0. Outils Mathématiques Utiles (Modulo `n`)

- **Inverse Modulaire `a⁻¹ (mod n)` avec Euclide Étendu**

> **But** : Trouver `x` tel que `a·x ≡ 1 (mod n)` (donc `x = a⁻¹ mod n`).
> **Condition** : L'inverse existe si `pgcd(a, n) = 1`.

Algorithme Euclide Étendu (version fiche-exam)

1. Initialiser :
   - `r0 = n`, `r1 = a`
   - `x0 = 0`, `x1 = 1`
2. Tant que `r1 ≠ 0` :
   - `q = r0 // r1`
   - `(r0, r1) = (r1, r0 - q * r1)`
   - `(x0, x1) = (x1, x0 - q * x1)`
3. À la fin, si `r0 = 1` alors l'inverse est `x0 mod n` (si négatif, ajouter `n`).

Exemple : Calculer l'inverse de 17 modulo 43

Étape	r0	r1	q	x0	x1
init	43	17		0	1
1	17	9	2	1	-2
2	9	8	1	-2	3
3	8	1	1	3	-5
4	1	0	8	-5	

- À la fin : `r0 = 1`, `x0 = -5`
- Donc l'inverse est `-5 mod 43 = 38`
- Vérif : `17 × 38 = 646 ≡ 1 mod 43`

> **Astuce** : Si `n` est premier, on peut aussi utiliser `a⁻¹ ≡ a^(n-2) mod n` (exponentiation rapide).

- **Exponentiation Modulaire Rapide (`base^exp mod n`)**

Algorithme (fiche-exam, version binaire droite-gauche)

1. `res = 1`
2. `base = base mod n`
3. Tant que `exp > 0` :
   - Si `exp` impair : `res = (res × base) mod n`
   - `base = (base × base) mod n`
   - `exp = exp // 2`
4. Résultat final : `res`

Exemple : Calculer 7^13 mod 17

- 13 en binaire : 1101
- Étapes :
  - res=1, base=7
  - bit 1 : res=7, base=49→15, exp=6
  - bit 0 : res=7, base=225→4, exp=3
  - bit 1 : res=28→11, base=16, exp=1
  - bit 1 : res=176→6, base=1, exp=0
- Résultat : **6**

---

1. Opérations de Base sur la Courbe `y² = x³ + αx + β (mod q)`

- **Addition de Points sur une Courbe Elliptique (fiche-exam)**

> **But** : Calculer `R = P + S` où `P = (xP, yP)`, `S = (xS, yS)` sur la courbe.

Cas 1 : P ≠ S

1. Calculer `λ = (yS - yP) * (xS - xP)⁻¹ mod q`
2. `xR = λ² - xP - xS mod q`
3. `yR = λ(xP - xR) - yP mod q`

Cas 2 : P = S (Doublement)

1. Calculer `λ = (3xP² + α) * (2yP)⁻¹ mod q`
2. `xR = λ² - 2xP mod q`
3. `yR = λ(xP - xR) - yP mod q`

Cas particuliers

- Si `P = O` (point à l'infini), `R = S`
- Si `S = O`, `R = P`
- Si `xP = xS` et `yP = -yS mod q`, `R = O`

Exemple : Soit `P = (2, 7)`, `S = (5, 3)`, courbe sur `F_11`, `α = 1`

- $\lambda = (3 - 7) * (5 - 2)^{-1} \bmod 11 = (-4) * 3^{-1} \bmod 11$
- $3^{-1} \bmod 11 = 4$ (car $3\times4=12\equiv1 \bmod 11$)
- $\lambda = (-4)\times4 = -16 \equiv 7 \bmod 11$
- $xR = 7^2 - 2 - 5 = 49 - 7 = 42 \equiv 9 \bmod 11$
- $yR = 7\times(2-9) - 7 = 7\times(-7) - 7 = -49 - 7 = -56 \equiv 1 \bmod 11$
- Donc R = (9, 1)

- **Multiplication Scalaire sur Courbe Elliptique (Double-and-Add, fiche-exam)**

> **But** : Calculer `Q = kP` (k entier, P point)

Algorithme

1. Écrire `k` en binaire : `k = (d_n ... d_0)_2`
2. `Q = O` (point à l'infini)

