S}_1]$ et $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}}(t + T_{\mathcal{K}} + T_{\mathcal{E}})$, (où $T_{\mathcal{K}}$ est le temps de la génération des clés $\mathcal{K}$, et $T_{\mathcal{E}}$ est le temps pour évaluer un chiffrement $\mathcal{E}$).

Rappel : On définit par $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}}(t)$ l'avantage $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}}(\mathcal{A})$ maximal sur tous les attaquants $\mathcal{A}$ en temps t.

Q-2. Que peut-on dire du Jeu 1, dans le cas où $d = b$ (et u aléatoire), et donc de la probabilité de l'événement S_1 conditionné à $b = d$?

Q-3. Que peut-on dire du Jeu 1, dans le cas où $d \neq b$ (et u aléatoire). En constatant qu'aucune information sur u n'est révélée, évaluer la probabilité de l'événement S_1 conditionné à $b \neq d$?

Q-4. En déduire une borne sur $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}'}(t)$, en fonction de $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{S}}(t + T_{\mathcal{K}} + T_{\mathcal{E}})$.

8.2 Chiffrement El Gamal

On va alors se concentrer sur le chiffrement El Gamal comme instantiation du schéma $\mathcal{S}$.

— Données communes : un groupe cyclique $G = \langle g \rangle$ d'ordre premier q. L'élément g est le générateur de référence.
— $\mathcal{K}$ choisit sk aléatoirement dans $\mathbb{Z}_q$, et calcule $\mathsf{pk} = g^{\mathsf{sk}}$.
— Pour chiffrer un message $m \in G$, avec un aléa $r \in \mathbb{Z}_q$, on calcule $c = (\alpha = g^r, \beta = \mathsf{pk}^r \times m)$.
— Le déchiffrement de $c = (\alpha, \beta)$ se fait en calculant $m = \beta/\alpha^{\mathsf{sk}}$.

Q-5. Montrer la sécurité sémantique de ce schéma sous l'hypothèse Diffie-Hellman décisionnelle qu'on rappellera.

8.3 Preuves d'égalité des chiffrés

Notre objectif sera de déchiffrer le chiffrement « Double El Gamal » (chiffrement multiple $\mathcal{S}'$ ci-dessus avec le chiffrement El Gamal comme schéma de base $\mathcal{S}$) avec une seule clé de déchiffrement (sk_0 ou sk_1), choisie au moment de la génération des clés. Dans ce cas, il faut empêcher un attaquant de générer un chiffré non-valide (avec deux clairs différents). Il suffit pour cela de lui faire ajouter une preuve que les deux sous-chiffrés contiennent le même clair, tout en ne révélant rien sur ce clair.

Considérons le protocole suivant au sujet du chiffré $c = (c_0, c_1)$ où

$$c_0 = (\alpha_0 = g^{r_0}, \beta_0 = \mathsf{pk}_0^{r_0} \times m_0) \qquad c_1 = (\alpha_1 = g^{r_1}, \beta_1 = \mathsf{pk}_1^{r_1} \times m_1)$$

— on choisit $a, b \in \mathbb{Z}_q$, et on calcule $A_0 = g^{a_0}$, $A_1 = g^{a_1}$ et $B = \mathsf{pk}_0^{a_0}/\mathsf{pk}_1^{a_1}$;
— étant donné un challenge $e \in \mathbb{Z}_q$, on calcule $b_0 = a_0 - r_0 e \bmod q$ et $b_1 = a_1 - r_1 e \bmod q$, que l'on retourne.

Q-6. Montrer que la vérification de

$$A_0 = g^{b_0}\alpha_0^e \qquad A_1 = g^{b_1}\alpha_1^e \qquad B = \mathsf{pk}_1^{b_1}/\mathsf{pk}_0^{b_0} \times (\beta_1/\beta_0)^e$$

conduit à une preuve d'égalité des clairs dans les chiffrés c_0 et c_1 qui est « complète » (*complete*) et « consistante » (*sound*).

Q-7. Montrer que ce protocole est « zero-knowledge » face à un vérifieur honnête (le challenge e est choisi aléatoirement).

Q-8. Décrire la preuve non-interactive d'égalité des clairs, dans le modèle de l'oracle aléatoire.

Q-9. Décrire le schéma de chiffrement « Double El Gamal », qui retourne un chiffré sous forme de triplet (c_0, c_1, c_2), où (c_0, c_1) est le chiffrement multiple ci-dessus à base de El Gamal, et c_2 est la preuve non-interactive ci-dessus.

8.4 Sécurité contre les attaques à chiffrés choisis

On reprend le jeu 1 décrit ci-dessus, où $\mathcal{S}$ est le chiffrement El Gamal et $\mathcal{S}'$ est le « Double El Gamal », au cours duquel le simulateur connaît sk_{1-u}. On utilise l'attaquant $\mathcal{A}$ **à chiffrés choisis** contre la sécurité sémantique de $\mathcal{S}'$ pour casser la sécurité sémantique de $\mathcal{S}$ sur la clé pk_u, **sans chiffrés choisis** :

— on fait générer pk_u, puis on génère le couple $(\mathsf{pk}_{1-u}, \mathsf{sk}_{1-u})$. On définit alors $\mathsf{pk}' = (\mathsf{pk}_0, \mathsf{pk}_1)$, que l'on transmet à l'attaquant $\mathcal{A}$;

— l'attaquant $\mathcal{A}$ choisit deux messages m^0 et m^1 ;
— on demande le chiffrement c_u de m_b par pk_u (sans connaître b), on choisit d aléatoire (en espérant que $d = b$), et on génère le *chiffré*

$$c = (c_0, c_1, c_2) \qquad \text{où} \qquad c_{1-u} = \mathcal{E}_{\mathsf{pk}_{1-u}}(m^d, r_{1-u}) \ ;$$

— l'attaquant $\mathcal{A}$ retourne son choix b' pour b, que l'on transmet.

Q-10. Expliquer comment on génère c_2 dans notre simulation.

Q-11. Montrer comment on peut simuler l'oracle de déchiffrement $\mathcal{D}'_{\mathsf{sk}'}$. Sans exhiber la probabilité d'erreur, expliquer dans quels cas la simulation peut différer du déchiffrement.

8.5 Déchiffrement distribué

Q-12. Expliquer comment on peut distribuer le déchiffrement de ce « Double El Gamal » parmi deux autorités.

Q-13. Expliquer pourquoi il est toujours sûr contre les attaques à chiffrés choisis, pour un attaquant qui serait l'une des deux autorités de déchiffrement, contrairement à une variante du El Gamal de base qui inclurait de la redondance dans le clair (à vérifier après un déchiffrement préliminaire, avant de retourner le clair).

9. Etude d'un schéma de chiffrement

9.1 Les courbes elliptiques et les couplages

Sur certaines courbes elliptiques (dont on notera $(\mathbb{G}, +)$ un sous-groupe d'ordre premier q, engendré par P), il existe une application $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, où $\mathbb{G}_T$ est un sous-groupe du groupe multiplicatif (d'ordre q) d'une extension finie d'un corps fini, telle que :

— e est efficacement calculable (en temps t_e) ;

— e est bilinéaire dans $(\mathbb{G}_T, \times)$, ce qui signifie que pour tous $P, Q \in \mathbb{G}$ et $a, b \in \mathbb{Z}_q$, alors $e(a \cdot P, b \cdot Q) = e(P, Q)^{ab} \in \mathbb{G}_T$.

— e est non-dégénérée, ce qui signifie que $e(P, P)$ est un générateur de $\mathbb{G}_T$.

On peut alors définir les problèmes Diffie-Hellman usuels :

— le problème Diffie-Hellman calculatoire dans le groupe $\mathbb{G} = \langle P \rangle$ —noté $\mathsf{CDH}_{\mathbb{G},P}$— qui consiste à calculer $ab \cdot P$, à partir de $P_a = a \cdot P, P_b = b \cdot P \in \mathbb{G}$. On mesure le succès d'un attaquant $\mathcal{A}$ par

$$\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},P}(\mathcal{A}) = \Pr_{a,b \in \mathbb{Z}_q}[\mathcal{A}(a \cdot P, b \cdot P) = ab \cdot P].$$

— le problème Diffie-Hellman décisionnel dans le groupe $\mathbb{G} = \langle P \rangle$ —noté $\mathsf{DDH}_{\mathbb{G},P}$— qui consiste à décider si un élément Q donné est le $\mathsf{CDH}_{\mathbb{G},P}$ de (P_a, P_b), ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G},P}(\mathcal{A}) = \left| \Pr_{a,b \in \mathbb{Z}_q}[\mathcal{A}(a \cdot P, b \cdot P, ab \cdot P) = 1] - \Pr_{a,b,c \in \mathbb{Z}_q}[\mathcal{A}(a \cdot P, b \cdot P, c \cdot P) = 1] \right|.$$

Pour toutes les mesures de succès ou d'avantages ci-dessus (et à venir), on définit par $\mathsf{Succ}(t)$ ou $\mathsf{Adv}(t)$, les valeurs maximales atteintes pour des attaquants qui fonctionnent en temps borné par t.

Q-1. Il existe de tels contextes $(\mathbb{G} = \langle P \rangle, \mathbb{G}_T, e)$ pour lesquels on peut admettre la difficulté du problème Diffie-Hellman calculatoire sur $\mathbb{G}$ (pour tout temps t raisonnable, $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},P}(t)$ est très petit). Montrer cependant qu'il n'est pas raisonnable de supposer la difficulté du problème Diffie-Hellman décisionnel sur $\mathbb{G}$.

On définit alors les problèmes décisionnels suivants :

— le problème Diffie-Hellman décisionnel bilinéaire dans $\mathbb{G} = \langle P \rangle$ —noté $\mathsf{DBDH}_{\mathbb{G},P}$— qui consiste à décider, étant donné $(P_a = a \cdot P, P_b = b \cdot P, P_c = c \cdot P) \in \mathbb{G}^3$ et $Q \in \mathbb{G}$, si $Q = abc \cdot P$ ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},P}(\mathcal{A}) = \left| \begin{array}{l} \Pr_{a,b,c \in \mathbb{Z}_q}[\mathcal{A}(a \cdot P, b \cdot P, c \cdot P, abc \cdot P) = 1] \\ \quad - \Pr_{a,b,c,d \in \mathbb{Z}_q}[\mathcal{A}(a \cdot P, b \cdot P, c \cdot P, d \cdot P) = 1] \end{array} \right|.$$

— le problème Diffie-Hellman décisionnel bilinéaire dans $\mathbb{G} = \langle P \rangle$ et $\mathbb{G}_T$ —noté $\mathsf{DBDH}_{\mathbb{G},\mathbb{G}_T,P}$— qui consiste à décider, étant donné $(P_a = a \cdot P, P_b = b \cdot P, P_c = c \cdot P) \in \mathbb{G}^3$ et $\alpha \in \mathbb{G}_T$, si $\alpha = e(P, P)^{abc}$ ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},\mathbb{G}_T,P}(\mathcal{A}) = \left| \begin{array}{l} \Pr[\mathcal{A}(a \cdot P, b \cdot P, c \cdot P, e(P, P)^{abc}) = 1] \\ \quad - \Pr[\mathcal{A}(a \cdot P, b \cdot P, c \cdot P, e(P, P)^{d}) = 1] \end{array} \right|.$$

Q-2. Donner une relation entre $\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},P}(t)$ et $\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},\mathbb{G}_T,P}(t')$, où t et t' sont clairement reliés (notamment avec t_e, le temps de calcul pour évaluer un couplage e).

9.2 Un schéma de signature

Considérons le schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ suivant, dans un contexte $(\mathbb{G}, \mathbb{G}_T, e)$ comme défini ci-dessus, avec P un générateur de $\mathbb{G}$, d'ordre premier q. On a également une fonction de hachage $H : \{0,1\}^\star \to \mathbb{G}$.

