Probabilistic procedure

Let E be an arbitrary EC, $k \in \mathbb{N}$ determining the prob. of failure (or width of the interval of messages.)
$SQR(z, p)$ returns a square root of z modulo p

Alg. 13 / Mapping of a Message M on a point of an EC E

Input: $E / \mathbb{F}_p$, $0 \leq M < \left\lfloor \frac{p}{2^k} \right\rfloor$

Output: A point (x, y) on the EC E with probability $1 - \frac{1}{2^{2^k}}$

$x \leftarrow 2^k \cdot M$
Repeat
  $z \leftarrow x^3 + ax + b \mod p$
  $x \leftarrow x + 1$
Until z is quadratic residue or $x = 2^k(M+1)$
if z is quadratic residue
  $y \leftarrow SQR(z, p)$
  return $(x-1, y)$
else
  return FAIL
end if

Remarks to probabilistic procedure of determining points on EC to a msg M

- Obviously the procedure returns a point on the EC with high prob.
- In $\mathbb{F}_p$ half of the elements are quadratic residues. Knowing that you may pick any quadratic residue with probability $1/2$ (even if they are neighbours or in an interval), the probability of failing to find one in the above alg. is given by $\left(\frac{1}{2}\right)^{2^k}$
- If a point (x, y) on the EC is given, the corresponding original integer message will be $M = \left\lfloor \frac{x}{2^k} \right\rfloor$
- Analogously to the deterministic approach, the message space needs to be reduced such that a unique (de)mapping is possible.

13.4.3 ElGamal on elliptic curves (ECElGamal)

A's private key: random number $x_a \in \{2, \dots, n-2\}$

A's public key: $x_a \cdot P$ where P is on EC with $\text{ord}(P) = n$

B wants to encrypt $m \in \langle P \rangle$, for getting m we may apply 13.4.2.

(i) B chooses random $k \in \{2, \dots, n-2\}$, computes $Q = k \cdot P$

(ii) B computes $R = k(x_a \cdot P) + m$

(iii) $B \to A: (Q, R)$

A deciphers

(i) A computes $x_a \cdot Q = x_a \cdot k \cdot P$

(ii) A computes $R - x_a \cdot Q = k \cdot x_a \cdot P + m - x_a k \cdot P = m$

Use demapping of 13.4.2 to get the corresponding integer value.

13.4.4 Elliptic Curve Integrated Encryption Scheme (ECIES)

ECIES is a variant of ElGamal introduced by Bellare, Rogaway.

A Diffie-Hellman shared secret is used to derive two symmetric keys K_1 and K_2. K_1 is used for symmetric encryption and K_2 for ciphertext authentication.

We need the following primitives

- Symmetric encryption function: ENC_{K_1} (e.g. AES)
  with corresponding decryption: DEC_{K_1}
- Message authentication code: MAC_{K_2} (e.g. HMAC)
- Key derivation function: KDF

$(K_1, K_2) = KDF(S) = H(S,0) \| H(S,1) \| H(S,2) \| \dots$
until enough bits for K_1 and K_2 are generated. H is a hash function

System parameters :

$\mathbb{F}_p$, p prime, $E: y^2 = x^3 + ax + b$ / $\mathbb{F}_p$, $P \in E(\mathbb{F}_p)$
s.t. $\text{ord}(P) = n$ is prime, $h = \frac{\#E(\mathbb{F}_p)}{n}$

Key generation

- Choose a random $d \in \{2, \dots, n-2\}$ (private key)
- Compute $Q = d \cdot P$ (public key)

Encryption of $m \in \{0,1\}^*$

1) Choose a random $k \in \{2, \dots, n-2\}$
2) $R \leftarrow k \cdot P$ $z \leftarrow h\,k \cdot Q$, if $z = 0$ goto 1)
3) $(K_1, K_2) = KDF(x_z, R)$ where x_z is the x-coordinate of z
4) $C \leftarrow ENC_{K_1}(m)$ $t \leftarrow MAC_{K_2}(C)$
5) Send (R, C, t)

Decryption

1) Ensure the validity of R : i) $R \neq 0$ ii) $R \in E(\mathbb{F}_p)$
2) $z \leftarrow h \cdot d \cdot R$, check whether $z \neq 0$
3) $(K_1, K_2) = KDF(x_z, R)$
4) $t' \leftarrow MAC_{K_2}(C)$, check whether $t' = t$
5) $m \leftarrow DE(K_1, C)$

To show that encryption works, only prove that z is computed correctly:

$h \cdot d \cdot R = h \cdot d \cdot k \cdot P = h \cdot k \cdot Q = z$ ✓

13.4.5 Elliptic Curve Digital Signature Algorithm (ECDSA)

$EC : E / \mathbb{F}_p$, generator P, $\text{ord}(P) = n$ with $L_n = \lceil \log_2(n) \rceil$ and a hash function h.

x: private key $2 < x < n-1$
$Y = x \cdot P$ is the public key.

Alg 14/ Creating signature (r, s) on a message m

Input: Message m

Output: A signature (r, s) on m

$e \leftarrow h(m)$

$z \leftarrow$ The L_n leftmost bits of e

Repeat

  Repeat

    Select random $1 \leq k \leq n-1$ with $gcd(k, n) = 1$

    $(x_1, y_1) \leftarrow k \cdot P$

    $r \leftarrow x_1 \bmod n$

  Until $r \neq 0$

  $s \leftarrow k^{-1}(z + x \cdot r) \bmod n$ //

Until $gcd(s, n) = 1$

Return (r, s)

Alg 15/ Verifying signature (r, s) on a message m

Input: Message m and (r, s)

Output: Acceptance or denial of signature (r, s) on message m

Verify that $1 \leq r, s \leq n-1$

$e \leftarrow h(m)$

$z \leftarrow$ the L_n leftmost bits of e

$w \leftarrow s^{-1} \bmod n$

$u_1 \leftarrow w z \bmod n$

$u_2 \leftarrow r \cdot w \bmod n$

$(v, y_1) \leftarrow u_1 \cdot P + u_2 \cdot Y$

if $v \bmod n = r$ then

  return accept

else

  return deny

end if

The verification process holds true as

$$(r, y_1) = u_1 \cdot P + u_2 \cdot Y = s^{-1} z \cdot P + r \cdot s^{-1} x \cdot P$$

$$= s^{-1}(z + r \cdot x) \cdot P = k \cdot P = (x_1, y_1) \equiv (r, y_1) \bmod n$$