| | |
|---|---|
| **Public parameter creation** | |
| A trusted party chooses and publishes a (large) prime $p$ and an integer $g$ having large prime order in $\mathbb{F}_p^*$. | |
| **Private computations** | |
| **Alice** | **Bob** |
| Choose a secret integer $a$. Compute $A \equiv g^a \pmod{p}$. | Choose a secret integer $b$. Compute $B \equiv g^b \pmod{p}$. |
| **Public exchange of values** | |
| Alice sends $A$ to Bob $\longrightarrow$ $A$ | |
| $B$ $\longleftarrow$ Bob sends $B$ to Alice | |
| **Further private computations** | |
| **Alice** | **Bob** |
| Compute the number $B^a \pmod{p}$. | Compute the number $A^b \pmod{p}$. |
| The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$. | |

Table 2.2: Diffie–Hellman key exchange

| | |
|---|---|
| **Public parameter creation** | |
| A trusted party chooses and publishes a large prime $p$ and an element $g$ modulo $p$ of large (prime) order. | |
| **Alice** | **Bob** |
| **Key creation** | |
| Choose private key $1 \le a \le p-1$. Compute $A = g^a \pmod{p}$. Publish the public key $A$. | |
| **Encryption** | |
| | Choose plaintext $m$. Choose random element $k$. Use Alice's public key $A$ to compute $c_1 = g^k \pmod{p}$ and $c_2 = mA^k \pmod{p}$. Send ciphertext $(c_1, c_2)$ to Alice. |
| **Decryption** | |
| Compute $(c_1^a)^{-1} \cdot c_2 \pmod{p}$. This quantity is equal to $m$. | |

Table 2.3: Elgamal key creation, encryption, and decryption

| | |
|---|---|
| **Bob** | **Alice** |
| **Key creation** | |
| Choose secret primes $p$ and $q$. Choose encryption exponent $e$ with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and $e$. | |
| **Encryption** | |
| | Choose plaintext $m$. Use Bob's public key $(N, e)$ to compute $c \equiv m^e \pmod{N}$. Send ciphertext $c$ to Bob. |
| **Decryption** | |
| Compute $d$ satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$. Compute $m' \equiv c^d \pmod{N}$. Then $m'$ equals the plaintext $m$. | |

Table 3.1: RSA key creation, encryption, and decryption

| | |
|---|---|
| **Samantha** | **Victor** |
| **Key creation** | |
| Choose secret primes $p$ and $q$. Choose verification exponent $e$ with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and $e$. | |
| **Signing** | |
| Compute $d$ satisfying $de \equiv 1 \pmod{(p-1)(q-1)}$. Sign document $D$ by computing $S \equiv D^d \pmod{N}$. | |
| **Verification** | |
| | Compute $S^e \bmod N$ and verify that it is equal to $D$. |

Table 4.1: RSA digital signatures

| | |
|---|---|
| **Public parameter creation** | |
| A trusted party chooses and publishes a large prime $p$ and primitive root $g$ modulo $p$. | |
| **Samantha** | **Victor** |
| **Key creation** | |
| Choose secret signing key $1 \le a \le p-1$. Compute $A = g^a \pmod{p}$. Publish the verification key $A$. | |
| **Signing** | |
| Choose document $D \bmod p$. Choose random element $1 < k < p$ satisfying $\gcd(k, p-1) = 1$. Compute signature $S_1 \equiv g^k \pmod{p}$ and $S_2 \equiv (D - aS_1)k^{-1} \pmod{p-1}$. | |
| **Verification** | |
| | Compute $A^{S_1} S_1^{S_2} \bmod p$. Verify that it is equal to $g^D \bmod p$. |