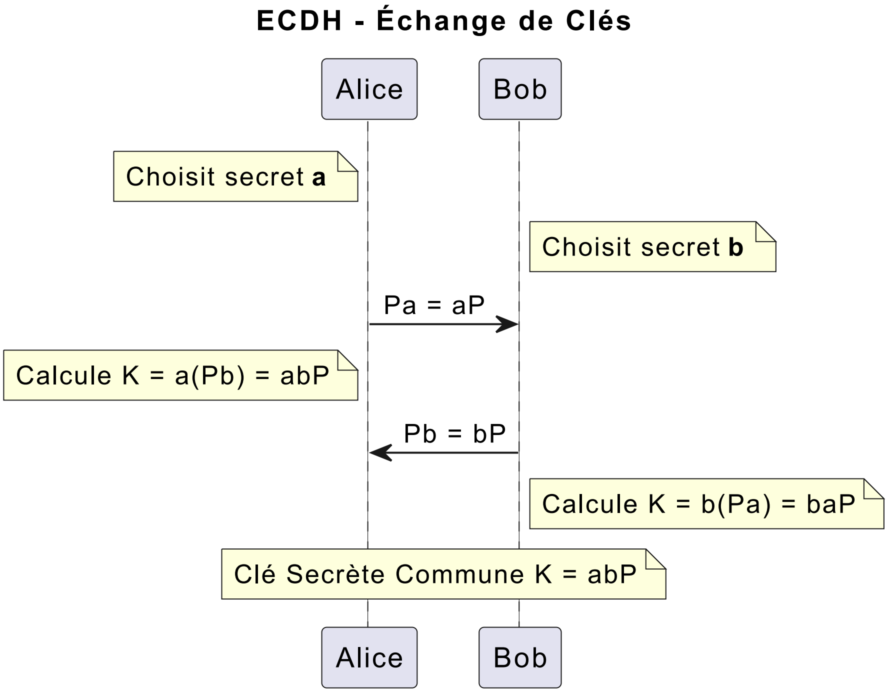
3. Pour chaque bit de gauche à droite :
    - Doubler `Q` (`Q = 2Q`)
    - Si le bit vaut 1, `Q = Q + P`
4. Résultat : `Q`

Exemple : Calculer `11P` (11 = 1011_2)

- Q = O
- bit 1 : Q = O×2 = O, Q = O+P = P
- bit 0 : Q = 2P, pas d'addition
- bit 1 : Q = 2×(2P) = 4P, Q = 4P+P = 5P
- bit 1 : Q = 2×5P = 10P, Q = 10P+P = 11P