— $\mathcal{K}(1^k)$: la clé publique pk de vérification est définie par un point $X = x \cdot P$; la clé de signature sk consiste en le scalaire $x \in \mathbb{Z}_q$;
— $\mathcal{S}(\mathsf{sk}, m)$: la signature d'un message $m \in \{0,1\}^\star$ est $\sigma = x \cdot H(m) \in \mathbb{G}$;
— $\mathcal{V}(\mathsf{pk}, m, \sigma)$: la vérification d'un couple $(m, \sigma) \in \{0,1\}^\star \times \mathbb{G}$ consiste en le test

$$e(P, \sigma) \stackrel{?}{=} e(X, H(m)).$$

Q-3. Montrer que ce schéma est correct (une signature correctement fabriquée sera acceptée).

Q-4. Montrer que, dans le modèle de l'oracle aléatoire (H est supposée être une fonction parfaitement aléatoire sur $\mathbb{G}$), ce schéma est infalsifiable à partir de la clé publique (niveau de sécurité $\mathsf{EF} - \mathsf{NMA}$ – pour *Existential Unforgeability under No Message Attacks*).
— On précisera l'hypothèse algorithmique nécessaire.
— On présentera une réduction, puis on conclura avec une borne sur $\mathsf{Succ}^{\mathsf{ef-nma}}_{\mathsf{Sign}}(t)$.
On pourra utiliser t_m, le temps d'une multiplication d'un point de $\mathbb{G}$ par un scalaire dans cette relation.

Rappel : Le succès $\mathsf{Succ}^{\mathsf{ef-nma}}_{\mathsf{Sign}}(\mathcal{A})$ d'un attaquant $\mathcal{A}$ pour produire une falsification existentielle contre un schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ est

$$\mathsf{Succ}^{\mathsf{ef-nma}}_{\mathsf{Sign}}(\mathcal{A}) = \Pr\left[\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m, \sigma) \leftarrow \mathcal{A}(\mathsf{pk}) : \mathcal{V}(\mathsf{pk}, m, \sigma) = 1\ \right].$$

Q-5. Montrer que, dans le modèle de l'oracle aléatoire, ce schéma est infalsifiable, même selon des attaques à messages choisis (niveau de sécurité $\mathsf{EF} - \mathsf{CMA}$).
— On précisera l'hypothèse algorithmique nécessaire.
— On présentera une réduction, puis on conclura avec une borne sur $\mathsf{Succ}^{\mathsf{ef-cma}}_{\mathsf{Sign}}(t)$.

Rappel : Le succès $\mathsf{Succ}^{\mathsf{ef-cma}}_{\mathsf{Sign}}(\mathcal{A})$ d'un attaquant $\mathcal{A}$ pour produire une falsification existentielle selon une attaque à messages choisis contre un schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ est

$$\mathsf{Succ}^{\mathsf{ef-cma}}_{\mathsf{Sign}}(\mathcal{A}) = \Pr\left[\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m, \sigma) \leftarrow \mathcal{A}^{\mathcal{S}}(\mathsf{pk}) : \mathcal{V}(\mathsf{pk}, m, \sigma) = 1\ \right].$$

On remarque que pendant l'attaque, $\mathcal{A}$ a accès à l'oracle de signature $\mathcal{S}$. Mais bien évidemment, la falsification doit être sur un nouveau message m, non demandé à $\mathcal{S}$.

Q-6. Sachant que les points du groupe $\mathbb{G}$ peuvent être codés sur moins de 200 bits, commenter les propriétés de ce schéma de signature.

9.3 Chiffrement basé sur l'identité

L'avantage du chiffrement asymétrique, par rapport au chiffrement symétrique, est la possibilité d'envoyer un message chiffré à un interlocuteur avec qui l'on n'a jamais mis de secret en commun : seule sa clé publique est nécessaire. Néanmoins, cette clé publique doit être authentique, et donc, une autorité de certification est nécessaire.

Le chiffrement basé sur l'idendité permet de se passer de cette autorité de certification, puisque la clé publique de chacun est sa propre identité (ou son adresse e-mail), et une autorité se charge de distribuer les clés de déchiffrement associées.

Considérons donc le schéma de chiffrement basé sur l'identité (*Identity-Based Encryption*) $\mathsf{IBE} = (\mathcal{K}, \mathsf{Extract}, \mathcal{E}, \mathcal{D})$ suivant :

- Données communes : le contexte ci-dessus, $(\mathbb{G} = \langle P \rangle, \mathbb{G}_T, e)$, où P est un générateur d'ordre premier q. Nous avons également une fonction de hachage $H : \{0,1\}^\star \to \mathbb{G}$.
- $\mathcal{K}(1^k)$: génération des clés de l'autorité, qui consistent en la clé secrète maître, un scalaire $s \in \mathbb{Z}_q$, et la clé publique maître, $Q = s \cdot P$. La clé Q est connue de tous.
- $\mathsf{Extract}(s, \mathsf{ID})$: l'algorithme d'extraction fournit, à l'aide de la clé secrète maître s, la clé de déchiffrement pour l'utilisateur d'identité ID. Il s'agit de $\mathsf{sk} = s \cdot H(\mathsf{ID})$.
- $\mathcal{E}(\mathsf{ID}, m)$: pour chiffrer un message $m \in \mathbb{G}_T$ à un individu ID, on génère un aléa $r \in \mathbb{Z}_q$, puis le chiffré consiste en le couple $c = (R = r \cdot P, K \times m)$ où $K = e(H(\mathsf{ID}), Q)^r$.
- $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer un couple $c = (R, C)$, on calcule $K = e(\mathsf{sk}, R)$, puis on démasque $m = C/K$.

Q-7. Montrer qu'il est difficile, dans le modèle de l'oracle aléatoire, de générer une clé sk valide pour une nouvelle identité, sans l'aide de l'autorité, même après plusieurs requêtes d'extraction.

Q-8. Montrer que ce schéma est correct (un chiffré sera convenablement déchiffré).

Q-9. Montrer que ce schéma satisfait le niveau de sécurité $\mathsf{IND} - \mathsf{CPA}$, sous une hypothèse bien précise. On présentera la réduction.

Note : L'avantage $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathsf{PKE}}(\mathcal{A})$ d'un attaquant $\mathcal{A}$ contre l'indistingabilité d'un schéma de chiffrement $\mathsf{PKE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ est

$$\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathsf{PKE}}(\mathcal{A}) = 2 \times \Pr[(s, Q) \leftarrow \mathcal{K}(1^k), (\mathsf{ID}, m_0, m_1) \leftarrow \mathcal{A}_1(Q), c = \mathcal{E}(\mathsf{ID}, m_b) : \mathcal{A}_2(c) = b] - 1.$$

On remarque que la définition est ici classique, puisque l'oracle $\mathsf{Extract}$ n'apparaît pas.

Dans un tel environnement, basé sur l'identité, la notion $\mathsf{IND} - \mathsf{CPA}$ est faible, puisque l'attaquant n'a pas accès à l'oracle $\mathsf{Extract}$.

Q-10. Si l'on donne accès à l'attaquant à l'oracle $\mathsf{Extract}$, quelles sont les contraintes à imposer quant à ces requêtes pour que le niveau de sécurité soit accessible ?

Q-11. Montrer que ce schéma satisfait ce niveau de sécurité $\mathsf{IND} - \mathsf{ID\text{-}CPA}$ pour « Indistingabilité selon des attaques à clairs et identités choisis » (chiffrement des messages de son choix, et extraction des clés secrètes pour les identités de son choix).

- On précisera l'hypothèse algorithmique nécessaire.
- On présentera une réduction, puis on conclura avec une borne sur $\mathsf{Succ}^{\mathsf{ind-id-cpa}}_{\mathsf{IBE}}(t)$.

Q-12. Montrer qu'en masquant avec $H'(K)$, au lieu de K, on peut reposer sur une hypothèse algorithmique plus faible, dans le modèle de l'oracle aléatoire.

10. Le chiffrement asymétrique anonyme

10.1 Définitions

Un schéma de chiffrement à clé publique est défini par 3 algorithmes :
— $\mathcal{K}(1^k)$, qui génère un couple de clés $(\mathsf{pk}, \mathsf{sk})$, de chiffrement et de déchiffrement, en fonction du paramètre de sécurité k ;
— $\mathcal{E}(\mathsf{pk}; m)$, qui produit un chiffré c de m sous la clé pk (avec un ruban aléatoire) ;
— $\mathcal{D}(\mathsf{sk}; c)$, qui déchiffre c sous sk, et retourne donc, soit le clair m s'il existe, soit $\perp$ si le chiffré n'est pas correct.

La notion de sécurité pour le chiffrement est la *confidentialité* du clair, malgré la vue du chiffré, pour toute clé publique : aucun attaquant n'est en mesure d'avoir un avantage non-négligeable dans le jeu suivant, contre le challenger
— le challenger génère une liste de couples $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathcal{K}(1^k)$, et fournit les clés publiques pk_i à l'attaquant ;
— l'attaquant choisit une clé publique $\mathsf{pk} \in \{\mathsf{pk}_i\}$ et deux messages (m_0, m_1), qu'il envoie au challenger ;
— le challenger choisit un bit $b \leftarrow \{0, 1\}$, et chiffre m_b sous pk dans $c \leftarrow \mathcal{E}(\mathsf{pk}; m_b)$, qu'il envoie à l'attaquant ;
— l'attaquant retourne son avis b' au sujet de b.

On mesure l'avantage d'un attaquant $\mathcal{A}$, contre la *sécurité sémantique* définie par le jeu ci-dessus, par

$$\mathsf{Adv}(\mathcal{A}) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] = 2 \times \Pr[b' = b] - 1.$$

Q-1. Soit $\mathcal{B}$ un algorithme capable d'extraire le bit de poids faible de $m \in \{0, 1\}^n$ à partir de $c = \mathcal{E}(\mathsf{pk}; m)$ avec probabilité $1/2 + \varepsilon$:

$$\Pr[\mathcal{B}(\mathsf{pk}, c) = b | b \xleftarrow{R} \{0, 1\}, m \xleftarrow{R} \{0, 1\}^{n-1} \times \{b\}, c \leftarrow \mathcal{E}(\mathsf{pk}, m)] \geq \frac{1}{2} + \varepsilon.$$

Montrer comment on peut construire un attaquant $\mathcal{A}$ contre la sécurité sémantique, et exprimer l'avantage.

Masquer le contenu d'un message est important, mais parfois l'identité du destinataire est aussi une information sensible. Supposons donc qu'un certain nombre de clés publiques sont possibles, et on souhaite garantir l'*anonymat* du destinataire : aucune information sur la clé publique du destinataire ne doit fuir dans le chiffré (ce qui permettrait de lever un doute sur la clé publique utilisée, et ainsi connaître le destinataire).

Q-2. Définir le jeu d'un attaquant contre un challenger pour spécifier une telle notion d'anonymat.

10.2 Chiffrement ElGamal

On rappelle le chiffrement ElGamal, avec g un générateur de $\mathbb{G}$, d'ordre premier q :
— $\mathcal{K}(1^k)$: la clé privée est un scalaire $\mathsf{sk} = x \in \mathbb{Z}_q^\star$, et la clé publique est $\mathsf{pk} = y = g^x$;
— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \mathbb{G}$, on choisit $r \in \mathbb{Z}^\star$, puis on calcule

$$c_1 = g^r, \quad c_2 = y^r \cdot m.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule $m = c_2 / c_1^x$.

Q-3. Montrer que ce schéma garantit la sécurité sémantique (le jeu ci-dessus, sans aucun oracle de déchiffrement). On précisera sous quelle hypothèse algorithmique.

Q-4. Montrer qu'il garantit également l'anonymat du destinataire (votre notion de sécurité). On précisera sous quelle hypothèse algorithmique.

Q-5. En revanche, si l'adversaire a accès aux oracles de déchiffrement, montrer comment casser la sécurité sémantique.

