ECDSA & EdDSA algorithms

I. ECDSA ALGORITHM

Algorithm 1 ECDSA key generation

Input: a prime p and coefficients (a, b) of E_{ECDSA}.

Generate G, which generates a cyclic group of prime order n.
Choose a random integer $x \in$ [1,n-1].
Compute $\boxed{B = xG}$.

Output: $K_{pub} = (p, a, b, n, G, B)$ and $K_{pr} = (x)$.

Algorithm 2 ECDSA signature generation

Input: K_{pub}, K_{pr}, and $H =$ hash of message .

$r\ ,\ s \leftarrow 0\ ,\ 0$
while $r = 0\ or\ s = 0$ **do**
  Choose a random integer $k \in$ [1,n-1].
  Compute $\boxed{R = kG}$
  if $r \neq 0$ **then**
    $s = \boxed{k^{-1}}\ (H + rK_{pr})\ (mod\ n)$
    if $s \neq 0$ **then**
      return (r, s)
    else
      $r \leftarrow 0$
    end if
  end if
end while

Output: (r, s)

Algorithm 3 ECDSA verification

Input: K_{pub}, (r, s), and $H =$ hash of message.

if $r, s \notin$ [1,n-1] **then**
  return Signature Invalid
end if
Compute $w = \boxed{s^{-1}}\ (mod\ n)$
Compute $u_1 = Hw\ (mod\ n)$ and $u_2 = rw\ (mod\ n)$
Compute $X = \boxed{u_1G + u_2B}$
if $X =$ point at infinity **then**
  return Signature Invalid
end if
if $X_x = r$ **then**
  return Signature Valid
else
  return Signature Invalid
end if

Output: *valid* or *invalid.*

II. EDDSA ALGORITHM

Algorithm 4 EdDSA key generation

Input: a prime p and coefficients (a, d) of E_{EdDSA}.

Generate G, which generates a cyclic group of prime order n.
Choose a random integer $x \in$ [1,n-1].
Compute $\boxed{B = xG}$.

Output: $K_{pub} = (p, a, d, n, G, B)$ and $K_{pr} = (x)$.

Algorithm 5 EdDSA signing

Input: K_{pub}, K_{pr}, and $H =$ hash of message.

Compute $r = H\ (mod\ n)$
Compute $\boxed{R = rG}$
Compute $h = Hash(R,\ B,\ H)$
Compute $s = (s + hK_{pr})\ (mod\ n)$
return (R, s)

Output: (R, s)

Algorithm 6 EdDSA verification

Input: K_{pub}, (R, s), and $H =$ hash of message.

if $s \notin$ [1,n-1] **then**
  return Signature Invalid
end if
Compute $h = Hash(R,\ K_{pub},\ H)$
Compute $\boxed{P_1 = sG}$
Compute $\boxed{P_2 = R + hB}$
if $P_1 = P_2$ **then**
  return Signature Valid
else
  return Signature Invalid
end if

Output: *valid* or *invalid.*