Table 4.2: The Elgamal digital signature algorithm

| | |
|---|---|
| **Public parameter creation** | |
| A trusted party chooses and publishes large primes $p$ and $q$ satisfying $p \equiv 1 \pmod{q}$ and an element $g$ of order $q$ modulo $p$. | |
| **Samantha** | **Victor** |
| **Key creation** | |
| Choose secret signing key $1 \le a \le q-1$. Compute $A = g^a \pmod{p}$. Publish the verification key $A$. | |
| **Signing** | |
| Choose document $D \bmod q$. Choose random element $1 < k < q$. Compute signature $S_1 \equiv (g^k \bmod p) \bmod q$ and $S_2 \equiv (D + aS_1)k^{-1} \pmod{q}$. | |
| **Verification** | |
| | Compute $V_1 \equiv DS_2^{-1} \pmod{q}$ and $V_2 \equiv S_1 S_2^{-1} \pmod{q}$. Verify that $(g^{V_1} A^{V_2} \bmod p) \bmod q = S_1$. |

Table 4.3: The digital signature algorithm (DSA)

| **Public parameter creation** | |
|---|---|
| A trusted party chooses and publishes a (large) prime $p$, an elliptic curve $E$ over $\mathbb{F}_p$, and a point $P$ in $E(\mathbb{F}_p)$. | |
| **Private computations** | |
| **Alice** | **Bob** |
| Chooses a secret integer $n_A$.<br>Computes the point $Q_A = n_A P$. | Chooses a secret integer $n_B$.<br>Computes the point $Q_B = n_B P$. |
| **Public exchange of values** | |
| Alice sends $Q_A$ to Bob $\longrightarrow$ | $Q_A$ |
| $Q_B$ $\longleftarrow$ | Bob sends $Q_B$ to Alice |
| **Further private computations** | |
| **Alice** | **Bob** |
| Computes the point $n_A Q_B$. | Computes the point $n_B Q_A$. |
| The shared secret value is $n_A Q_B = n_A(n_B P) = n_B(n_A P) = n_B Q_A$. | |

Table 6.5: Diffie–Hellman key exchange using elliptic curves

| **Public parameter creation** | |
|---|---|
| A trusted party chooses a finite field $\mathbb{F}_p$, an elliptic curve $E/\mathbb{F}_p$, and a point $G \in E(\mathbb{F}_p)$ of large prime order $q$. | |
| **Samantha** | **Victor** |
| **Key creation** | |
| Choose secret signing key<br>$1 < s < q-1$.<br>Compute $V = sG \in E(\mathbb{F}_p)$.<br>Publish the verification key $V$. | |
| **Signing** | |
| Choose document $d$ mod $q$.<br>Choose random element $e$ mod $q$.<br>Compute $eG \in E(\mathbb{F}_p)$ and then,<br>$s_1 = x(eG) \bmod q$ and<br>$s_2 \equiv (d + s s_1)e^{-1} \pmod{q}$.<br>Publish the signature $(s_1, s_2)$. | |
| **Verification** | |
| | Compute $v_1 \equiv d s_2^{-1} \pmod{q}$ and<br>$v_2 \equiv s_1 s_2^{-1} \pmod{q}$.<br>Compute $v_1 G + v_2 V \in E(\mathbb{F}_p)$ and verify that<br>$x(v_1 G + v_2 V) \bmod q = s_1$. |

Table 6.7: The elliptic curve digital signature algorithm (ECDSA)

| **Public Parameter Creation** | |
|---|---|
| A trusted party chooses and publishes a (large) prime $p$, an elliptic curve $E$ over $\mathbb{F}_p$, and a point $P$ in $E(\mathbb{F}_p)$. | |
| **Alice** | **Bob** |
| **Key Creation** | |
| Chooses a secret multiplier $n_A$.<br>Computes $Q_A = n_A P$.<br>Publishes the public key $Q_A$. | |
| **Encryption** | |
| | Chooses plaintext values $m_1$ and $m_2$ modulo $p$.<br>Chooses a random number $k$.<br>Computes $R = kP$.<br>Computes $S = kQ_A$ and writes it as $S = (x_S, y_S)$.<br>Sets $c_1 \equiv x_S m_1 \pmod{p}$ and $c_2 \equiv y_S m_2 \pmod{p}$.<br>Sends ciphertext $(R, c_1, c_2)$ to Alice. |
| **Decryption** | |
| Computes $T = n_A R$ and writes it as $T = (x_T, y_T)$.<br>Sets $m_1' \equiv x_T^{-1} c_1 \pmod{p}$ and $m_2' \equiv y_T^{-1} c_2 \pmod{p}$.<br>Then $m_1' = m_1$ and $m_2' = m_2$. | |