Q-6. Montrer que de tels oracles permettent également de casser l'anonymat du destinataire.

10.3 Chiffrement RSA-OAEP

On rappelle le chiffrement RSA :

— $\mathcal{K}(1^k)$: la clé publique est un module RSA $n = pq$ sur k bits ($2^{k-1} < n < 2^k$) et l'exposant de chiffrement $\mathsf{pk} = (n, e)$, tandis que la clé privée est l'exposant de déchiffrement $\mathsf{sk} = d$;

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \mathbb{Z}_n^\star$, on calcule

$$c = m^e \bmod n.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule $m = c^d \bmod n$.

Q-7. Montrer que ce schéma ne garantit pas la sécurité sémantique.

On applique donc le padding OAEP (sans redondance) pour le renforcer, où $k = k_1 + k_2 + 1$ (ainsi, $n > 2^{k_1+k_2}$), avec $G : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$ et $H : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$:

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \{0,1\}^{k_1}$, on choisit un aléa $r \in \{0,1\}^{k_2}$, on calcule

$$x = \mathsf{OAEP}(m, r) = s\|t \text{ où } s = G(r) \oplus m, t = H(s) \oplus r, \quad c = \mathsf{RSA}(n, e, x) = x^e \bmod n.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule

$$x = c^d \bmod n = s\|t, \quad r = t \oplus H(s), \quad m = s \oplus G(r).$$

Q-8. Montrer que cette construction RSA-OAEP ne garantit toujours pas l'anonymat.

On considère les 6 applications suivantes de $\{0,1\}^k$ dans $\{0,1\}^k$:

$f^1_{n,e}(x)$
  `if` $x < n$ `then` $u \leftarrow x^e \bmod n$
  `else` $u \leftarrow x$
  `return` u

$f^2_{n,e}(u)$
  `if` $u < 2^k - n$ `then` $v \leftarrow u + n$
  `elseif` $2^k - n \le u < n$ `then` $v \leftarrow u$
  `else` $v \leftarrow u - n$
  `return` v

$f^3_{n,e}(v)$
  `if` $v < n$ `then` $y \leftarrow v^e \bmod n$
  `else` $y \leftarrow v$
  `return` y

$g^1_{n,d}(u)$
  `if` $u < n$ `then` $x \leftarrow u^d \bmod n$
  `else` $x \leftarrow u$
  `return` x

$g^2_{n,d}(v)$
  `if` $v < 2^k - n$ `then` $u \leftarrow v + n$
  `if` $2^k - n \le v < n$ `then` $u \leftarrow v$
  `else` $u \leftarrow v - n$
  `return` u

$g^3_{n,d}(y)$
  `if` $y < n$ `then` $v \leftarrow y^d \bmod n$
  `else` $v \leftarrow y$
  `return` v

Q-9. Montrer qu'il s'agit de 6 bijections, et que pour $i = 1, 2, 3$, $f^i_{n,e}$ et $g^i_{n,d}$ sont réciproques l'une de l'autre..

On remplace $\mathsf{RSA}(n, e, x) = x^e \bmod n$ pour $x \in \mathbb{Z}_n^\star$, par $\mathsf{RSACD}(n, e, x) = f_{n,e}^3(f_{n,e}^2(f_{n,e}^1(x)))$ pour $x \in \{0,1\}^k$. On va alors montrer que l'inversion de RSACD est aussi difficile que l'inversion de RSA.

Pour cela, on procède en 3 temps, en utilisant un algorithme $\mathcal{A}$ contre $\mathsf{RSACD}(n, e, x) = f_{n,e}^3(v)$, où $v = f_{n,e}^2(u)$ et $u = f_{n,e}^1(x)$, qui retrouve x avec probabilité ε.

Q-10. Montrer que si l'algorithme $\mathcal{A}$ retrouve x à partir de $y = \mathsf{RSACD}(n, e, x)$, avec probabilité ε_1 lorsque v est uniformément distribué dans $]0, n[$, alors on peut construire un algorithme $\mathcal{B}_1$ qui inverse RSA avec probabilité ε_1.

Q-11. Montrer que si l'algorithme $\mathcal{A}$ retrouve x à partir de $y = \mathsf{RSACD}(n, e, x)$, avec probabilité ε_2 lorsque v est uniformément distribué dans $]2^k - n, 2^k[$, alors on peut construire un algorithme $\mathcal{B}_2$ qui inverse RSA avec probabilité ε_2.

Q-12. En déduire un algorithme qui utilise $\mathcal{A}$ pour inverser RSA, et préciser la probabilité de succès.

10.4 Chiffrement RSACD-OAEP

Considérons désormais la construction suivante, avec $k = k_1 + k_2$:

— $\mathcal{K}(1^k)$: la clé publique est un module RSA $n = pq$ sur k bits ($2^{k-1} < n < 2^k$) et l'exposant de chiffrement $\mathsf{pk} = (n, e)$, tandis que la clé privée est l'exposant de déchiffrement $\mathsf{sk} = d$;

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \{0,1\}^{k_1}$, on choisit un aléa $r \in \{0,1\}^{k_2}$, on calcule

$$x = \mathsf{OAEP}(m, r) \quad c = \mathsf{RSACD}(n, e, x).$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule

$$x = \mathsf{RSACD}^{-1}(n, d, c) \quad (m, r) = \mathsf{OAEP}^{-1}(x).$$

Q-13. Discuter la propriété d'anonymat de ce schéma.

11. Le chiffrement linéaire

11.1 Les courbes elliptiques et les couplages

Sur certaines courbes elliptiques (dont on notera $(\mathbb{G}, +)$ un sous-groupe d'ordre premier q, engendré par un point P), il existe une application $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, où $\mathbb{G}_T$ est un sous-groupe du groupe multiplicatif (d'ordre q) d'une extension finie d'un corps fini, telle que :

— e est efficacement calculable ;

— e est bilinéaire de $\mathbb{G} \times \mathbb{G}$ dans $\mathbb{G}_T$: pour tous $P, Q \in \mathbb{G}$ et $a, b \in \mathbb{Z}_q$, $e(a \cdot P, b \cdot Q) = e(P, Q)^{ab} \in \mathbb{G}_T$;

— e est non-dégénérée : $e(P, P) \neq 1$.

Note On utilisera donc une notation additive pour le groupe $\mathbb{G}$ et une notation multiplicative pour le groupe $\mathbb{G}_T$.

On peut définir les problèmes Diffie-Hellman usuels :

— le problème Diffie-Hellman calculatoire dans le groupe $\mathbb{G}$ —noté $\mathsf{CDH}_{\mathbb{G}}$— qui consiste à calculer $ab \cdot P$, à partir de $P \in \mathbb{G}$, $P_a = a \cdot P$, $P_b = b \cdot P$, pour $a, b \in \mathbb{Z}_q$. On mesure le succès d'un attaquant $\mathcal{A}$ par

$$\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G}}(\mathcal{A}) = \Pr_{\substack{P \in \mathbb{G} \\ a,b \in \mathbb{Z}_q}} [\mathcal{A}(P, a \cdot P, b \cdot P) = ab \cdot P].$$

— le problème Diffie-Hellman décisionnel dans le groupe $\mathbb{G}$ —noté $\mathsf{DDH}_{\mathbb{G}}$— qui consiste à décider si un élément $Q \in \mathbb{G}$ donné est le $\mathsf{CDH}_{\mathbb{G}}$ de (P, P_a, P_b), noté $\mathsf{CDH}_{\mathbb{G}}(P, P_a, P_b)$, ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G}}(\mathcal{A}) = \left| \Pr_{\substack{P \in \mathbb{G} \\ a,b \in \mathbb{Z}_q}} [\mathcal{A}(P, a \cdot P, b \cdot P, ab \cdot P) = 1] - \Pr_{\substack{P \in \mathbb{G} \\ a,b,c \in \mathbb{Z}_q}} [\mathcal{A}(P, a \cdot P, b \cdot P, c \cdot P) = 1] \right|.$$

Pour toutes les mesures de succès ou d'avantages ci-dessus (et à venir), on définit par $\mathsf{Succ}(t)$ ou $\mathsf{Adv}(t)$, les valeurs maximales atteintes pour des attaquants bornés en temps par t.

Q-1. Il existe de tels contextes $(\mathbb{G}, \mathbb{G}_T, e)$ pour lesquels on peut admettre la difficulté du problème Diffie-Hellman calculatoire sur $\mathbb{G}$ (pour tout temps t raisonnable, $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G}}(t)$ est très petit).
Montrer cependant qu'il n'est pas raisonnable de supposer la difficulté du problème Diffie-Hellman décisionnel sur $\mathbb{G}$: on exhibera un distingueur pour le $\mathsf{DDH}_{\mathbb{G}}$.

Outre les problèmes Diffie-Hellman bilinéaires, que nous n'utiliserons pas ici, on définit les problèmes Diffie-Hellman "linéaires" suivants :

— le problème Diffie-Hellman linéaire calculatoire dans $\mathbb{G}$ —noté $\mathsf{CLDH}_{\mathbb{G}}$— qui consiste à calculer, étant donné $(P, Q, R) \in \mathbb{G}^3$ et $(U = a \cdot P, V = b \cdot Q)$ pour $a, b \in \mathbb{Z}_q$, le résultat $W = (a + b) \cdot R$. On mesure le succès d'un attaquant $\mathcal{A}$ par

$$\mathsf{Succ}^{\mathsf{cldh}}_{\mathbb{G}}(\mathcal{A}) = \Pr_{\substack{P,Q,R \in \mathbb{G} \\ a,b \in \mathbb{Z}_q}} [\mathcal{A}(P, Q, R, a \cdot P, b \cdot Q) = (a + b) \cdot R].$$

— le problème Diffie-Hellman linéaire décisionnel dans $\mathbb{G}$ —noté $\mathsf{DLDH}_{\mathbb{G}}$— qui consiste à décider, étant donné $(P, Q, R) \in \mathbb{G}^3$, $(U = a \cdot P, V = b \cdot Q)$ pour $a, b \in \mathbb{Z}_q$ et $W \in \mathbb{G}$, si $W = (a + b) \cdot R$ ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{dldh}}_{\mathbb{G}}(\mathcal{A}) = \left| \begin{array}{r} \Pr_{\substack{P,Q,R \in \mathbb{G} \\ a,b \in \mathbb{Z}_q}} [\mathcal{A}(P, Q, R, a \cdot P, b \cdot Q, (a + b) \cdot R) = 1] \\ - \Pr_{\substack{P,Q,R \in \mathbb{G} \\ a,b,c \in \mathbb{Z}_q}} [\mathcal{A}(a \cdot P, b \cdot Q, c \cdot R) = 1] \end{array} \right|.$$

Q-2. Montrer comment résoudre le problème $\mathsf{CDH}_{\mathbb{G}}$ à partir d'un algorithme qui résoud le problème $\mathsf{CLDH}_{\mathbb{G}}$, en temps t avec probabilité ε.

Q-3. Montrer comment décider le problème $\mathsf{DDH}_{\mathbb{G}}$ à partir d'un algorithme qui décide le problème $\mathsf{DLDH}_{\mathbb{G}}$, en temps t avec avantage ε.

Q-4. Discuter la difficulté des problèmes $\mathsf{CLDH}_{\mathbb{G}}$ et $\mathsf{DLDH}_{\mathbb{G}}$ dans des groupes $\mathbb{G}$ équipés d'une application bilinéaire.

11.2 Un schéma de chiffrement

Considérons le schéma de chiffrement $\mathsf{Enc} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ suivant, dans un contexte $(\mathbb{G}, \mathbb{G}_T, e)$ comme défini ci-dessus, avec R un générateur de $\mathbb{G}$, d'ordre premier q.

