The RSA Cryptosystem

Factoring large integers
Factorization

- Given an integer $N > 1$, it has a unique factorization into primes:
  \[ N = p_1^{r_1} \cdot p_2^{r_2} \cdot \ldots \cdot p_k^{r_k}; \]
- Elementary-school algorithm for factoring:
  - List primes and try each in order
- Inefficient: If $N$ is $k$ bits long, primes $\leq \sqrt{N}$, or less than $k/2$ bits long should be tried.
  - There are $\approx 2^{k/2}/k$ such primes. Exponential workload
RSA Integers

- Let $k = 1024, 1536, 2048$, for instance.
- Generate two primes $p, q$, of bitlength $k/2$.
- Put $N = p \cdot q$.
- Under most conditions, it is hard to factor $N$.
  - The primes $p, q$ must not be of a weak form.
- All known algorithms to factor such numbers have super-polynomial running times.
- Publish $N$; keep $p, q$ secret.
Size of the RSA ring

- Let $Z_N^*$ be the set of integers $a$ such that:
  - $a$ is in $[1, N-1]$;
  - $a$ relatively prime to $N$:
    - $a$ and $N$ have no common divisors;
- Then $Z_N^*$ contains $(p-1)(q-1)$ elements.
  - For each integer $L$:
    - let $\phi(L)$ be the size of the subset of $[1, L-1]$ made of numbers relatively prime to $L$.
  - We have seen that if $N = pq$, a product of primes:
    - $\phi(N) = (p - 1)(q - 1)$
Exponentiation modulo $N$

- If $a$ is relatively prime to $\phi(N)$, then:
  - Let $b$ be such that $ab + k \phi(L) = 1$.
  - Such $b$ exists by the Euclidean algorithm for GCD.
- It follows that for every $M$ in $\mathbb{Z}_N^*$:
  - $C = M^a \mod N$
  - $D = C^b \mod N$
  - Then $D = M \mod N$ (!!!)
Inverting exponents

- If $p$, $q$ are known:
  - First compute $\phi(N)$
  - Then use GCD algorithm to find, for each exponent $a$, its inverse.

- If $p$ and $q$ are not known:
  - Given an exponent, one cannot find its inverse without first having to factor $N$.

- Public Key: $(N, e)$

- Private Key: $(p, q, d)$, where $d = e^{-1}$
To encrypt using RSA

- Retrieve the public key \((N, e)\)
- First, encode a block \(B\) of bitlength smaller than \(k = |N|\), as an element \(M\) of \(\mathbb{Z}_N^*\):
  - \(M = \text{Encode}(N, B)\) // This is not an encryption // everybody knows how to decode
  - Then compute \(C = M^e \mod N\).
- Send \(C\) to recipient, who then uses private \(d\):
  - Recovers \(M = C^d \mod N\).
  - Decodes \(B = \text{Decode}(N, M)\).
To sign using RSA

- Compute message digest $H$ of message $M$.
- Encode $H$ as an element $K$ of $\mathbb{Z}_N^*$
  - $K = \text{Encode}(N, H)$
- Use private key $(p, q, d)$:
  - $S = K^d \mod N$.
- To verify signature $S$ on Message $M$
  - Recompute: $H = \text{digest}(M)$
  - $K = \text{Encode}(N, H)$
  - Use public key $(N, e)$ to check: $(K = S^e \mod N)$?