

RSA Encryption

- ▶ Rivest, Shamir, Adleman (1978)
- ▶ Internet encryption scheme that uses two large prime numbers to generate both the public and private keys; could be decrypted via factorization
- ▶ Factoring numbers is the most difficult if a number has just two prime factors of roughly equal length → considered classically **intractable**
 - ▶ Intractable means the only solution is brute-force

Shor's Factoring Algorithm

- ▶ Peter Shor, 1994
- ▶ One of the first quantum algorithms to garner wide-spread interest in quantum computing
- ▶ Currently it is not feasible to solve real factorization problems with Shor's algorithm, but quantum computers are getting larger every year
- ▶ Not only is it an interesting problem, it opens up many security concerns
- ▶ **Note:** This will be our first example of a hybrid classical-quantum algorithm

Modular Arithmetic

▶ Definition:

$$a = b \bmod N \longrightarrow b = qN + a \text{ for some } q$$

▶ **Note that this is similar to but not the same as the % operator in Python.**

Example

▶ $3 \bmod(12)$

▶ $-9 \bmod(12)$

Example

▶ $3 \bmod(12)$

▶ $3 = \underline{\quad}(12) + \underline{\quad}$

▶ $-9 \bmod(12)$

▶ $-9 = \underline{\quad}(12) + \underline{\quad}$

Example

▶ $3 \bmod(12)$

▶ $3 = -1(12) + \underline{\quad}$

▶ $-9 \bmod(12)$

▶ $-9 = -2(12) + \underline{\quad}$

Example

- ▶ $3 \bmod(12)$
 - ▶ $3 = -1(12) + 15$
- ▶ $-9 \bmod(12)$
 - ▶ $-9 = -2(12) + 15$
- ▶ Note there are other options here as well.

Example

- ▶ $3 \bmod(12) = 3$
- ▶ $-9 \bmod(12) = 3$
- ▶ 3 and -9 are **congruent mod 12** since they have the same modulus
- ▶ 3, -9 are in the same **mod 12 equivalence class** since they can be related through mod 12.

Simple Algebraic Rules Hold for Modular Arithmetic

$$(x + y) \bmod N = (x \bmod N + y \bmod N)$$

$$(xy) \bmod N = (x \bmod N)(y \bmod N)$$

Greatest Common Divisor (GCD)

- ▶ Any integer N can be represented as a product of prime numbers, p :

$$N = p_1 p_2 p_3 \cdots p_\epsilon$$

- ▶ Each value, p_i is called a factor of N
- ▶ For two integers their greatest common divisor is the largest prime factor they both share
- ▶ Example: GCD of 15 and 21?
 - ▶ $\text{gcd}(15, 21) = ?$

Factorization

- ▶ Assume that an integer N has only two prime factors (p and q). Then we can write N as:

$$N = pq$$

- ▶ The goal of factorization is to find p and q given N

Factorization of N

- ▶ $N = pq$, for some p and q of similar length is equivalent to stating:

$$x^2 = 1 \pmod{N}$$

$$1 = rN + x^2$$

$$0 = rN + (x^2 - 1)$$

$$(x^2 - 1) = 0 \pmod{N}$$

$$(x + 1)(x - 1) = 0 \pmod{N}$$

- ▶ That is to say that N divides evenly (no remainder) into $(x + 1)(x - 1)$
- ▶ We can then find the prime factors if we compute:
 - ▶ $p = \gcd(N, x+1)$
 - ▶ $q = \gcd(N, x-1)$

But what is x ?

- ▶ Classically we have to iterate overall possible numbers OR pick random numbers
- ▶ Let's assume we brute force pick random prime numbers less than N and see if $x = 0 \pmod{N}$
- ▶ The number of prime numbers less than a given number N is:

$$\frac{N}{\ln(N)}$$

RSA Encryption Again

- ▶ Consider the following values:
 - ▶ $p, q =$ (large) random prime numbers
 - ▶ $n = pq$
 - ▶ $r = (p-1)(q-1)$
 - ▶ $e = 3, 5, 17,$ or 65537
 - ▶ $d = e^{-1} \bmod(r)$
 - ▶ $de = 1 \bmod(r)$
- ▶ Public key (everyone can see):
 - ▶ e and n
- ▶ Private key (only members of the transaction can see):
 - ▶ d

RSA Encryption and Decryption

- ▶ To encrypt some message, m , to pass with RSA encryption (so the rest of the internet can not read it):
 - ▶ $m^e \bmod(n)$
- ▶ To decrypt the encrypted message, c (for ciphertext) (so the receiver of the message can read it):
 - ▶ $c^d \bmod(n)$