— $\mathcal{K}(1^k)$: la clé privée est un couple $\mathsf{sk} = (x, y) \in (\mathbb{Z}_q^\star)^2$,
et la clé publique est $\mathsf{pk} = (P = x^{-1} \cdot R, Q = y^{-1} \cdot R)$;

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \mathbb{G}$, on choisit $a, b \in \mathbb{Z}^\star$, puis on calcule

$$c_1 = a \cdot P, \quad c_2 = b \cdot Q, \quad K = (a+b) \cdot R, \quad c_3 = K + m.$$

Q-5. Expliquer le déchiffrement à partir de $\mathsf{sk} = (x, y)$.

Q-6. Montrer que ce schéma atteint le niveau de sécurité $\mathsf{IND} - \mathsf{CPA}$. On précisera sous quelle hypothèse algorithmique.

Soit un chiffré $(c_1 = a \cdot P, c_2 = b \cdot Q, c_3 = (a+b) \cdot R + m)$. On définit $\pi = a \cdot R$.

Q-7. Montrer que (c_1, c_2, c_3) est un chiffré de 0 (soit donc $c_3 = (a+b) \cdot R$) si et seulement si

$$e(R, c_1) = e(P, \pi) \quad \text{et} \quad e(R, c_2) = e(c_3 - \pi, Q).$$

11.3 Mise en gage

Un protocole de mise en gage permet de s'engager sur une valeur (un bit β) lors de la phase d'engagement, $C = \mathsf{Commit}(\beta)$, sans ne rien révéler sur cette valeur, mais sans pouvoir la changer ultérieurement, lors de l'ouverture, $\mathsf{Open}(C, \beta)$.

Soit R un générateur de $\mathbb{G}$, d'ordre premier q. On considère l'algorithme de génération de paramètres $\mathsf{Gen}_1 \to (P, Q, R, U, V, W)$:

— on choisit aléatoirement $x, y \in \mathbb{Z}_q^\star$, et on définit $P = x^{-1} \cdot R$ et $Q = y^{-1} \cdot R$) ;

— on chiffre R sous $\mathsf{pk} = (P, Q)$:

$$U = a \cdot P, \quad V = b \cdot Q, \quad W = (a+b) \cdot R + R.$$

La mise en gage d'un bit β consiste en le triplet $\mathsf{Commit}(\beta; r, s) = (C_1, C_2, C_3)$

$$C_1 = \beta \cdot U + r \cdot P \quad C_2 = \beta \cdot V + s \cdot Q \quad C_3 = \beta \cdot W + (r+s) \cdot R.$$

L'ouverture consiste en la publication de (β, r, s).

Q-8. Montrer que ce schéma de mise en gage est *perfectly-binding* : une mise en gage ne peut être ouverte que d'une seule manière (même pour un attaquant tout-puissant).

Q-9. Montrer que ce schéma de mise en gage est *computationally-blinding* : la mise en gage ne révèle pas d'information sur le bit engagé, à un attaquant de puissance de calcul bornée (en revanche, un attaquant tout-puissant peut extraire de l'information).

Q-10. Montrer que la connaissance d'une trappe permet d'extraire efficacement β de (C_1, C_2, C_3).

On modifie maintenant l'algorithme d'initialisation en $\mathsf{Gen}_2 \to (P, Q, R, U, V, W)$: Soit R un générateur de $\mathbb{G}$, d'ordre premier q. On considère l'algorithme de génération de paramètres $\mathsf{Gen}_1 \to (P, Q, R, U, V, W)$:

— on choisit aléatoirement $x, y \in \mathbb{Z}_q^\star$, et on définit $P = x^{-1} \cdot R$ et $Q = y^{-1} \cdot R)$;

— on chiffre 0 (au lieu de R) sous $\mathsf{pk} = (P, Q)$:

$$U = a \cdot P, \quad V = b \cdot Q, \quad W = (a + b) \cdot R + R.$$

Q-11. Expliquer en quoi les deux processus Gen_1 et Gen_2 fournissent des paramètres (P, Q, R, U, V, W) indistingables.

Q-12. Montrer que ce schéma de mise en gage (avec Gen_2) est désormais *perfectly-blinding* : la mise en gage ne révèle aucune information sur le bit engagé (même à un attaquant tout-puissant).

Q-13. Montrer que ce schéma de mise en gage est désormais *computationally-binding* : une mise en gage ne peut être ouverte que d'une seule manière, à moins d'être en mesure de résoudre un problème difficile.

Soient les paramètres (P, Q, R, U, V, W), une mise en gage $(C_1, C_2, C_2) = \mathsf{Commit}(\beta; r, s)$.

Q-14. Montrer que, si (P, Q, R, U, V, W) provient de Gen_1, $\pi = r \cdot R$ est une preuve que $\beta = 0$. On précisera les tests à vérifier.

Q-15. Montrer que, si (P, Q, R, U, V, W) provient de Gen_2, la connaissance de $\mathsf{sk} = (x, y)$ permet de générer une preuve "acceptable" d'engagement de 0, pour toute mise en gage (C_1, C_2, C_3), et donc y compris pour une mise en gage de 1, sans connaître la valeur engagée ni les aléas utilisés.

12. Le chiffrement basé sur l'identité

12.1 Le logarithme discret et les couplages

Sur certaines structure algébriques (notamment courbes elliptiques), dont on notera $(\mathbb{G}, \times)$ un sous-groupe d'ordre premier q, engendré par g, il existe une application $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, où $\mathbb{G}_T$ est un sous-groupe du groupe multiplicatif (d'ordre q) d'une extension finie d'un corps fini, telle que :

— e est efficacement calculable (en temps t_e) ;
— e est bilinéaire dans $(\mathbb{G}_T, \times)$: pour tous $g, h \in \mathbb{G}$ et $a, b \in \mathbb{Z}_q$, $e(g^a, h^b) = e(g, h)^{ab} \in \mathbb{G}_T$.
— e est non-dégénérée : $e(g, g)$ est un générateur de $\mathbb{G}_T$.

Cette application est appelée "couplage" sur le groupe $\mathbb{G}$.

On peut alors définir les problèmes Diffie-Hellman usuels :

— le problème Diffie-Hellman calculatoire dans le groupe $\mathbb{G} = \langle g \rangle$ —noté $\mathsf{CDH}_{\mathbb{G},g}$— qui consiste à calculer g^{ab}, à partir de $A = g^a, B = g^b \in \mathbb{G}$. On mesure le succès d'un attaquant $\mathcal{A}$ par

$$\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},g}(\mathcal{A}) = \Pr_{a,b\in\mathbb{Z}_q}[\mathcal{A}(g^a, g^b) = g^{ab}].$$

— le problème Diffie-Hellman décisionnel dans le groupe $\mathbb{G} = \langle g \rangle$ —noté $\mathsf{DDH}_{\mathbb{G},g}$— qui consiste à décider si un élément C donné est le $\mathsf{CDH}_{\mathbb{G},g}$ de (A, B), ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{ddh}}_{\mathbb{G},g}(\mathcal{A}) = \left| \Pr_{a,b\in\mathbb{Z}_q}[\mathcal{A}(g^a, g^b, g^{ab}) = 1] - \Pr_{a,b,c\in\mathbb{Z}_q}[\mathcal{A}(g^a, g^b, g^c) = 1] \right|.$$

Pour toutes les mesures de succès ou d'avantages ci-dessus (et à venir), on définit par $\mathsf{Succ}(t)$ ou $\mathsf{Adv}(t)$, les valeurs maximales atteintes pour des attaquants en temps borné par t.

Q-1. Il existe de tels contextes $(\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e)$ pour lesquels on peut admettre la difficulté du problème Diffie-Hellman calculatoire sur $\mathbb{G}$ (pour tout temps t raisonnable, $\mathsf{Succ}^{\mathsf{cdh}}_{\mathbb{G},g}(t)$ est très petit). Montrer cependant qu'il n'est pas raisonnable de supposer la difficulté du problème Diffie-Hellman décisionnel sur $\mathbb{G}$.

On définit alors les problèmes décisionnels suivants :

— le problème Diffie-Hellman décisionnel bilinéaire dans $\mathbb{G} = \langle g \rangle$ —noté $\mathsf{DBDH}_{\mathbb{G},g}$— qui consiste à décider, étant donné $(A = g^a, B = g^b, C = g^c) \in \mathbb{G}^3$ et $D \in \mathbb{G}$, si $D = g^{abc}$ ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},g}(\mathcal{A}) = \left| \Pr_{a,b,c\in\mathbb{Z}_q}[\mathcal{A}(g^a, g^b, g^c, g^{abc}) = 1] - \Pr_{a,b,c,d\in\mathbb{Z}_q}[\mathcal{A}(g^a, g^b, g^c, g^d) = 1] \right|.$$

— le problème Diffie-Hellman décisionnel bilinéaire dans $\mathbb{G} = \langle g \rangle$ et $\mathbb{G}_T$ —noté $\mathsf{DBDH}_{\mathbb{G},\mathbb{G}_T,g}$— qui consiste à décider, étant donné $(A = g^a, B = g^b, C = g^c) \in \mathbb{G}^3$ et $\alpha \in \mathbb{G}_T$, si $\alpha = e(g, g)^{abc}$ ou non. On mesure l'avantage d'un distingueur $\mathcal{A}$ par

$$\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},\mathbb{G}_T,g}(\mathcal{A}) = \left| \Pr[\mathcal{A}(g^a, g^b, g^c, e(g, g)^{abc}) = 1] - \Pr[\mathcal{A}(g^a, g^b, g^c, e(g, g)^d) = 1] \right|.$$

Q-2. Donner une relation entre $\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},g}(t)$ et $\mathsf{Adv}^{\mathsf{dbdh}}_{\mathbb{G},\mathbb{G}_T,g}(t')$, où t et t' sont clairement reliés (notamment avec t_e, le temps de calcul pour évaluer un couplage e).

12.2 Un schéma de signature

Considérons le schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ suivant, dans un contexte $(\mathbb{G}, \mathbb{G}_T, e)$ comme défini ci-dessus, avec g, h et u, trois générateurs indépendants de $\mathbb{G}$, d'ordre premier q.

— $\mathcal{K}(1^k)$: la clé publique Spk de vérification est définie (en plus de (g, h, u)) par $G = g^x$; la clé de signature Ssk consiste en $H = h^x$;

— $\mathcal{S}(\mathsf{Ssk}, m)$: la signature d'un message $m \in \mathbb{Z}_q$ est $\sigma = (\sigma_1 = H \times F(m)^r, \sigma_2 = g^r) \in \mathbb{G}^2$, où r est un scalaire aléatoire de $\mathbb{Z}_q$, et $F(t) = G^t \times u$ pour tout scalaire $t \in \mathbb{Z}_q$;

— $\mathcal{V}(\mathsf{Spk}, m, \sigma)$: la vérification d'un couple $(m, \sigma) \in \mathbb{Z}_q \times \mathbb{G}^2$ consiste en le test

$$e(G, h) \stackrel{?}{=} e(g, \sigma_1)/e(\sigma_2, F(m)).$$

Q-3. Montrer que ce schéma est correct (une signature correctement fabriquée sera acceptée).

Supposons que l'on définisse $G = g^a$, $h = g^b$ et $u = G^{-m^*} g^\beta$ pour $a, b, m^*, \beta \in \mathbb{Z}_q$. Si on ne connaît pas a et b, on ne connaît pas la clé de signature H associée à la clé publique G. On ne peut donc pas signer tous les messages. En revanche, on suppose que l'on connaît m^* et β.

Q-4. Montrer qu'une signature valide $\sigma = (\sigma_1, \sigma_2)$ de m^* fournit $\mathsf{CDH}_{\mathbb{G},g}(G, h)$.

Q-5. Montrer alors que ce schéma de signature est sûr contre les falsifications sélectives (où l'adversaire annonce le message m^* sur lequel il compte émettre une falsification avant d'avoir vu les paramètres et les clés, mais sans messages connus).

— On précisera l'hypothèse algorithmique nécessaire.

— On présentera une réduction, puis on conclura avec une borne sur $\mathsf{Succ}^{\mathsf{sf-nma}}_{\mathsf{Sign}}(t)$.

