8.28 Example (*ElGamal encryption using the multiplicative group of $\mathbb{F}_{2^m}$, with artificially small parameters*)
Key generation. Entity A selects the group G to be the multiplicative group of the finite field $\mathbb{F}_{2^4}$, whose elements are represented by the polynomials over $\mathbb{F}_2$ of degree less than 4, and where multiplication is performed modulo the irreducible polynomial $f(x) = x^4 + x + 1$ (cf. Example 2.231). For convenience, a field element $a_3x^3 + a_2x^2 + a_1x + a_0$ is represented by the binary string $(a_3a_2a_1a_0)$. The group G has order $n = 15$ and a generator is $\alpha = (0010)$.
A chooses the private key $a = 7$ and computes $\alpha^a = \alpha^7 = (1011)$. A's public key is $\alpha^a = (1011)$ (together with $\alpha = (0010)$ and the polynomial $f(x)$ which defines the multiplication in G, if these parameters are not common to all entities).
Encryption. To encrypt a message $m = (1100)$, B selects a random integer $k = 11$ and computes $\gamma = \alpha^{11} = (1110)$, $(\alpha^a)^{11} = (0100)$, and $\delta = m \cdot (\alpha^a)^{11} = (0101)$. B sends $\gamma = (1110)$ and $\delta = (0101)$ to A.
Decryption. To decrypt, A computes $\gamma^a = (0100)$, $(\gamma^a)^{-1} = (1101)$ and finally recovers m by computing $m = (\gamma^{-a}) \cdot \delta = (1100)$. □

8.5 McEliece public-key encryption

The McEliece public-key encryption scheme is based on error-correcting codes. The idea behind this scheme is to first select a particular code for which an efficient decoding algorithm is known, and then to disguise the code as a general linear code (see Note 12.36). Since the problem of decoding an arbitrary linear code is **NP**-hard (Definition 2.73), a description of the original code can serve as the private key, while a description of the transformed code serves as the public key.

The McEliece encryption scheme (when used with Goppa codes) has resisted cryptanalysis to date. It is also notable as being the first public-key encryption scheme to use randomization in the encryption process. Although very efficient, the McEliece encryption scheme has received little attention in practice because of the very large public keys (see Remark 8.33).

8.29 Algorithm Key generation for McEliece public-key encryption

SUMMARY: each entity creates a public key and a corresponding private key.

1. Integers k, n, and t are fixed as common system parameters.
2. Each entity A should perform steps 3 – 7.
3. Choose a $k \times n$ generator matrix G for a binary (n, k)-linear code which can correct t errors, and for which an efficient decoding algorithm is known. (See Note 12.36.)
4. Select a random $k \times k$ binary non-singular matrix S.
5. Select a random $n \times n$ permutation matrix P.
6. Compute the $k \times n$ matrix $\widehat{G} = SGP$.
7. A's public key is $(\widehat{G}, t)$; A's private key is (S, G, P).

8.30 Algorithm McEliece public-key encryption

SUMMARY: B encrypts a message m for A, which A decrypts.

1. *Encryption.* B should do the following:
   (a) Obtain A's authentic public key $(\widehat{G}, t)$.
   (b) Represent the message as a binary string m of length k.
   (c) Choose a random binary error vector z of length n having at most t 1's.
   (d) Compute the binary vector $c = m\widehat{G} + z$.
   (e) Send the ciphertext c to A.
2. *Decryption.* To recover plaintext m from c, A should do the following:
   (a) Compute $\widehat{c} = cP^{-1}$, where P^{-1} is the inverse of the matrix P.
   (b) Use the decoding algorithm for the code generated by G to decode $\widehat{c}$ to $\widehat{m}$.
   (c) Compute $m = \widehat{m}S^{-1}$.

Proof that decryption works. Since

$$\widehat{c} \;=\; cP^{-1} \;=\; (m\widehat{G} + z)P^{-1} \;=\; (mSGP + z)P^{-1} \;=\; (mS)G + zP^{-1},$$

and zP^{-1} is a vector with at most t 1's, the decoding algorithm for the code generated by G corrects $\widehat{c}$ to $\widehat{m} = mS$. Finally, $\widehat{m}S^{-1} = m$, and, hence, decryption works.

A special type of error-correcting code, called a *Goppa code*, may be used in step 3 of the key generation. For each irreducible polynomial $g(x)$ of degree t over $\mathbb{F}_{2^m}$, there exists a binary Goppa code of length $n = 2^m$ and dimension $k \geq n - mt$ capable of correcting any pattern of t or fewer errors. Furthermore, efficient decoding algorithms are known for such codes.

8.31 Note (*security of McEliece encryption*) There are two basic kinds of attacks known.

(i) From the public information, an adversary may try to compute the key G or a key G' for a Goppa code equivalent to the one with generator matrix G. There is no efficient method known for accomplishing this.

(ii) An adversary may try to recover the plaintext m directly given some ciphertext c. The adversary picks k columns at random from $\widehat{G}$. If $\widehat{G}_k$, c_k and z_k denote the restriction of $\widehat{G}$, c and z, respectively, to these k columns, then $(c_k + z_k) = m\widehat{G}_k$. If $z_k = 0$ and if $\widehat{G}_k$ is non-singular, then m can be recovered by solving the system of equations $c_k = m\widehat{G}_k$. Since the probability that $z_k = 0$, i.e., the selected k bits were not in error, is only $\binom{n-t}{k}/\binom{n}{k}$, the probability of this attack succeeding is negligibly small.

8.32 Note (*recommended parameter sizes*) The original parameters suggested by McEliece were $n = 1024$, $t = 50$, and $k \geq 524$. Based on the security analysis (Note 8.31), an optimum choice of parameters for the Goppa code which maximizes the adversary's work factor appears to be $n = 1024$, $t = 38$, and $k \geq 644$.

8.33 Remark (*McEliece encryption in practice*) Although the encryption and decryption operations are relatively fast, the McEliece scheme suffers from the drawback that the public key is very large. A (less significant) drawback is that there is message expansion by a factor of n/k. For the recommended parameters $n = 1024$, $t = 38$, $k \geq 644$, the public key is about 2^{19} bits in size, while the message expansion factor is about 1.6. For these reasons, the scheme receives little attention in practice.