Example

- ▶ Let $p = 11$, $q = 3$ (so $n = 33$), $r = (10)(2) = 20$, $e = 3$, and $d = 3^{-1} \bmod(33) = 7$.
- ▶ Let the message to be $e = 7$.

$$m^e \bmod(n) = 7^3 \bmod(33) = 343 \bmod(33) = 13$$

- ▶ The encrypted message to be passed along is 13 (ciphertext)

Example Continued

- ▶ To decrypt on the other side of the transaction:

$$c^d \bmod(n) = 13^7 \bmod(33) = 62748517 \bmod(33) = 7$$

- ▶ The encryption process does not matter, so it can be public. Decryption needs to be private so it is only done by the people who are conducting the transaction
 - ▶ Decryption is based around the private key d , which is based on r , which is based on the prime factors p, q
 - ▶ So if someone could factor n (which is in the public key) they could crack the decryption

Consider the case of an RSA using a 6 bit number for n

- ▶ Not realistic but for demonstration
- ▶ $s = s_0 2^0 + s_1 2^1 + s_2 2^2 + s_3 2^3 + s_4 2^4 + s_5 2^5$
- ▶ Assume s_5 is 1, then the number is at least $2^5 = 32$
- ▶ Then p and q have to be prime numbers smaller than 32 (need to find just one)

$$\frac{32}{\ln(32)} = 9$$

- ▶ Assume $s_0 = s_1 = s_2 = s_3 = s_4 = s_5 = 1$

$$\frac{63}{\ln(63)} = 15$$

Consider the case of an RSA using a 1,024 bit number for n

$$\frac{2^{1024}}{\ln(2^{1024})} = 1.26 \times 10^{305}$$

Consider the case of an RSA using a 4,096 bit number for n

$$\frac{2^{4095}}{\ln(2^{4095})} = 1.84 \times 10^{1229}$$

Modular Exponentiation

- Consider $a = 2$ and $N = 9$. Calculate $a^n = \underline{\hspace{1cm}} \bmod(N)$ for increasing values of n

$$2^0 = \underline{\hspace{1cm}} \bmod(9)$$

$$2^1 = \underline{\hspace{1cm}} \bmod(9)$$

$$2^2 = \underline{\hspace{1cm}} \bmod(9)$$

$$2^3 = \underline{\hspace{1cm}} \bmod(9)$$

$$2^4 = \underline{\hspace{1cm}} \bmod(9)$$

$$2^5 = \underline{\hspace{1cm}} \bmod(9)$$

Modular Exponentiation

- Consider $a = 2$ and $N = 9$. Calculate $a^n = \underline{\hspace{1cm}} \bmod(N)$ for increasing values of n

$$2^0 = 1 \bmod(9)$$

$$2^1 = 2 \bmod(9)$$

$$2^2 = 4 \bmod(9)$$

$$2^3 = 8 \bmod(9)$$

$$2^4 = 7 \bmod(9)$$

$$2^5 = 4 \bmod(9)$$

Modular Exponentiation

- Consider $a = 2$ and $N = 9$. Calculate $a^n = \underline{\hspace{1cm}} \bmod(N)$ for increasing values of n

$$2^0 = 1 \bmod(9) \quad 2^6 = 1 \bmod(9)$$

$$2^1 = 2 \bmod(9) \quad 2^7 = 2 \bmod(9)$$

$$2^2 = 4 \bmod(9) \quad 2^8 = 4 \bmod(9)$$

$$2^3 = 8 \bmod(9) \quad 2^9 = 8 \bmod(9)$$

$$2^4 = 7 \bmod(9) \quad 2^{10} = 7 \bmod(9)$$

$$2^5 = 4 \bmod(9) \quad 2^{11} = 4 \bmod(9)$$

Modular Exponentiation

- Consider $a = 2$ and $N = 9$. Calculate $a^n = \underline{\quad} \text{mod}(N)$ for increasing values of n

$$2^0 = 1 \text{mod}(9) \quad 2^6 = 1 \text{mod}(9) \quad 2^{12} = 1 \text{mod}(9)$$

$$2^1 = 2 \text{mod}(9) \quad 2^7 = 2 \text{mod}(9) \quad 2^{13} = 2 \text{mod}(9)$$

$$2^2 = 4 \text{mod}(9) \quad 2^8 = 4 \text{mod}(9) \quad 2^{14} = 4 \text{mod}(9)$$

$$2^3 = 8 \text{mod}(9) \quad 2^9 = 8 \text{mod}(9) \quad 2^{15} = 8 \text{mod}(9)$$

$$2^4 = 7 \text{mod}(9) \quad 2^{10} = 7 \text{mod}(9) \quad 2^{16} = 7 \text{mod}(9)$$

$$2^5 = 4 \text{mod}(9) \quad 2^{11} = 4 \text{mod}(9) \quad 2^{17} = 4 \text{mod}(9)$$