Table 6.13: Menezes–Vanstone variant of Elgamal (Exercises 6.17, 6.18)

| **Alice** | **Bob** |
|---|---|
| **Key Creation** | |
| Choose a large integer modulus $q$.<br>Choose secret integers $f$ and $g$ with $f < \sqrt{q/2}$, $\sqrt{q/4} < g < \sqrt{q/2}$, and $\gcd(f, qg) = 1$.<br>Compute $h \equiv f^{-1} g \pmod{q}$.<br>Publish the public key $(q, h)$. | |
| **Encryption** | |
| Choose a random $r < \sqrt{q/2}$ | Choose plaintext $m$ with $m < \sqrt{q/4}$.<br>Use Alice's public key $(q, h)$ to compute $e \equiv rh + m \pmod{q}$.<br>Send ciphertext $e$ to Alice. |
| **Decryption** | |
| Compute $a \equiv fe \pmod{q}$ with $0 < a < q$.<br>Compute $b \equiv f^{-1} a \pmod{g}$ with $0 < b < g$.<br>Then $b$ is the plaintext $m$. | |

Table 7.1: A congruential public key cryptosystem

| **Public parameter creation** | |
|---|---|
| A trusted party chooses public parameters $(N, p, q, d)$ with $N$ and $p$ prime, $\gcd(p, q) = \gcd(N, q) = 1$, and $q > (6d+1)p$. | |
| **Alice** | **Bob** |
| **Key creation** | |
| Choose private $\boldsymbol{f} \in \mathcal{T}(d+1, d)$ that is invertible in $R_q$ and $R_p$.<br>Choose private $\boldsymbol{g} \in \mathcal{T}(d, d)$.<br>Compute $\boldsymbol{F}_q$, the inverse of $\boldsymbol{f}$ in $R_q$.<br>Compute $\boldsymbol{F}_p$, the inverse of $\boldsymbol{f}$ in $R_p$.<br>Publish the public key $\boldsymbol{h} = \boldsymbol{F}_q \star \boldsymbol{g}$. | |
| **Encryption** | |
| Should be $c\ell_p(m)$ (centerlift from $R_p$ to $R$) | Choose plaintext $\boldsymbol{m} \in R_p$.<br>Choose a random $\boldsymbol{r} \in \mathcal{T}(d, d)$.<br>Use Alice's public key $\boldsymbol{h}$ to compute $\boldsymbol{e} \equiv p\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \pmod{q}$.<br>Send ciphertext $\boldsymbol{e}$ to Alice. |
| **Decryption** | |
| Compute<br>$\boldsymbol{f} \star \boldsymbol{e} \equiv p\boldsymbol{g} \star \boldsymbol{r} + \boldsymbol{f} \star \boldsymbol{m} \pmod{q}$.<br>Center-lift to $\boldsymbol{a} \in R$ and compute<br>$\boldsymbol{m} \equiv \boldsymbol{F}_p \star \boldsymbol{a} \pmod{p}$. | |

Table 7.4: NTRUEncryt: the NTRU public key cryptosystem

Relevant definitions: (in NTRU)

$R = \mathbb{Z}[x]/(x^N - 1)$ ; elements represented by $N$ coefficients.

$\mathcal{T}(d_1, d_2)$ = elements of $R$ with exactly $d_1$ coefficients equal to 1, $d_2$ coefficients equal to $-1$ & the rest equal to 0.

$R_q = (\mathbb{Z}/q)[x]/(x^N - 1)$