On pourra utiliser t_{exp}, le temps d'une exponentiation dans $\mathbb{G}$ dans cette relation.

Note : Le succès $\mathsf{Succ}^{\mathsf{sf-nma}}_{\mathsf{Sign}}(\mathcal{A})$ d'un attaquant $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ pour produire une falsification sélective sans message connu contre un schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ est

$$\mathsf{Succ}^{\mathsf{sf-nma}}_{\mathsf{Sign}}(\mathcal{A}) = \Pr\left[\begin{array}{l} (m, \mathsf{state}) \leftarrow \mathcal{A}_1(), (\mathsf{Spk}, \mathsf{Ssk}) \leftarrow \mathcal{K}(1^k) \\ \sigma \leftarrow \mathcal{A}(\mathsf{Spk}, \mathsf{state}) \end{array} : \mathcal{V}(\mathsf{Spk}, m, \sigma) = 1 \right].$$

En gardant les notations/définitions ci-dessus, pour tout message $m \in \mathbb{Z}_q$, on définit, pour $\rho \in \mathbb{Z}_q$,

$$\sigma_1 = h^{-\beta/(m-m^*)} F(m)^\rho \quad \sigma_2 = g^\rho h^{1/(m^*-m)}.$$

Q-6. Montrer qu'un tel couple (σ_1, σ_2) est une signature valide de m. On précisera la valeur de r implicitement utilisée.

Q-7. Conclure sur le niveau de sécurité de ce schéma de signature : contre quel type d'attaques est-il sûr ?

12.3 Chiffrement basé sur l'identité

L'avantage du chiffrement asymétrique, par rapport au chiffrement symétrique, est la possibilité d'envoyer un message chiffré à un interlocuteur avec qui l'on n'a jamais mis de secret en commun : seule sa clé publique est nécessaire. Néanmoins, cette clé publique doit être authentique, et donc, une autorité de certification est nécessaire.

Le chiffrement basé sur l'idendité permet de se passer de cette autorité de certification, puisque la clé publique de chacun est sa propre identité (ou son adresse e-mail) que l'on notera ID, et une autorité se charge de distribuer les clés de déchiffrement associées.

Considérons le schéma de chiffrement basé sur l'identité (IBE = *Identity-Based Encryption*) $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{Extract}, \mathcal{E}, \mathcal{D})$ suivant :

— Données communes : le contexte ci-dessus, ($\mathbb{G} = \langle g \rangle, \mathbb{G}_T, e$), où g est un générateur d'ordre premier q. Nous avons également deux autres générateurs h et u.
— $\mathsf{Setup}(1^k)$: génération des paramètres du système, qui consistent en la clé publique maître $\mathsf{mpk} = G = g^s$, et la clé secrète maître $\mathsf{msk} = H = h^s$, pour $s \in \mathbb{Z}_q$. La clé G est connue de tous, ainsi que g, h et u.
— $\mathsf{Extract}(\mathsf{msk}, \mathsf{ID})$: l'algorithme d'extraction fournit, à l'aide de la clé secrète maître H, la clé de déchiffrement pour l'utilisateur d'identité $\mathsf{ID} \in \mathbb{Z}_q$. Il s'agit de $\mathsf{Esk} = (\mathsf{sk}_1 = H \times F(\mathsf{ID})^r, \mathsf{sk}_2 = g^r)$ pour $r \in \mathbb{Z}_q$ et $F(x)$ définie par $u \times G^x$, pour tout scalaire $x \in \mathbb{Z}_q$.
— $\mathcal{E}(\mathsf{mpk}, \mathsf{ID}, m)$: pour chiffrer un message $m \in \mathbb{G}_T$ à un individu ID, on génère un aléa $t \in \mathbb{Z}_q$, puis le chiffré consiste en le triplet $c = (c_1 = F(\mathsf{ID})^t, c_2 = g^t, c_3 = K \times m)$ où $K = e(G, h)^t$.
— $\mathcal{D}(\mathsf{Esk}, c)$: pour déchiffrer un triplet $c = (c_1, c_2, c_3)$, on calcule $K = e(c_2, \mathsf{sk}_1)/e(\mathsf{sk}_2, c_1)$, puis on démasque $m = c_3/K$.

Q-8. Montrer qu'il est difficile de générer une clé Esk valide pour une identité ID^* annoncée à l'avance, sans l'aide de l'autorité, même après plusieurs requêtes d'extraction.

Q-9. Montrer que ce schéma est correct (un chiffré sera convenablement déchiffré).

Q-10. Montrer que pour une identité ID^* choisie initialement par l'attaquant, ce schéma est sémantiquement sûr, sous une hypothèse bien précise. On présentera la réduction.

Note : L'avantage $\mathsf{Adv}^{\mathsf{sid}^*-\mathsf{ind}-\mathsf{cpa}}_{\mathsf{PKE}}(\mathcal{A})$ d'un attaquant $\mathcal{A}$ contre l'indistingabilité à clairs choisis, mais à identité sélective, d'un schéma de chiffrement basé sur l'identié $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{Extract}, \mathcal{E}, \mathcal{D})$ est

$$\mathsf{Adv}^{\mathsf{sid}^*-\mathsf{ind}-\mathsf{cpa}}_{\mathsf{IBE}}(\mathcal{A}) = 2\times\Pr\left[\begin{array}{ll}(\mathsf{ID}^*, \mathsf{state}) \leftarrow \mathcal{A}_1(), (\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{Setup}(1^k) & \\ (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_2(\mathsf{mpk}, \mathsf{state}) & : \mathcal{A}_3(c, \mathsf{state}) = b \\ c = \mathcal{E}(\mathsf{mpk}, \mathsf{ID}^*, m_b) & \end{array}\right] - 1.$$

On remarque que la définition est ici presque classique (par rapport à un schéma de chiffrement à clé publique), puisque l'oracle $\mathsf{Extract}$ n'apparaît pas. On permet néanmoins à l'attaquant de choisir la clé publique (l'identité ID^*) pour laquelle sera généré le chiffré challenge avant de voir les paramètres/clés du système.

Dans un tel environnement, basé sur l'identité, la notion $\mathsf{SID}^* - \mathsf{IND} - \mathsf{CPA}$ est faible (d'où le "*"), puisque l'attaquant n'a pas accès à l'oracle $\mathsf{Extract}$.

Q-11. Si l'on donne accès à l'attaquant à l'oracle $\mathsf{Extract}$, quelles sont les contraintes à imposer quant à ces requêtes pour que le niveau de sécurité soit accessible ?

Q-12. Montrer que ce schéma satisfait le niveau de sécurité $\mathsf{SID} - \mathsf{IND} - \mathsf{CPA}$: « Indistingabilité selon des attaques à clairs et identités choisis, sous identité challenge sélective » (chiffrement des messages de son choix, et extraction des clés secrètes pour les identités de son choix, mais engagement préalable sur l'identité ID^* du chiffré challenge).
— On précisera l'hypothèse algorithmique nécessaire.
— On présentera une réduction.

12.4 Identité Sélective et Chiffrement $\mathsf{IND-CCA2}$

Cette notion de chiffrement basé sur l'identité avec sélection préalable de l'identité challenge est finalement relativement forte. Bien que l'on sache faire encore mieux (permettre le choix de cette identité au moment de la requête au challenger, et même de donner accès à un oracle de déchiffrement), un tel niveau de sécurité permet de générer un schéma de chiffrement à clé publique sémantiquement sûr selon des attaques à chiffrés choisis.

Considérons un schéma de chiffrement basé sur l'identité $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{Extract}, \mathcal{E}, \mathcal{D})$, ainsi qu'un schéma de signature $\mathsf{Sign} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$.

On définit le schéma de chiffrement suivant :

- Génération de clés : on exécute le Setup de l'IBE, pour obtenir $(\mathsf{msk}, \mathsf{mpk})$, puis on définit les clés du schéma de chiffrement $\mathsf{Esk} \leftarrow \mathsf{msk}$ et $\mathsf{Epk} \leftarrow \mathsf{mpk}$;
- Chiffrement : pour chiffrer un message m sous Epk
  - génération de clés de signature : $\mathcal{K}() \rightarrow (\mathsf{Ssk}, \mathsf{Spk})$
  - chiffrement de m avec IBE sous l'identité Spk : $c = \mathcal{E}(\mathsf{Epk}, \mathsf{Spk}, m)$;
  - signature de c : $\sigma = \mathcal{S}(\mathsf{Ssk}, c)$.

  Le chiffré est constitué de $(\mathsf{Spk}, c, \sigma)$.
- Déchiffrement : pour déchiffrer $(\mathsf{Spk}, c, \sigma)$, avec la clé secrète $\mathsf{Esk} = \mathsf{msk}$
  - vérification de signature : $\mathcal{V}(\mathsf{Spk}, c, \sigma)$? En cas d'erreur, on refusera le chiffré (chiffré non valide) ;
  - extraction : $\mathsf{sk} = \mathsf{Extract}(\mathsf{msk}, \mathsf{Spk})$;
  - déchiffrement : $m = \mathcal{D}(\mathsf{sk}, c)$.

Q-13. Montrer que ce schéma est correct (un chiffré sera convenablement déchiffré).

Q-14. Montrer que si le schéma IBE est $\mathsf{SID-IND-CPA}$, et que la signature est non-malléable (impossibilité de générer une nouvelle signature même d'un message déjà signé) avec au plus un accès à l'oracle de signature, alors ce schéma de chiffrement est $\mathsf{IND-CCA2}$.

Information : On commencera la simulation avec une clé de vérification Spk^* (dont on ne connaît pas la clé de signature). Puis on injectera cette clé dans le chiffré challenge. On montrera comment simuler les réponses aux requêtes de déchiffrement.

13. Group Key Exchange

A group key agreement protocol involves n players $U_0, \ldots, U_{n-1}$, that communicate on a public channel. They eventually agree on a common secret value K to be thereafter used to establish a secure (private) channel.

In the following protocol, the players $U_0, \ldots, U_{n-1}$ are organized in a ring, and indices are modulo n: $u_n = u_0$. We also assume that they each own a pair of signing-verification keys $(\mathsf{sk}_i, \mathsf{vk}_i)$. The protocol works as follows, in a group $\mathbb{G}$, of prime order q, with a generator g:

- each user U_i chooses a random scalar x_i and publishes $X_i = g^{x_i}$;
- each user U_i computes $Y_i = X_{i-1}^{x_i}$, $Y_{i+1} = X_{i+1}^{x_i}$ and publishes $Z_i = Y_{i+1}/Y_i$;
- each user U_i computes $T = \prod_{i=1}^{n} Z_i$ and checks whether $T = 1$ or not.
  If the equality holds, U_i can compute the Y_j, for $j = i - 1$ to $n - i + 1$.
  The common secret key is $K = \prod_{i=1}^{n} Y_i$.

Q-1. Explain how they can compute all the Y_k's and then K.

Let us define the distributions

$$\begin{array}{rcl}
\mathcal{D}_0 & = & \{(g^{x_1}, \ \ldots, \ g^{x_n}, \ g^{x_1 x_2}, \ g^{x_2 x_3}, \ \ldots, \ g^{x_{n-1} x_n}, \ g^{x_n x_1} \ ) \in \mathbb{G}^{2n}, |, x_i \stackrel{R}{\leftarrow} \mathbb{Z}_q\} \\
\mathcal{D}_1 & = & \{(g^{x_1}, \ \ldots, \ g^{x_n}, \ g^{y_1}, \ \ g^{y_2}, \ \ \ldots, \ g^{y_{n-1}}, \ \ g^{y_n} \ \ ) \in \mathbb{G}^{2n}, |, x_i, y_i \stackrel{R}{\leftarrow} \mathbb{Z}_q\}
\end{array}$$

Q-2. Show that the two distributions are indistinguishable under the DDH assumption. To this aim

- give a sequence of hybrid distributions;
- give the relation between advantages of a distinguisher of $\mathcal{D}_0$ and $\mathcal{D}_1$, and a DDH-distinguisher.