---

2. Échange de Clés ECDH

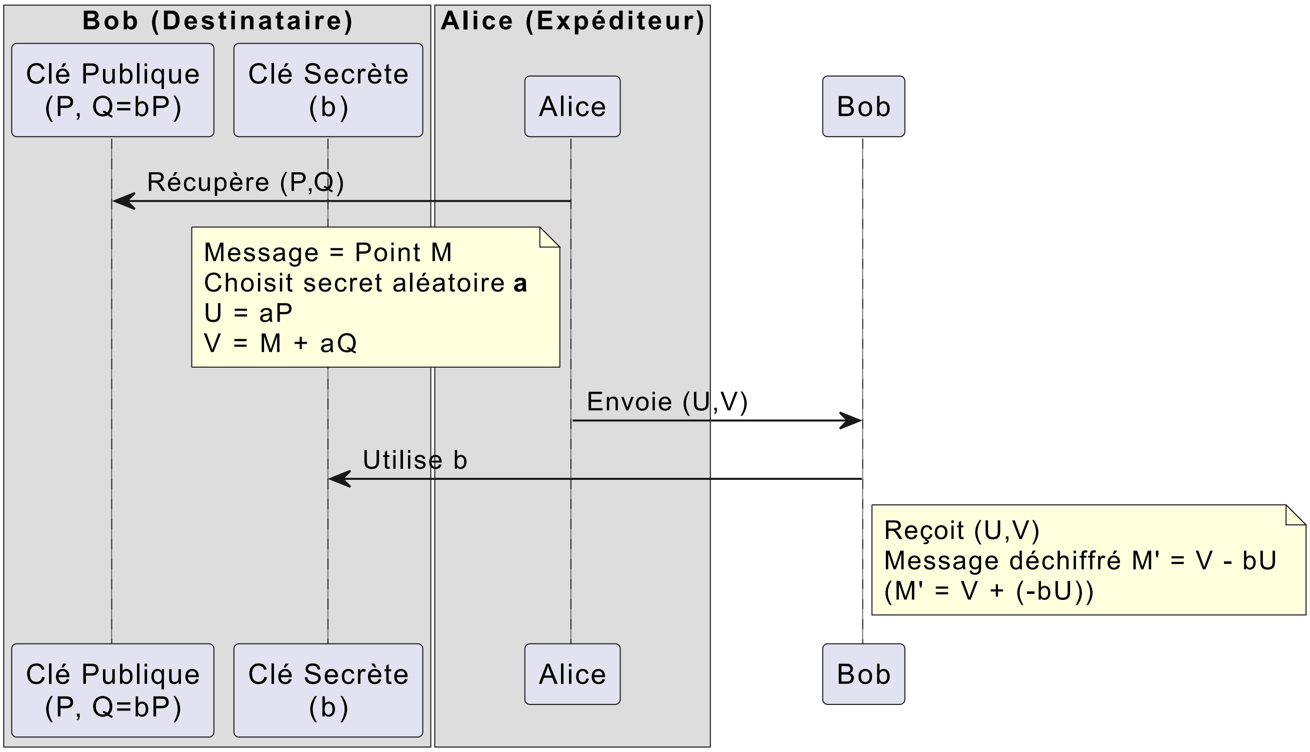


- **Public:** Courbe `Cq`, Point `P`.
- Alice: secrète `a`. Calcule `Pa = aP`. Envoie `Pa` à Bob.
- Bob: secrète `b`. Calcule `Pb = bP`. Envoie `Pb` à Alice.
- Alice calcule clé commune: `K = a * Pb = abP`.
- Bob calcule clé commune: `K = b * Pa = baP`.
- **Avantages:** Fournit un secret partagé sans échange préalable de secret. Sécurité basée sur ECDLP.
- **Inconvénients:** Vulnérable à l'attaque de l'homme du milieu (MitM) si les clés publiques ne sont pas authentifiées.

---

3. Chiffrement ElGamal-ECC (Version 1)

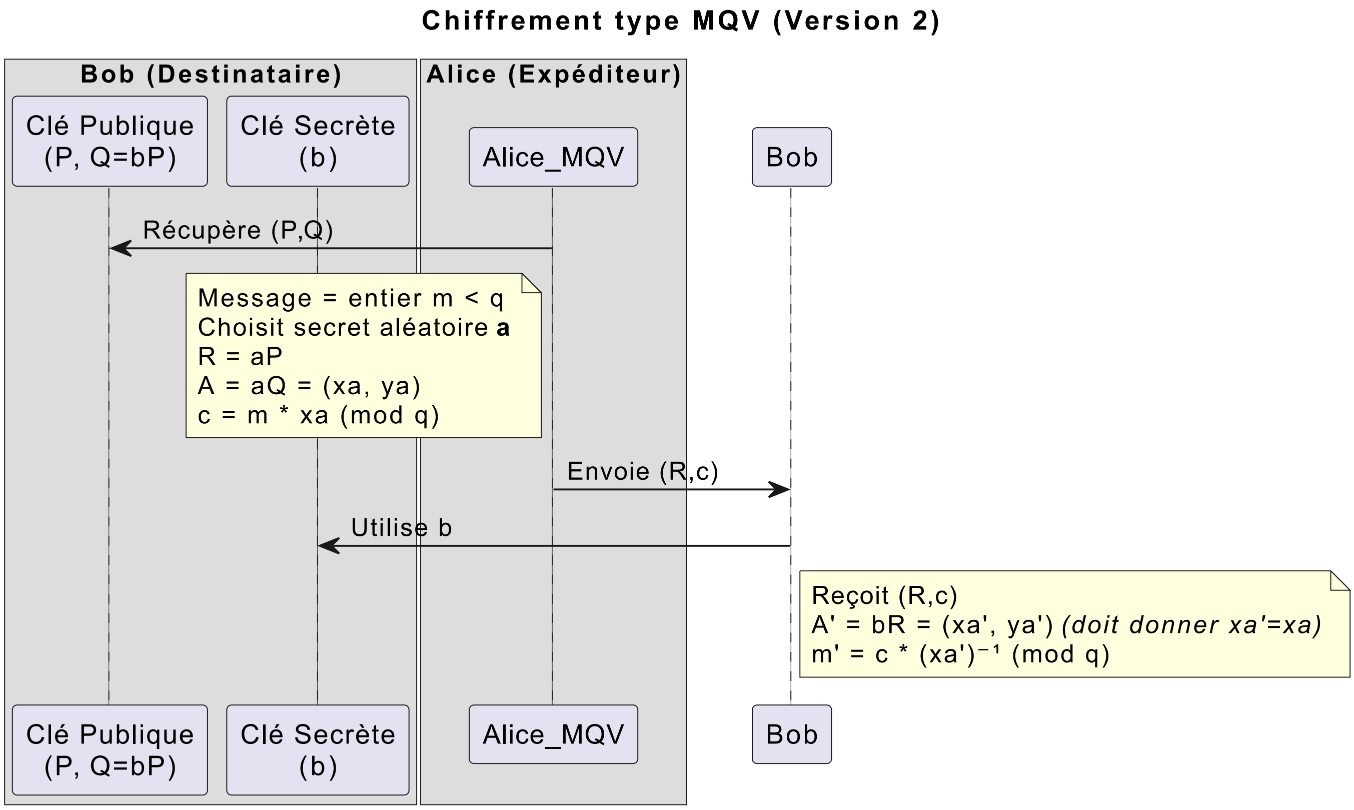
Chiffrement ElGamal-ECC (Version 1)



