6.10. Number Theory and Cryptography: A Brief Glimpse

Note. In this section, we very briefly discuss applications of number theory to cryptography. We contrast this with coding theory, and give three examples.

Note. In cryptography, the goal is to disguise a message so that the message is difficult to read. This is opposed to coding theory in which the goal is to encode a message in such a way that it is *easy* to read (and errors can be corrected). Cryptography is discussed in my online notes for Applied Abstract Algebra (not a formal ETSU class); see Chapter 5. These notes also describe coding theory (see Chapter 4). Codes are discussed in my online notes for Applied Combinatorics and Problem Solving (MATH 3340); see Chapter 6. (Several of these notes are still in preparation.)

Note. Cryptography is needed to protect transmitted information from unwanted viewers. For example, cryptography is used in the transmission of military strategy, corporate security, or simply in protecting banking information like your account number. Most cryptographic applications involve encrypting messages using numbers and then scrambling the numbers in some complicated way that requires the recipient to have a key to unscramble the message.

Example 6.10.A. Suppose we want to send a message using our ordinary alphabet of 26 letters. Assign the letters in order to the elements of the set $\{0, 1, 2, \ldots, 25\}$. These are the *plaintext* symbols. Choose an element $a \in \mathbb{Z}_{26}^*$ and an element $b \in \{0, 1, 2, \ldots, 25\}$; see Section 6.8. Primitive Roots and Card Shuffling for details on $\mathbb{Z}_{26}^*$. The values a and b are the keys of the encryption. When the message $y \equiv ax+b \pmod{26}$ is received, the original message x is computed as $x \equiv a^{-1}(y-b) \pmod{26}$. Notice that $a^{-1} \in \mathbb{Z}_{26}^*$ exists; this is why it is necessary to choose a from $\mathbb{Z}_{26}^*$ and not simply from $\mathbb{Z}_{26}$. This is an example of an *affine encryption.*

Note. In *public key cryptography*, a message is encrypted and the extra information beyond a key is required to read the message, with this extra information called a *secret key* which is unavailable to the sender of the message. We now give two examples of public key cryptosystems.

Example 6.10.B. In the *RSA cryptosystem* positive integers m and k, announced publicly, make up the "public key" $K = (m, k)$. Parameter k is chosen to be relatively prime to $\varphi(m)$ and $P = \{0, 1, \ldots, m-1\}$ is used as the plaintext and the "cypertext." The encryption is performed by the function $e_K : P \to P$ where $e_K(x) \equiv x^k \pmod{m}$. The decryption is given by $d_K : P \to P$ where $d_K(y) \equiv y^u \pmod{m}$ and u is a solution to the congruence $ku \equiv 1 \pmod{\varphi(m)}$. With $ku = 1 + t\varphi(m)$ for some t we then have

$$d_K(e_K(x)) \equiv (x^k)^u \equiv x^{ku} \equiv x^{1+t\varphi(m)} \equiv x \cdot (x^{\varphi(m)})^t \equiv x \cdot (1)^t \equiv x \pmod{m},$$

by Euclid's Theorem (Theorem 6.52) and Corollary 6.43. So d_K does in fact decrypt as desired. But if $K = (m, k)$ is public information, then we must depend on some property of m and k that make it difficult to decrypt the message. Large prime numbers are used; m is taken as the product $m = pq$ of two large prime numbers p and q. If p and q are known, then we have $\varphi(m) = \varphi(p)\varphi(q) = (p-1)(q-1)$ by Example 6.50. Then the solution u to the congruence $ku \equiv 1 \pmod{\varphi(m)}$ can be solved, and $d_K(y) \equiv y^u \pmod{m}$ can be applied to decrypt. But the difficulty comes with finding the factorization $m = pq$ if p and q are sufficiently large. (Gerstein says: "Imagine primes with a hundred digits or more." See page 347.) Without the factorization, the computation of $\varphi(m)$ is "out of reach" so that funding u cannot be done. But the factorization is used in order to find $\varphi(m)$. If one could "easily" find $\varphi(m)$, then the decryption would be easy. But notice that if we know that $m = pq$ (but we don't know p or q) and we know $\varphi(m)$, then we have

$$\varphi(m) = (p-1)(q-1) = pq - (p+q) + 1 = m - (p+q) + 1.$$

Since $q = m/p$, this equation becomes $\varphi(m) = m - (p + m/p) + 1$ or $p\varphi(m) = mp - p^2 - m + p$ or $p^2 + (\varphi(m) - m - 1)p + m = 0$. Then solving for p in terms of m and $\varphi(m)$ (which then determines $q = m/p$) we have

$$p = \frac{m - \varphi(m) + 1 \pm \sqrt{(m - \varphi(m) + 1)^2 - 4m}}{2}.$$

But since p (and q) are directly related to m and $\varphi(m)$ by this formula, it is just as difficult to compute p and q directly as it is to compute $\varphi(m)$ directly; these parameters are "computationally equivalent." Research reveals that factoring large numbers is "extraordinarily difficult," so we know that computing $\varphi(m)$ is equivalently difficult. So publicly knowing $K = (m, k)$ is not the valuable/difficult information, but instead the security of the encryption relies on the unknown factors p and q.

Example 6.10.C. A second public key cryptosystem is called the *El Gamal system*, introduced in 1985. The public key is of the form $K = (p, r, \beta)$, where p is a large prime, r is a primitive root mod p, and β is an element of $\mathbb{Z}_p^*$. The decryption requires knowing the discrete logarithm $\log_r \beta$ (while the encryption does not require know this); see Definition 6.79. Without giving details, if β is "chosen appropriately" then finding $\log_r \beta$ can be "extremely difficult" (using Gerstein's terms from page 348).

Revised: 3/4/2022