Solution Sheet 12

Cryptography and Security 2022

Solution 1 Generating Prime Numbers

1. The number of prime numbers in this set is $(2^\ell - 1)P(2^\ell - 1) - 2^{\ell-1}P(2^{\ell-1}) \approx \frac{2^\ell - 1}{\ell \ln 2} - \frac{2^{\ell-1}}{(\ell-1)\ln 2} \approx \frac{2^{\ell-1}}{\ell \ln 2}$. So, the probability is approximately $\frac{1}{\ell \ln 2}$.

2. The complement of this set has no prime numbers. So, all prime numbers are in this set of odd numbers. This means that by picking elements in this set, the probability that it is prime is twice what it was with the complete set. So, the number of trials is divided by two.

3. We know that e is valid for the modulus pq if and only if $\mathsf{gcd}(e, \varphi(pq)) = 1$. This is if and only if $\mathsf{gcd}(3, (p-1)(q-1)) = 1$. This is if and only if $\mathsf{gcd}(3, p-1) = 1$ and $\mathsf{gcd}(3, q-1) = 1$.

   We still have to show that for a prime p, $\mathsf{gcd}(3, p-1) = 1$ is equivalent to $p \bmod 3 = 2$.

   Actually, $p \bmod 3$ cannot be 0 (since p is large and prime). So, $p \bmod 3$ can only be 1 or 2. Clearly, the case of 3 being coprime with $p-1$ can only correspond to the $p \bmod 3 = 2$ case.

   Hence, 3 is valid for the modulus pq if and only if $p \bmod 3 = 2$ and $q \bmod 3 = 2$.

4. GenRSA has to generate some primes p and q such that $p \bmod 3 = 2$ and $q \bmod 3 = 2$, due to the previous question. So, by putting the requirement on the value modulo 3 earlier, we have an algorithm producing equivalent outputs.

   A random prime number is equal to 2 modulo 3 with probability $\frac{1}{2}$. So, in the previous case, we had to iterate 4 times the generation of the two primes to have them both equal to 2 modulo 3. In total, we had to run the prime number generation 8 times.

   With the new algorithm, we iterate the prime number generation twice for each of the prime numbers, so have 4 prime number generations in total.

   This is twice faster.

5. 2 suggests to focus on numbers which are equal to 1 modulo 2. Now, we want to focus on numbers which are equal to 2 modulo 3. Based on the Chinese Remainder Theorem, we could just focus on numbers which are equal to 5 modulo 6. We consider the following algorithm:

   $\mathsf{GenPrime}''(\ell)$
   1: **repeat**
   2: pick x such that $2^{\ell-1}/6 \leq x < 2^\ell/6$ at random
   3: set $p = 6x + 5$
   4: **until** p is prime
   5: output p

   Clearly, is produces equivalent outputs but avoids all values modulo 6 which will make p invalid. So, this new algorithm is 6 times faster than $\mathsf{GenPrime}$.

   When put in GenRSA', this is thus 6 times faster. Since GenRSA' is twice faster than GenRSA, we obtain an algorithm which is 12 times faster than GenRSA.

6. Instead of picking numbers at random, we should make sure that they are not multiples of 5, 7, 11, 13, ... For this, we can pick random numbers in $\mathbf{Z}_5^*$, $\mathbf{Z}_7^*$, $\mathbf{Z}_{11}^*$, $\mathbf{Z}_{13}^*$, ..., combine them with the Chinese Remainder Theorem together with the 5 modulo 6, obtain a suitable residue r modulo $m = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdots$, and take $p = mx + r$ for $x \in [2^{\ell-1}/m, 2^\ell/m[$.

   For instance, if we continue until the prime number 23, the complexity is twice faster than the method of the previous question since

$$\frac{4}{5} \times \frac{6}{7} \times \frac{10}{11} \times \frac{12}{13} \times \frac{16}{17} \times \frac{18}{19} \times \frac{22}{23} \approx 0.49$$

Solution 2 Security Issues in ECDSA

1. k, r, s, $H(M)$ are integers (taken modulo n). (Strictly speaking, $H(M)$ is a bitstring which shall be converted into an integer.) y_1 is a field element. $\mathcal{O}$ is the point at infinity, the neutral element of the elliptic curve.

2. There are two types of finite fields which are popular for ECDSA: the field $\mathbf{Z}_q$ when q is a prime number and the field $\mathsf{GF}(q)$ when q is a power of 2. In the former case, we manipulate integers and reduce them modulo q. In the latter case, we manipulate polynomials with coefficients modulo 2 and reduce them modulo a reference irreducible polynomial.

3. If the key is valid, we have $Q = dG$ and G is on the curve. So, Q lies on the curve. Then, $nQ = n(dG) = d(nG)$. Since G has order n, we have $nG = \mathcal{O}$. Furthermore, $d\mathcal{O} = \mathcal{O}$. So, $nQ = \mathcal{O}$. Then, since $d \in \mathbf{Z}_n^*$, $dG \neq \mathcal{O}$. So, $Q \neq \mathcal{O}$. Q passes all verifications.

   Since r is the result of a modulo n computation, we have $r \in \mathbf{Z}_n$. Since $r = 0$ is excluded from the signature generation and n is prime, we have $r \in \mathbf{Z}_n^*$.

   Finally, we have

$$u_1 G + u_2 Q = \left(\frac{H(M)}{s} G + \frac{r}{s} d \right) G = kG$$

   Due to the signature generation, we know that $(x_1, y_1) = kG$ is such that $r = \bar{x}_1 \bmod n$, so the signature is valid.

4. Because ECDSA uses elliptic curves on which it is hard to compute the discrete logarithm. Computing d given Q and the group material is exactly the problem of computing the discrete logarithm of Q.

5. Let $(x_1, y_1) = kG$. We have $r = \bar{x}_1 \bmod n = r'$ and

$$s = \frac{H(M) + dr}{k} \bmod n \qquad s' = \frac{H(M') + dr}{k} \bmod n$$

   So,

$$\frac{H(M)}{s} + d\frac{r}{s} \equiv \frac{H(M')}{s'} + d\frac{r}{s'} \pmod{n}$$

   thus

$$d = \frac{sH(M') - s'H(M)}{(s' - s)r} \bmod n$$

   which can be computed.