Q-3. Detail a security model (with a game between the adversary and a challenger) for **passive attacks only**, in the vein of the Real-or-Random, to characterize the privacy of the common key K. Provide a security proof against passive adversaries.

Q-4. Enhance the security model to cover **active adversaries**. Explain how one can use the signing keys to provide authentication of the users to all the other players. Describe the full protocol, trying to minimize the use of the signatures.

Q-5. State and prove the security result about privacy of the common key with respect to outsider adversaries (with no appropriate signing keys to be part of the group).

Q-6. Insider adversaries or corrupted players (that can communicate between them) may want to bias the common key K so that an outsider friend (with whom they cannot directly communicate) can more easily guess it, and then eavesdrop the later communication under K:

- explain how $\lceil n/2 \rceil$ insider adversaries (or players under the control of the adversary) can bias the key K (set a few bits): Consider an even number of users, where honest and corrupted players alternate on the ring;
- explain how 2 insider adversaries can decide on the value of the key K, and set it to any pre-determined value K', if they are neighbors. A computational assumption is required by the adversary to hide this attack, but it can be justified.

Q-7. Discuss how one could avoid this kind of attack, and thus provide *contributiveness*: if not too many players are under the control of the adversary, the key K is uniformly distributed.

14. Anonymous Encryption and Robustness

The primary goal of an encryption scheme is *data confidentiality*: one cannot learn anything about the plaintext.

This security notion is modeled by the *indistinguishability* game: given an encryption scheme $\mathcal{S} = (\mathsf{Setup}, \mathcal{KG}, \mathcal{E}, \mathcal{D})$, the challenger runs the setup algorithm Setup and the key generation algorithm $\mathcal{KG}$ to generate the encryption key ek and the associated decryption key dk. It provides the adversary $\mathcal{A}$ with the encryption key, and $\mathcal{A}$ outputs two messages (m_0, m_1). The challenger flips a bit $b \xleftarrow{R} \{0,1\}$ and provides $\mathcal{A}$ with an encryption $c^* = \mathcal{E}_{\mathsf{ek}}(m_b)$ of m_b under ek. Eventually, $\mathcal{A}$ has to guess b. To this aim, it outputs b'.

$\mathsf{Exp}^{\mathsf{ind}-b}_{\mathcal{S},\mathcal{A}}(k)$
1. $\mathsf{params} \leftarrow \mathsf{Setup}(1^k)$
2. $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathcal{KG}(\mathsf{params})$
3. $(m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{params}, \mathsf{ek})$
4. $c^* \leftarrow \mathcal{E}_{\mathsf{ek}}(m_b)$
5. $b' \leftarrow \mathcal{A}(\mathsf{state}, c^*)$
6. RETURN b'

The quality of the adversary $\mathcal{A}$ is measured by its advantage

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A}) = \Pr[1 \leftarrow \mathcal{A} \mid b = 1] - \Pr[1 \leftarrow \mathcal{A} \mid b = 0] = \Pr[\mathsf{Exp}^{\mathsf{ind}-1}_{\mathcal{S},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind}-0}_{\mathcal{S},\mathcal{A}}(k) = 1].$$

The security of the encryption scheme $\mathcal{S}$ is measured by the advantage of the best adversary within time t:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(t) = \max_{\mathcal{A} \le t} \{\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A})\}.$$

Another security goal is *anonymity*: one cannot learn anything about the recipient. More precisely, between two possible public keys, no adversary should be able to guess which one has been used to generate the ciphertext.

Q-1. Let us consider that every users are associated with pairs $(\mathsf{ek}, \mathsf{dk})$.
How would you formalize anonymity, with a security game as above?
Note that this notion has been termed: *key privacy*

Let us consider the following RSA encryption variant:

- $\mathsf{Setup}(k)$: $\mathsf{params} = (k, \ell, \mathcal{H})$, where $\mathcal{H}$ is a random hash function onto $\{0,1\}^\ell$;
- $\mathcal{KG}(\mathsf{params})$: Choose two random k-bit prime integers p and q, set $n = pq$ and $e = 2^{16}+1$, and define $\mathsf{ek} = (\mathcal{H}, n, e)$, $\mathsf{dk} = (\mathcal{H}, n, d = e^{-1} \bmod \varphi(n))$;
- $\mathcal{E}_{\mathsf{ek}}(m)$, for $m \in \{0,1\}^\ell$: choose $r \xleftarrow{R} \mathbb{Z}_n^\star$, and set $c = (c_1 = r^e \bmod n, c_2 = \mathcal{H}(r) \oplus m)$;
- $\mathcal{D}_{\mathsf{dk}}(c)$, for $c = (c_1, c_2) \in \mathbb{Z}_n^\star \times \{0,1\}^\ell$: $m = c_2 \oplus \mathcal{H}(c_1^d \bmod n)$.

Q-2. Show that this RSA encryption scheme provides *data confidentiality*, in the random oracle model.
Detail a proof by successive alterations of the security game, and precise the security level under the intractability of the RSA problem.

Q-3. Does this RSA encryption scheme provide *key privacy*?
If yes, prove it; if not, exhibit a distinguisher, and evaluate its quality, with any reasonable assumption on the distribution of the integers you would need.

Q-4. How one can improve the *key privacy* property for this scheme?

Let us consider the ElGamal encryption:

- $\mathsf{Setup}(k)$: $\mathsf{params} = (p, g, \mathbb{G})$, where g is a generator of the cyclic group $\mathbb{G}$ of prime order p;
- $\mathcal{KG}(\mathsf{params})$: Choose a random $x \xleftarrow{R} \mathbb{Z}_p$ and define $\mathsf{ek} = (\mathbb{G}, p, g, y = g^x)$, $\mathsf{dk} = (\mathbb{G}, p, g, x)$;
- $\mathcal{E}_{\mathsf{ek}}(m)$, for $m \in \mathbb{G}$: choose $r \xleftarrow{R} \mathbb{Z}_p$, and set $c = (c_1 = g^r, c_2 = y^r \times m)$;
- $\mathcal{D}_{\mathsf{dk}}(c)$, for $c = (c_1, c_2) \in \mathbb{G}^2$: $m = c_2/c_1^x$.

Q-5. Show that the ElGamal encryption scheme provides both *data confidentiality* and *key privacy*.
Detail the proofs by successive alterations of the security games, and precise the security levels under the intractability of the DDH problem.

The *robustness* property is an additional property that allows any recipient to easily check whether the ciphertext is targeted to him. More precisely, an encryption scheme is *(weakly) robust* if no adversary can find a message M and two users (or equivalently two public keys) ek_0 and ek_1 such that an honestly generated ciphertext c of M under ek_0 can be decrypted successfully under the decryption key associated to ek_1.

Q-6. Is the ElGamal encryption scheme *(weakly) robust*?

A natural solution is to add redundancy:

- $\mathsf{Setup}(k)$: $\mathsf{params} = (p, g, \mathbb{G}, \mathcal{H})$, where $\mathcal{H}$ is a random hash function onto $\{0,1\}^\ell$;
- $\mathcal{KG}(\mathsf{params})$: Choose a random $x \xleftarrow{R} \mathbb{Z}_p$ and define $\mathsf{ek} = (\mathbb{G}, p, g, \mathcal{H}, y = g^x)$, $\mathsf{dk} = (\mathbb{G}, p, g, \mathcal{H}, x)$;
- $\mathcal{E}_{\mathsf{ek}}(m)$, for $m \in \mathbb{G}$: choose $r \xleftarrow{R} \mathbb{Z}_p$, and set $c = (c_1 = g^r, c_2 = y^r \times m, c_3 = \mathcal{H}(m))$;
- $\mathcal{D}_{\mathsf{dk}}(c)$, for $c = (c_1, c_2, c_3) \in \mathbb{G}^2 \times \{0,1\}^\ell$: $m = c_2/c_1^x$, check whether $c_3 = \mathcal{H}(m)$ before outputting m or "invalid".

Q-7. Does this ElGamal variant provide *data confidentialy*, *key privacy* and *(weak) robustness*?

Another variant with redundancy:

- $\mathsf{Setup}(k)$: $\mathsf{params} = (p, g, h, \mathbb{G})$, where g and h are independent generators of $\mathbb{G}$;
- $\mathcal{KG}(\mathsf{params})$: Choose random $x_1, x_2 \xleftarrow{R} \mathbb{Z}_p$ and define $\mathsf{ek} = (\mathbb{G}, p, g, h, y_1 = g^{x_1}, y_2 = g^{x_2})$, $\mathsf{dk} = (\mathbb{G}, p, g, h, x_1, x_2)$;
- $\mathcal{E}_{\mathsf{ek}}(m)$, for $m \in \mathbb{G}$: choose $r \xleftarrow{R} \mathbb{Z}_p$, and set $c = (c_1 = g^r, c_2 = y_1^r \times m, c_3 = y_2^r)$;
- $\mathcal{D}_{\mathsf{dk}}(c)$, for $c = (c_1, c_2, c_3) \in \mathbb{G}^3$: $m = c_2/c_1^{x_1}$, check whether $c_1^{x_2} = c_3$ before outputting m or "invalid".

Q-8. Does this ElGamal variant provide *data confidentialy*, *key privacy* and *(weak) robustness*?

Q-9. Discuss about generic techniques to make an anonymous encryption scheme robust.

15. Twin-Diffie-Hellman

The goal of this exercise is to study variants of ElGamal key encapsulation, based on the **Twin Diffie-Hellman**.

15.1 Diffie-Hellman Problems

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q. For any $X = g^x, Y = g^y \in \mathbb{G}$, we denote $\mathsf{DH}_g(X, Y) = g^{xy}$, the Diffie-Hellman value of X and Y in basis g. The Diffie-Hellman problems in basis g are defined by:

- The Computational Diffie-Hellman problem (CDH_g): Given $X, Y \xleftarrow{R} \mathbb{G}$, compute $Z = \mathsf{DH}_g(X, Y)$;
- The Decisional Diffie-Hellman problem (DDH_g): For $X, Y, Z \xleftarrow{R} \mathbb{G}$, decide between the two tuples $(g, X, Y, \mathsf{DH}_g(X, Y))$ —a Diffie-Hellman tuple— and (g, X, Y, Z) —a random tuple—.

We can even define more restricted versions of the Diffie-Hellman problems, for a fixed $X \in \mathbb{G}$ too:

- The Computational Diffie-Hellman problem ($\mathsf{CDH}_{g,X}$): Given $Y \xleftarrow{R} \mathbb{G}$, compute $Z = \mathsf{DH}_g(X, Y)$;
- The Decisional Diffie-Hellman problem ($\mathsf{DDH}_{g,X}$): For $Y, Z \xleftarrow{R} \mathbb{G}$, decide between the two tuples $(g, X, Y, \mathsf{DH}_g(X, Y))$ —a Diffie-Hellman tuple— and (g, X, Y, Z) —a random tuple—.

Let us consider an adversary that solves the ℓ-Set Restricted Diffie-Hellman problem ($\mathsf{SCDH}_{g,X,\ell}$), with success ε within time t: for any fixed $g, X \in \mathbb{G}$, but given a random $Y \xleftarrow{R} \mathbb{G}$, $\mathcal{A}$ outputs, within time t, a set S of size at most ℓ, such that $Z = \mathsf{DH}_g(X, Y) \in S$ with probability greater than ε: $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(\mathcal{A}) \geq \varepsilon$.

Q-1. Let us consider the algorithm $\mathcal{B}$ that runs $\mathcal{A}$, and then randomly chooses a candidate in S as a solution to the $\mathsf{CDH}_{g,X}$ problem. What is its success probability (a lower bound)?

Q-2. Give a relation between $\mathsf{Succ}^{\mathsf{cdh}_{g,X}}(t)$ and $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(t)$, ignoring all the additional computations that $\mathcal{B}$ does beyond $\mathcal{A}$.