- **Clés Bob:** Secret `b`. Public `(P, Q = bP)`.
- **Alice chiffre Point `M`:** Secret `a`.
    1. `U = aP`
    2. `V = M + aQ`
    3. Envoie `(U,V)`.
- **Bob déchiffre `(U,V)`:**
    1. `M' = V - bU = V + (-bU)`.
- **Avantages:** Chiffrement probabiliste (sécurité sémantique si ECDLP difficile). Bien étudié.
- **Inconvénients:** Nécessite de mapper le message en un point de la courbe (non trivial). Le chiffré est constitué de deux points (taille double).

---

4. Chiffrement type MQV (Version 2)



- **Clés Bob:** Secret b. Public $(P, Q = bP)$.
- **Alice chiffre entier m:** Secret a.
  1. $R = aP$
  2. $A = aQ = (xa, ya)$
  3. $c = m * xa \pmod q$
  4. Envoie (R,c).
- **Bob déchiffre (R,c):**
  1. $A' = bR = (xa', ya')$ (Normalement $xa' = xa$)
  2. $m' = c * (xa')^{-1} \pmod q$.
- **Avantages:** Chiffre un entier directement sans mapping vers un point. Chiffré plus compact (un point et un entier).
- **Inconvénients:** La sécurité repose sur ECDLP et la sécurité de "masquer" m avec xa.

---

5. Signature ECDSA-like (Version du cours)

- **Signature ECDSA (fiche-exam)**

> **But** : Signer un message m avec la clé privée a, courbe d'ordre l, point de base P.

Signature

1. Choisir k aléatoire $< l$, $pgcd(k, l) = 1$
2. Calculer $R = kP = (xr, yr)$
3. Calculer $s = k^{-1} \times (m - a \cdot xr) \bmod l$
4. Signature = (m, s, Q, R) où $Q = aP$

Vérification

1. Vérifier `0 < s < l` et que `R` est un point valide
2. Calculer `V1 = xr·Q + s·R`
3. Calculer `V2 = m·P`
4. Signature valide si `V1 = V2`

Exemple : Soit `l = 13`, `a = 3`, `k = 5`, `m = 7`, `P` d'ordre 13, `Q = 3P`

- R = 5P = (xr, yr) (supposons xr = 4)
- k^{-1} mod 13 = 8 (car 5×8=40≡1 mod 13)
- s = 8 × (7 - 3×4) mod 13 = 8 × (7 - 12) = 8 × (-5) = -40 ≡ 5 mod 13
- Signature : (7, 5, Q, R)

Conseils Clés pour l'Examen QCM:

- **Identifier le Protocole:** La question porte-t-elle sur ECDH, ElGamal, MQV, ou Signature ? Cela guide les formules à utiliser.
- **Table d'Addition:** Si fournie, elle est reine pour les additions/doublements de points.
- **Modulo, Modulo, Modulo:** TOUS les calculs finaux sont `mod q` (pour les coordonnées) ou `mod l` (pour les scalaires dans les signatures).
- **Inverse x^{-1}:** Soit par Euclide étendu, soit par `x^(p-2) mod p` si `p` est premier.

Bonne chance pour tes révisions et l'examen !