Lecture 35: Coding RSA

Assumption

- We are provided with a One_Rand_Bit() function. It outputs an unbiased independent random bit every time it is invoked.

Generate Random Integer $< 2^t$

Generate a uniformly random integer in the set $\{0, 1, \ldots, 2^t - 1\}$.
Random_Integer(t):

1. Let $m = 0$
2. For $i \in \{1, 2, \ldots, t\}$: $m = (m \ll 1) + \text{One\_Rand\_Bit}()$
3. Return m

Generate Random Integer $< N$

Generate a uniformly random integer in the set $\{0, 1, \ldots, N-1\}$ with probability at least $1 - 2^{-\lambda}$.

Random_Integer(N, λ):

1. Let t be such that $2^{t-1} \leqslant N < 2^t$
2. For $i \in \{1, 2, \ldots, \lambda\}$:
    1. $m = \text{Random\_Integer}(t)$
    2. If $(m < N)$: return m
3. Return -1

Generate a uniformly random integer in the set $\mathbb{Z}_N^*$. If $N = p \cdot q$, where p and q are n-bit primes, then the algorithm succeeds with probability at least $1 - 2^{-\lambda}$.

Random_Zstar(N, λ):

1. Let t be such that $2^{t-1} \leqslant N < 2^t$
2. For $i \in \{1, 2, \ldots, \lambda\}$:
    1. $m = \text{Random\_Integer}(t)$
    2. If $(m < N$ and $\gcd(m, N) == 1)$: return m
3. Return -1

GCD and Extended GCD Algorithms I

- Let us assume that divide(a, b) is a function that takes as input two integers a and b, and outputs (m, r), such that $m = \lfloor a/b \rfloor$ and $r = a - m \cdot b$
- Given this algorithm, let us write down the code of GCD algorithm

GCD(a, b):

1. While $(b \neq 0)$:
    1. $(m, r) = \text{divide}(a, b)$
    2. $a = b$ and $b = r$
2. Return a

Extended GCD algorithm on input (a, b) will output (g, α, β) such that $g = \alpha a + \beta b$ (over integers)

Extended_GCD(a, b):

1. If $(b == 0)$: Return $(a, 1, 0)$
2. $(m, r) = \text{divide}(a, b)$
3. $(g', \alpha', \beta') = \text{Extended\_GCD}(b, r)$
4. Return $(g', \beta', \alpha' - m\beta')$

RSA Encryption I

Gen():

1. $p = \text{Random\_Prime}(n)$
2. $q = \text{Random\_Prime}(n)$
3. Compute $N = p \cdot q$ and $\varphi(N) = (p-1)(q-1)$
4. Pick $e = \text{Random}_{\mathbb{Z}}\text{star}\varphi(N)$ and compute $(g, d, \star) = \text{Extended\_GCD}(e, \varphi(N))$. If $g \neq 1$, then repeat this step
5. Set $\text{pk} = (N, e)$
6. Set $\text{trap} = (\varphi(N), d)$
7. Return (pk, trap)

RSA Encryption II

The choosing of e succeeds with high probability if and only if $\varphi(N)$ does not have too many factors. So, it is recommended that we choose p, q as <u>safe primes</u>

Definition

If both x and $2x+1$ are primes, then x is called the Sophie Germain prime and $2x+1$ is called a Safe prime.

The infinitude and density of these primes are open problems. They are conjectured to be polynomially dense.

RSA Encryption III

$\mathrm{Enc}_{\mathrm{pk}}(m)$:

1. Let $\mathrm{pk} = (N, e)$
2. $r = \mathrm{Random\_Zstar}(N, 100)$
3. If $r = -1$: Set $r = 1$
4. Calculate $y = r^e$
5. $c = m \times y \mod N$
6. Return (y, c)

RSA Encryption IV

$\text{Dec}_{\text{pk,trap}}(c^+)$:

1. Let $c^+ = (y, c)$
2. Let $\text{pk} = (N, e)$
3. Let $\text{trap} = (\varphi(N), d)$
4. Compute $\widetilde{r} = y^d$
5. Compute $(1, \text{inv}(\widetilde{r}), \star) = \text{Extended\_GCD}(\widetilde{r}, N)$
6. Return $c \times \text{inv}(\widetilde{r}) \mod N$