Q-3. Let us now consider the algorithm $\mathcal{B}'$ that runs $\mathcal{A}$ on $Y^\alpha g^\beta$, for scalars α and β that it has randomly chosen, to get S':

1. How to convert the set S' of candidates for $\mathsf{DH}_g(X, Y^\alpha g^\beta)$ into a set S'' of candidates for $\mathsf{DH}_g(X, Y)$?

Let us now complete the algorithm $\mathcal{B}'$ by running $\mathcal{A}$ again, but on Y, to get S. Then, it computes $I = S \cap S''$: if $I = \emptyset$, it outputs "Failure", if I contains 2 elements or more, it outputs "Error", in the last case of one element, it outputs this value as the solution.

2. Explain why, if $\mathcal{A}$ succeeds in the 2 calls (on $Y^\alpha g^\beta$ and on Y), our algorithm $\mathcal{B}'$ outputs either the correct solution (success) or "Failure".
3. What is the probability that I contains a wrong solution (possibly additionally with the right one)?
4. What is the success probability of $\mathcal{B}'$ (a lower bound)?

Q-4. Give a new relation between $\mathsf{Succ}^{\mathsf{cdh}_{g,X}}(t)$ and $\mathsf{Succ}^{\mathsf{scdh}_{g,X,\ell}}(t)$, ignoring all the additional computations that $\mathcal{B}$ does beyond $\mathcal{A}$, and namely for computing the intersection I.
When is it better than the previous one (from Q-2)?

15.2 Key Encapsulation and Indistinguishability of Keys

A key encapsulation mechanism aims at generating a session key with a partner, in a non-interactive way. Such a scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$, is defined by 4 algorithms:

- $\mathsf{Setup}(1^k)$ generates the public parameters params;
- $\mathsf{KeyGen}(\mathsf{params})$ generates the pair of private and public keys $(\mathsf{dk}, \mathsf{ek})$;
- $\mathsf{Encaps}(\mathsf{ek})$ outputs a session key $K \in \{0,1\}^k$ and an encapsulation c of this key, under the public key ek;
- $\mathsf{Decaps}(\mathsf{dk}, c)$ outputs the session key K encapsulated in c under ek, if dk is the private key associated to the public key ek.

Such an encapsulated key should be indistinguishable from a random key to any third party, hence the *indistinguishability* security game: the challenger runs the setup algorithm Setup and the key generation algorithm KeyGen to generate the encapsulation key ek and the associated decapsulation key dk. It runs the encapsulation algorithm on ek, and gets the pair (K, c). The challenger flips a bit $b \xleftarrow{R} \{0,1\}$ and sets $K_b \leftarrow K$, while $K_{1-b} \xleftarrow{R} \{0,1\}^k$. It provides the triple (K_0, K_1, c) to $\mathcal{A}$. Eventually, $\mathcal{A}$ has to guess b. To this aim, it outputs b'.

$\mathsf{Exp}^{\mathsf{ind}-b}_{\mathcal{S},\mathcal{A}}(k)$
1. $\mathsf{params} \leftarrow \mathsf{Setup}(1^k)$
2. $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{params})$
3. $(K, c) \leftarrow \mathsf{Encaps}(\mathsf{ek})$
4. $K_b \leftarrow K$, $K_{1-b} \xleftarrow{R} \{0,1\}^k$
5. $b' \leftarrow \mathcal{A}(\mathsf{ek}, K_0, K_1, c)$
6. `RETURN` b'

The quality of the adversary $\mathcal{A}$ is measured by its advantage

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A}) = \Pr[1 \leftarrow \mathcal{A} \mid b = 1] - \Pr[1 \leftarrow \mathcal{A} \mid b = 0] = \Pr[\mathsf{Exp}^{\mathsf{ind}-1}_{\mathcal{S},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind}-0}_{\mathcal{S},\mathcal{A}}(k) = 1].$$

The security of the key encapsulation scheme $\mathcal{S}$ is measured by the advantage of the best adversary within time t:

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(t) = \max_{\mathcal{A} \le t} \{\mathsf{Adv}^{\mathsf{ind}}_{\mathcal{S}}(\mathcal{A})\}.$$

15.3 Hashed ElGamal

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q of length $2k$, together with a hash function $\mathcal{H}$ onto $\{0,1\}^k$, where k is the security parameter. The setup algorithm outputs the triple $\mathsf{params} = (g, q, \mathcal{H})$. Then,

- $\mathsf{KeyGen}(\mathsf{params})$: the private key is a scalar, $\mathsf{dk} = x \in \mathbb{Z}_q^\star$, and the public key is the associated group element $\mathsf{ek} = X = g^x$;
- $\mathsf{Encaps}(\mathsf{ek})$: in order to generate an encapsulated key under $\mathsf{ek} = X$, one chooses a random scalar $y \in \mathbb{Z}_q^\star$, and computes $c = Y = g^y$, while $K = \mathcal{H}(Y, Z)$, where $Z = X^y = \mathsf{DH}_g(X, Y)$;
- $\mathsf{Decaps}(\mathsf{dk}, c)$: in order to decapsulate c, with $\mathsf{dk} = x$, one computes $Z = c_1^x$ and then $K = \mathcal{H}(Y, Z)$.

Q-5. Show that this scheme provides *indistinguishability* under a SCDH problem, in the random oracle model. Detail the proof by successive alterations of the security game.

Q-6. Show that a decapsulation oracle provides a kind of DDH-oracle. Precise this oracle available to an adversary when it can ask decapsulation queries.

Then indistinguishability with decapsulation-oracle access relies on the Gap-Diffie-Hellman problem $\mathsf{GDH}_{g,X}$: Solve $\mathsf{CDH}_{g,X}$ given access to a $\mathsf{DDH}_{g,X}$ oracle.

15.4 Twin Diffie-Hellman Problems

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q. The Twin Diffie-Hellman problems are defined by:

- The Computational Twin Diffie-Hellman problem (CTDH_g): Given $X_1, X_2, Y \xleftarrow{R} \mathbb{G}$, compute $(Z_1, Z_2) = (\mathsf{DH}_g(X_1, Y), \mathsf{DH}_g(X_2, Y))$;
- The Decisional Twin Diffie-Hellman problem (DTDH_g): For $X_1, X_2, Y, Z_1, Z_2 \xleftarrow{R} \mathbb{G}$, decide between the tuples $(X_1, X_2, Y, \mathsf{DH}_g(X_1, Y), \mathsf{DH}_g(X_2, Y))$ and (X_1, X_2, Y, Z_1, Z_2).

We can also define the restricted versions $\mathsf{CTDH}_{g,X_1,X_2}$ and $\mathsf{DTDH}_{g,X_1,X_2}$, when X_1 and X_2 are fixed too, and thus instances are respectively a group element Y, or a triple (Y, Z_1, Z_2).

Q-7. For fixed $g, X_1 \in \mathbb{G}$, but any chosen $X_2 \in \mathbb{G}$ (any way one wants), prove that the $\mathsf{CTDH}_{g,X_1,X_2}$ problem is at least as hard as the CDH_{g,X_1} problem. Give the relation between the success probabilities.

Q-8. For fixed $g, X_1 \in \mathbb{G}$, let us choose random scalars $r, s \xleftarrow{R} \mathbb{Z}_q^\star$, and set $X_2 = g^s / X_1^r$.

1. Explain why no information leaks about s from X_1 and X_2.
2. When we receive a $\mathsf{DTDH}_{g,X_1,X_2}$ instance (Y, Z_1, Z_2), prove that $Z_1^r Z_2 = Y^s$ if and only if both $Z_1 = \mathsf{DH}_g(X_1, Y)$ and $Z_2 = \mathsf{DH}_g(X_2, Y)$, but with negligible probability, which provides a $\mathsf{DTDH}_{g,X_1,X_2}$ distinguisher.

15.5 Hashed Twin Diffie-Hellman

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q of length $2k$, together with a hash function $\mathcal{H}$ onto $\{0,1\}^k$. The setup algorithm outputs the triple $\mathsf{params} = (g, q, \mathcal{H})$. Then,

- $\mathsf{KeyGen}(\mathsf{params})$: the private key is a scalar, $\mathsf{dk} = (x_1, x_2) \in \mathbb{Z}_q^{\star 2}$, and the public key is the associated group elements $\mathsf{ek} = (X_1 = g^{x_1}, X_2 = g^{x_2})$;
- $\mathsf{Encaps}(\mathsf{ek})$: in order to encapsulate a key under $\mathsf{ek} = (X_1, X_2)$, one chooses a random scalar $y \in \mathbb{Z}_q^\star$, and computes $c = Y = g^y$, while $K = \mathcal{H}(Y, Z_1, Z_2)$, where $Z_1 = X_1^y$ and $Z_2 = X_2^y$;
- $\mathsf{Decaps}(\mathsf{dk}, c)$: in order to decapsulate c, with $\mathsf{dk} = x$, one computes $Z_1 = c^{x_1}$ and $Z_2 = c^{x_2}$ and eventually $K = \mathcal{H}(Y, Z_1, Z_2)$.

Q-9. For any fixed $X_1 \leftarrow \mathbb{G}$, and two random scalars $r, s \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$. If we define $\mathsf{ek} = (X_1, X_2 = g^s / X_1^r)$, show that the knowledge of (r, s) allows to simulate the decapsulation oracle, in the random oracle model, but with negligible probability.

Q-10. Show that this scheme provides *indistinguishability* even with *decapsulation-oracle access* under the CDH_g assumption, in the random oracle model.

16. Commiments Schemes

The goal of this part is to study various **commitment schemes**.

A commitment scheme is an essential primitive in cryptography, since it allows the committer to put a value in a box, so that nobody has any idea about the actual value (*hiding* property), but the committer cannot open the commitment in two different ways (*binding* property). It is also possible to prove relations between committed values, without revealing any additional information about them, excepted the fact that they satisfy these relations (*zero-knowledge* proofs).

More formally, a non-interactive commitment scheme is defined by three algorithms:

- $\mathsf{Setup}(1^k)$ outputs the public parameters of the system for a given security parameter k;
- $\mathsf{Commit}(m, r)$ takes the message m to commit to with some random coins r as inputs, and outputs the commitment c and an opening value d;
- $\mathsf{Verify}(c, m, d)$ takes the commitment c, the message m and an opening value d, and outputs yes or no, whether the verification succeeds or not.

The commitment c is sent to the receiver at the commit time, and the opening value d is sent together with the message m at the opening time, to allow verification.

The hiding property says that the commitment c does not leak information about m (either perfect secrecy, or computational indistinguishability), while the binding property says that no adversary (either powerful or computationally bounded) can generate c, $m \neq m'$ and d, d' such that both $\mathsf{Verify}(c, m, d)$ and $\mathsf{Verify}(c, m', d')$ accept.

In the following, we start with two simple commitment schemes, with complementary properties. The last part combines them to get a more powerful commitment scheme.

16.1 Pedersen Commitment

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q, and two random generators $g, h \in \mathbb{G}$. The Pedersen commitment scheme allows to commit to scalar elements from $\mathbb{Z}_q$:

Commitment: to commit to a scalar $m \in \mathbb{Z}_q$, one chooses a random $r \xleftarrow{\$} \mathbb{Z}_q$, and sets $c \leftarrow g^m h^r$, while the opening value is set to r;

Opening: to open a commitment $c \in \mathbb{G}$, one reveals the pair (m, r). If $c = g^m h^r$, the receiver accepts the opening to m, otherwise it refuses.

Q-1. Show that this commitment scheme is *perfectly hiding*: even a powerful adversary cannot have any idea about the committed value.

Q-2. Show that this commitment scheme is *computationally binding*: unless one can break a problem (to be specified), not adversary can open a commitment in two different ways.