Modular Exponentiation and Period Finding

- ▶ Consider $a = 2$ and $N = 9$. Calculate $a^n = ___ \text{mod}(N)$ for increasing values of n
- ▶ Repeated pattern: 1, 2, 4, 8, 7, 5
- ▶ The period: $r = 6$
- ▶ **Period finding** is a classically difficult problem BUT:
 - ▶ $p = \text{gcd}(N, x+1) = \text{gcd}(N, a^{\frac{r}{2}}+1)$
 - ▶ $q = \text{gcd}(N, x-1) = \text{gcd}(N, a^{\frac{r}{2}}-1)$
- ▶ Finding the period between N and a will allow us to find the factors of N !

Shor's Algorithm Step 1

- ▶ If N is the number we wish to factorize, choose an value for a where:
 - ▶ $1 < a < N$
 - ▶ $\gcd(a, N) = 1$
 - ▶ a and N share no common factors

Shor's Algorithm Step 2

- ▶ Find the period between a and N , classically hard so use a quantum algorithm! But how?
- ▶ Consider the quantum gate $U_{a,N}$ which will perform modular exponentiation:
 - ▶ $U_{a,N}|x\rangle = |x \bmod(N)\rangle$
 - ▶ Transforms the state $|x\rangle$ to the state $|x \bmod(N)\rangle$
 - ▶ $|x\rangle$ is just a random quantum state, $|x \bmod(N)\rangle$ is still a quantum state that is altered by the $U_{a,N}$ gate
- ▶ **Note 1:** The form and construction of the $U_{a,N}$ gate is beyond the scope of the class. It will be used as a black box gate.
- ▶ **Note 2:** $U_{a,N}$ can be represented as U in the remaining slides for brevity

The Modular Exponentiation Gate

- ▶ Apply the gate multiple times to get powers of a in $\text{mod}(N)$

$$U_{a,N}^0|1\rangle = |1 \bmod(N)\rangle$$

$$U_{a,N}^1|1\rangle = |a \bmod(N)\rangle$$

$$U_{a,N}^2|1\rangle = |a^2 \bmod(N)\rangle$$

$$U_{a,N}^3|1\rangle = |a^3 \bmod(N)\rangle$$

...

$$U_{a,N}^r|1\rangle = |a^r \bmod(N)\rangle = |1 \bmod(N)\rangle$$

Constructing the Statevector

- ▶ Consider the variable $\theta_s = 2\pi is$, where s is an integer, $0 \leq s < r-1$:
- ▶ Then, we can construct the following quantum state:

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{-j\theta_s/r} |a^j \bmod(N)\rangle$$

$$\begin{aligned} |u_s\rangle = & \frac{1}{\sqrt{r}} (e^{-0\theta_s/r} |a^0 \bmod(N)\rangle \\ & + e^{-1\theta_s/r} |a^1 \bmod(N)\rangle + \dots + \\ & e^{-(r-2)\theta_s/r} |a^{r-2} \bmod(N)\rangle \\ & + e^{-(r-1)\theta_s/r} |a^{r-1} \bmod(N)\rangle) \end{aligned}$$

Apply the Modular Exponentiation Gate

$$\begin{aligned}
 U|u_s\rangle &= \frac{1}{\sqrt{r}}(e^{-0\theta_s/r}U|a^0 \bmod(N)\rangle) \\
 &\quad + e^{-1\theta_s/r}U|a^1 \bmod(N)\rangle + \dots \\
 &\quad + e^{-(r-2)\theta_s/r}U|a^{r-2} \bmod(N)\rangle \\
 &\quad + e^{-(r-1)\theta_s/r}U|a^{r-1} \bmod(N)\rangle)
 \end{aligned}$$

Distribute the negative exponent term

$$\begin{aligned} U|u_s\rangle &= e^{\theta_s/r} \frac{1}{\sqrt{r}} (e^{-1\theta_s/r} |a^1 \bmod(N)\rangle \\ &\quad + e^{-2\theta_s/r} |a^2 \bmod(N)\rangle + \dots \\ &\quad + e^{-(r-1)\theta_s/r} |a^{r-1} \bmod(N)\rangle \\ &\quad + e^{-(r)\theta_s/r} |a^r \bmod(N)\rangle) \end{aligned}$$