Q-3. Show that this commitment scheme is *equivocal*: using a trapdoor (known to the simulator only, at the time of generation of the parameters (g, h), with an alternative but indistinguishable Setup algorithm), the simulator can generate a commitment $c \in \mathbb{G}$ so that it can open it later in any way of its choice.

Q-4. Under which condition can we use the binding property and the equivocality in a security proof?

Q-5. Under which condition can we use the hiding property and the equivocality in a security proof?

16.2 ElGamal Commitment

Let us consider $\mathbb{G} = \langle g \rangle$, a cyclic group of prime order q, and two random generators $g, h \in \mathbb{G}$. The ElGamal commitment scheme allows to commit to group elements from $\mathbb{G}$:

Commitment: to commit to a group element $M \in \mathbb{G}$, one chooses a random $r \xleftarrow{\$} \mathbb{Z}_q$, and sets $c \leftarrow (c_0 = g^r, c_1 = Mh^r)$, while the opening value is set to r;

Opening: to open a commitment $c \in \mathbb{G}$, one reveals the pair (M, r). If $c = (g^r, Mh^r)$, the receiver accepts the opening to M, otherwise it refuses.

Q-6. Show that this commitment scheme is *perfecly binding*: even a powerful adversary cannot open a commitment in two different ways.

Q-7. Show that this commitment scheme is *computationally hiding*: unless one can break a problem (to be specified), no adversary can distinguish commitments to M_0 or M_1 of its choice.

Q-8. Show that this commitment scheme is *extractable*: using a trapdoor ((known to the simulator only, at the time of generation of the parameters (g, h), with an alternative but indistinguishable Setup algorithm), the simulator can extract the committed value in any $c \in \mathbb{G}$.

Q-9. Under which condition can we use the binding property and the extractability in a security proof?

Q-10. Under which condition can we use the hiding property and the extractability in a security proof?

Q-11. This commitment scheme is called "ElGamal" Commitment, since this is the ElGamal encryption. How one could make the hiding property and the extractability compatible without any limitation?

16.3 Non-Interactive Commitments

Q-12. Show that a non-interactive commitment cannot be both *perfectly hiding* and *perfectly binding* (which would mean both hiding and binding against powerful adversaries).

An efficient construction in the random oracle model is $c = \mathsf{Commit}(m, r) = H(m, r)$, for a hash function $H : \{0,1\}^* \to \{0,1\}^{2k}$, on the message $m \in \{0,1\}^*$ to commit, with random coins $r \in \{0,1\}^{3k}$, for a security parameter k, while the opening value is set to r.

Q-13. Show that this is indeed a non-interactive commitment scheme (hiding and binding), and say under which assumptions (some limit or not on the number of queries to the random oracle).

Q-14. Show that it is also extractable and equivocal for a simulator that can access the list of query-answer pairs and that can program the random oracle in an indistinguishable way.

In order to build such an equivocal and extractable commitment scheme, in the standard model (without random oracles), let us start with two commitment schemes:

- an equivocal bit-commitment scheme $\mathsf{Commit}_{\mathsf{eq}}(b, r)$, for a bit $b \in \{0,1\}$ and random coins r, that outputs a commitment c, and the opening value $d \in \{0,1\}^{2k}$;

- an extractable commitment scheme $\mathsf{Commit}_{\mathsf{ext}}(D, r')$, for a bitstring $D \in \{0,1\}^{2k}$ and random coins r', that outputs a commitment c', and the opening value O.

We stress that the unique condition between the two commitment schemes is that the opening values of the former one can be committed with the latter (both in the same space $\{0,1\}^{2k}$).
On a message $m = (m_1, \ldots, m_\ell) \in \{0,1\}^\ell$ the commitment algorithm $\mathsf{Commit}(m, ((r_i)_i, (D'_i)_i, (r'_{i,b})_{i,b}))$ works as follows:

- for random coins r_i for $i = 1, \ldots, \ell$, set $(c_i, D_i) \leftarrow \mathsf{Commit}_{\mathsf{eq}}(m_i, r_i)$;
- for random coins $D'_i \xleftarrow{\$} \{0,1\}^{2k}$, set $d_{i,m_i} \leftarrow D_i$ and $d_{i,1-m_i} \leftarrow D'_i$, for $i = 1, \ldots, \ell$;
- for random coins $r'_{i,b}$, set $(c'_{i,b}, O_{i,b}) \leftarrow \mathsf{Commit}_{\mathsf{ext}}(d_{i,b}, r'_{i,b})$, for $i = 1, \ldots, \ell$ and $b = 0, 1$;
- output the commitment $(c_i, (c'_{i,b})_b)_i$, while the opening value is $(d_{i,m_i}, O_{i,m_i})_i$.

Q-15. Explain how works the Verify algorithm.

Q-16. Show this is indeed a commitment scheme: with both hiding and binding properties.

Q-17. Show this commitment scheme is also both equivocal and extractable.

17. Ring Signatures

The goal of this part is to study **ring signatures**.

A **signature scheme** allows a signer P to authenticate a message m in such a way that anybody can verify it, and nobody can impersonate P.
More formally, a signature scheme assumes a PKI (Public-Key Infrastructure): every player P_i owns a pair of signing/verification keys $(\mathsf{sk}_i, \mathsf{vk}_i)$, such that vk_i is publicly binded to his identity P_i. Then, there are two algorithms:

- $\mathsf{Sign}(\mathsf{sk}, m)$ takes the signing key sk and the message m as inputs, and outputs the signature σ;
- $\mathsf{Verify}(\mathsf{vk}, m, \sigma)$ takes the verification key vk, the message m and the signature σ as inputs, and outputs 1 if the signature is valid with respect to m and vk, and 0 otherwise.

The unforgeability result says that it is hard to generate a message/signature pair (m, σ) that is valid with respect to the verification key vk without the associated signing key sk:

$$\mathsf{Succ}^{\mathsf{euf-nma}}(\mathcal{A}) = \Pr[(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^k), (m, \sigma) \leftarrow \mathcal{A}(\mathsf{vk}) : \mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1] = \mathsf{negl}(k).$$

This is the Existential UnForgeability (EUF), without access to any signature, hence the No-Message Attack (NMA).

A **ring signature** allows a player P_j to anonymously authenticate a message m as a member of an ad-hoc group, called a ring R:

- $\mathsf{Sign}(\mathsf{sk}, R, m)$ takes a signing key sk, a ring $R = \{\mathsf{vk}_i\}$ of verification keys containing the verification key vk associated to sk, and the message m as inputs, and outputs the signature σ;
- $\mathsf{Verify}(R, m, \sigma)$ takes a ring $R = \{\mathsf{vk}_i\}$, the message m and the signature σ as inputs, and outputs 1 if the signature is valid with respect to m and R, and 0 otherwise.

The unforgeability result says that it is hard to generate a message/signature pair (m, σ) that is valid with respect to a ring R without a signing key associated to one of the verification keys in R.

> **Q-1.** Explain how a ring signature allows a member of a community to anonymously reveal and authenticate some information, just as a member of this community.

17.1 Hash-and-Sign RSA Signature

Let us consider the following generation of signing/verification keys, for a security parameter k:

- One chooses 2 random primes $2^k - 2^{k/2} < p, q < 2^k$, such that $65537 = 2^{16} + 1$ does not divides $p - 1$ nor $q - 1$;
- One sets $n \leftarrow pq$, $e \leftarrow 65537$, and $d = e^{-1} \bmod (p-1)(q-1)$;
- The signing key is set as $\mathsf{sk} = (n, d)$ and the verification key is set as $\mathsf{vk} = (n, e)$.

> **Q-2.** From the fact that the number $\pi(x)$ of prime integers smaller than x is approximately $x/\ln x$, and assuming that one can efficiently check whether an integer is prime or not, explain how efficiently one can generate such key pairs $(\mathsf{sk}, \mathsf{vk})$.

We additionally consider a hash function $\mathcal{H}$ onto $\{0,1\}^k$, that can be seen as $\{0,\ldots,2^k-1\}$. We model $\mathcal{H}$ as a random oracle (*i.e.*, every new query is answered by a truly random value in $\{0,\ldots,2^k-1\}$).

Q-3. Explain why, for any modulus n generated as above, we can consider that the output space of $\mathcal{H}$ is $\mathbb{Z}_n^\star$, up to a negligible probability. Note that the cardinality of $\mathbb{Z}_n^\star$ is $\varphi(n) = (p-1)(q-1)$.

For a PKI with such RSA keys for any player, we consider the following algorithms:

- $\mathsf{Sign}(\mathsf{sk}, m)$: for a message m and $\mathsf{sk} = (n,d)$, one outputs $\sigma = \mathcal{H}(m)^d \bmod n$;
- $\mathsf{Verify}(\mathsf{vk}, m, \sigma)$: for a message m, a signature σ, and $\mathsf{vk} = (n,e)$, one checks whether $\mathcal{H}(m) = \sigma^e \bmod n$.

Q-4. Prove that this signature scheme achieves the EUF-NMA security level under the RSA assumption. We stress that we just consider NMA security, and remind that an RSA challenge (n,e,y) expects to get as output an x such that $y = x^e \bmod n$.

17.2 Ring Signature

Still using the same PKI and hash function $\mathcal{H}$ onto $\{0,1\}^k$, we want to define a ring signature: we consider a ring $R = \{\mathsf{vk}_i = (n_i, e), i = 0,\ldots,\ell-1\}$, among which users, the j-th user wants to sign in the name of the ring R and owns the pair $(\mathsf{vk}, \mathsf{sk})$, with $\mathsf{vk} = (n,e) = \mathsf{vk}_j$ the verification key associated to the signing key $\mathsf{sk} = (n,d)$.

- One initializes the ring with $v_j \xleftarrow{\$} \{0,1\}^k$, and $h_j \leftarrow \mathcal{H}(R, m, j, v_j)$;
- For $i' = j+1,\ldots,\ell+j-1$, one sets $i = i' \bmod \ell$, chooses a random $x_i \xleftarrow{\$} \mathbb{Z}_{n_i}^\star$ and sets $y_i \leftarrow x_i^e \bmod n_i$, $v_i \leftarrow y_i \oplus h_{i-1}$, and $h_i \leftarrow \mathcal{H}(R, m, i, v_i)$;
- One closes the ring with $y_j \leftarrow v_j \oplus h_{j-1}$, and $x_j \xleftarrow{\$} y_j^d \bmod n$ (if $y_j \geq n$, one aborts);
- The signature consists of $\sigma = (v_0, x_0, \ldots, x_{\ell-1})$ for the ring $R = \{\mathsf{vk}_i, i = 0,\ldots,\ell-1\}$.

Q-5. Explain that this algorithm may fail (abort), but with negligible probability only.

Q-6. Describe the verification algorithm.

Q-7. Show that σ follows indistinguishable distributions whatever the index j is. Explain that the signer is perfectly anonymous among the ℓ users in the ring R.

We now show that it is hard for an adversary to generate a ring signature $\sigma = (v_0, x_0, \ldots, x_{\ell-1})$ on any message m on behalf of a ring R of size ℓ without being part of this ring.

Q-8. First, show that for a signature $\sigma = (v_0, x_0, \ldots, x_{\ell-1})$ to have a chance to be valid, all the queries $h_i \leftarrow \mathcal{H}(R, m, i, v_i)$, for the appropriate v_i must have been asked.

Q-9. Second, show that there must exist an index i such that the query $h_i \leftarrow \mathcal{H}(R, m, i, v_i)$ has been asked before $h_{i-1} \leftarrow \mathcal{H}(R, m, i-1, v_{i-1})$, where the indices are modulo ℓ.

Q-10. Eventually, describe in details a reduction that first guesses such an index i and then show that, if $\mathsf{vk}_i = (n_i, e)$ is not a key owned by the adversary, there is a good chance to break RSA for the modulus n_i and exponent e, by setting the value of $\mathcal{H}(R, m, i-1, v_{i-1})$ appropriately from the RSA challenge $(n = n_i, e, y)$.