Redefine the Last Term

- BUT note that $e^{-r\theta_s/r} = e^{-0\theta_s/r}$ and
 $|a^r \bmod(N)\rangle = |1 \bmod(N)\rangle = |a^0 \bmod N\rangle$

$$\begin{aligned}
 U|u_s\rangle &= e^{\theta_s/r} e^{\frac{1}{\sqrt{r}}} (e^{-1\theta_s/r} |a^1 \bmod(N)\rangle \\
 &\quad + e^{-2\theta_s/r} |a^2 \bmod(N)\rangle + \dots \\
 &\quad + e^{-(r-1)\theta_s/r} |a^{r-1} \bmod(N)\rangle \\
 &\quad + e^{-(0)\theta_s/r} |a^0 \bmod(N)\rangle) \\
 &= e^{\theta_s/r} |u_s\rangle
 \end{aligned}$$

Eigenvalue Problem!

$$U|u_s\rangle = e^{\theta_s/r}|u_s\rangle$$

$$U|u_s\rangle = e^{2\pi i(\frac{s}{r})}|u_s\rangle$$

- ▶ Measuring eigenvalues on a quantum computer *longrightrightarrow*
quantum phase estimation!

Constructing the Statevector

- ▶ Remember that $0 \leq s \leq r - 1$ so there are r possible values for $|u_s\rangle$
- ▶ It turns out that the easiest one to construct is a superposition of all r states:

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1 \bmod(N)\rangle = |1\rangle$$

- ▶ So we just need to construct the eigenvector qubit of the QPE algorithm in the $|1\rangle$ state!
 - ▶ **Consequence of the superposition:** Since the eigenvector qubit is in a superposition, the eigenvalue is also in a superposition of all possible s values until measurement \rightarrow the measurement collapses the eigenvalue to one value of s !

Shor's Algorithm Step 2 REVISED

- ▶ Construct a QPE algorithm with the $U_{a,N}$ and measure the eigenvalue. If the eigenvalue is 0 ($s = 0$), then remeasure.
- ▶ The eigenvalue is $e^{2\pi i(\frac{s}{r})}$ which we can convert into $\frac{s}{r}$.

Shor's Algorithm Step 3

- ▶ Convert the decimal approximation of $\frac{s}{r}$ to a fraction.
- ▶ This will be done on a classical computer using a process called **continued fractions**

Continued Fractions

- ▶ Example: Consider the decimal 0.312
- ▶ 0.312 can be written as $\frac{312}{1000}$ which is equivalent to:

$$0.312 = \frac{1}{\frac{1000}{312}}$$

- ▶ But $\frac{1000}{312} = 3 + \frac{1000-936}{312} = 3 + \frac{64}{312} = 3 + \frac{8}{39}$, so

$$0.312 = \frac{1}{3 + \frac{8}{39}}$$

Continued Fractions (Cont.)

- ▶ But that is equivalent to

$$0.312 = \frac{1}{3 + \frac{1}{\frac{39}{8}}}$$

- ▶ And here we have $\frac{39}{8} = 4 + \frac{7}{8}$, so

$$0.312 = \frac{1}{3 + \frac{1}{4 + \frac{7}{8}}}$$

- ▶ But this is the same as

$$0.312 = \frac{1}{3 + \frac{1}{4 + \frac{1}{\frac{8}{7}}}}$$

- ▶ And $\frac{8}{7} = 1 + \frac{1}{7}$, so

Continued Fractions (Cont.)

$$0.312 = \frac{1}{3 + \frac{1}{4 + \frac{1}{1 + \frac{1}{7}}}}$$

* Now we can approximate 0.312 as:

▶ $0.312 \approx \frac{1}{3}(0.333)$

▶ $0.312 \approx \frac{1}{3+1/4}(0.307)$

▶ $0.312 \approx \frac{1}{3+\frac{1}{4+1/1}}(0.312)$

▶ When doing this step with Shor's algorithm, choose the approximation where r is even and less than N .

Shor's Algorithm **Complete**

- ▶ Step 1: To factor a number, N , choose a number a such that $1 < a < N$ and $\gcd(a, N) = 1$
- ▶ Set up a QPE algorithm with the $U_{a, N}$ gate to get a decimal estimation of $\frac{s}{r}$ (**quantum step**)
- ▶ Use continued fractions to convert $\frac{s}{r}$ to a fraction and obtain r (**classical step**)
- ▶ $\gcd(a^{r/2} - 1, N)$ and/or $\gcd(a^{r/2} + 1, N)$ are likely the factors of N . If not, repeat the algorithm.

Shor's Algorithm in Practice

- ▶ To crack RSA encryptions, need a fault tolerant quantum computer with millions of qubits