Lecture 16: Fermat's Little Theorem and RSA Cryptography

Eric Vigoda
Discrete Mathematics for Computer Science

March 9, 2026

16 Fermat's Little Theorem and RSA Cryptography

In the last lecture we learned about multiplicative inverses in modular arithmetic. Today we'll use multiplicative inverses to prove a beautiful and important result known as Fermat's Little Theorem, which we'll utilize to devise the RSA public-key cryptosystem.

16.1 Review of Inverses

What is the integer x where

$$3x \equiv 1 \bmod 16$$

Let $x = 11$ then it's clear that $3 \times 11 \equiv 33 \equiv 1 \bmod 16$; We say that 11 is the multiplicative inverse of 3 modulo 16, and we denote it as:

$$3^{-1} \equiv 11 \bmod 16.$$

In general, for integers a, x and N where $N \geq 1$, if $ax \equiv 1 \bmod N$ then we say x is the multiplicative inverse of a modulo N, and we denote it as:

$$a^{-1} \equiv x \bmod N.$$

When does the inverse exist? And if it exists, how do we find it efficiently? The inverse of a modulo N exists precisely when a and N are relatively prime, meaning

$$a^{-1} \bmod N \text{ exists } \iff \gcd(a, N) = 1.$$

How do we compute their gcd? We use Euclid's algorithm, which is a simple recursive algorithm based on the following fact:

$$\text{for integers } x \geq y \geq 0,\ \gcd(x, y) = \gcd(y, x \bmod y).$$

If the $\gcd(a, N) = 1$ then how do we compute $a^{-1} \bmod N$? We run the Extended-Euclid algorithm which is a slight generalization of Euclid's GCD algorithm; we went through the algorithm in the last lecture.

16.2 Fermat's Little Theorem

Consider a prime number p. Then for all $a \in \{1, \ldots, p-1\}$, $\gcd(a, p) = 1$ since p is prime; hence, $a^{-1} \bmod p$ exists. This is a nice property of primes: for every a where $a \not\equiv 0 \bmod p$, then $a^{-1} \bmod p$ exists. We will see another nice property of primes which is captured in the following theorem known as Fermat's Little Theorem.

Theorem 16.1 (Fermat's Little Theorem)**.** *For prime p, for all a where $a \not\equiv 0 \bmod p$, then:*

$$a^{p-1} \equiv 1 \bmod p$$

Equivalently, for every integer a (even when $a \equiv 0 \bmod p$),

$$a^p \equiv a \pmod{p}.$$

Proof. Fix prime p and a where $a \not\equiv 0 \bmod p$. We can replace a by another number a' where $a \equiv a' \bmod p$ and thus we can assume a is in simplest terms and thus $a \in \{1, \ldots, p-1\}$.

Let

$$S = \{1, \ldots, p-1\}$$

Form an additional set S' by multiplying every element in S by a, this yields the following set:

$$S' = aS \pmod p = \{a \times 1 \bmod p, a \times 2 \bmod p, \ldots, a \times (p-1) \bmod p\}.$$

Both S and S' are sets (so the order of elements does not matter). We claim they have the same elements and thus are the same sets:

Claim 16.2.

$$S = S'$$

Let's assume the claim and then use it to finish the proof of Fermat's Little Theorem, and then we'll go back to prove the claim.

Assuming the claim, then multiplying the elements of S and multiplying the elements of S' will yield the same result (since the order of multiplications doesn't matter), and thus:

$$\begin{aligned}\prod_{i \in S} i &= \prod_{i' \in S'} i' \\ 1 \times 2 \times \cdots \times (p-1) &\equiv a \times 1 \times a \times 2 \times \cdots \times a \times (p-1) \pmod p \\ 1 \times 2 \times \cdots \times (p-1) &\equiv a^{p-1} \times 1 \times 2 \times \cdots \times (p-1) \pmod p\end{aligned}$$

Since p is prime, as we discussed earlier, $1^{-1} \bmod p$ exists, $2^{-1} \bmod p$ exists, and so on up to $(p-1)^{-1} \bmod p$ exists. Thus, we can multiply both sides of the above equation by these inverses and then rearrange the terms to simplify using the fact that $i \times i^{-1} \equiv 1 \bmod p$ for all $i \in \{1, \ldots, p-1\}$:

$$\begin{aligned}1 \times 2 \times \cdots \times (p-1) &\equiv a^{p-1} \times 1 \times 2 \times \cdots \times (p-1) \pmod p \\ 1 \times \cdots \times (p-1) \times 1^{-1} \times \cdots \times (p-1)^{-1} &\equiv a^{p-1} \times 1 \times \cdots \times (p-1) \times 1^{-1} \times \cdots \times (p-1)^{-1} \pmod p \\ (1 \times 1^{-1}) \cdots \times ((p-1) \times (p-1)^{-1}) &\equiv a^{p-1} \times (1 \times 1^{-1}) \times \cdots \times ((p-1) \times (p-1)^{-1}) \pmod p \\ (1) \cdots \times (1) &\equiv a^{p-1} \times (1) \times \cdots \times (1) \pmod p \\ 1 &\equiv a^{p-1} \pmod p,\end{aligned}$$

which proves the theorem. It only remains to prove the claim. □

Proof of Claim 16.2. Let's first show that all of the elements of S' are distinct. We'll do this by contradiction. Suppose for a pair of indices $1 \leq i, j \leq p-1$ where $i \neq j$, the i-th element of S' is the same as the j-th element of S'. In other words, suppose:

$$ai \equiv aj \bmod p$$

Since $a \not\equiv 0 \pmod p$ and p is prime, then $a^{-1} \bmod p$ exists, so let's multiply both sides by this inverse to obtain the following:

$$\begin{aligned}ai &\equiv aj \bmod p \\ a^{-1}ai &\equiv a^{-1}aj \bmod p \\ i &\equiv j \bmod p,\end{aligned}$$

but this contradicts our assumption that $i \neq j$ and $1 \leq i, j \leq p-1$. Thus, the $p-1$ elements of S' are distinct.

Now let's show that the elements of S' are not 0. Suppose $ai \equiv 0 \bmod p$, then we have the following:

$$\begin{aligned}ai &\equiv 0 \bmod p \\ a^{-1}ai &\equiv a^{-1}0 \bmod p \\ i &\equiv 0 \bmod p.\end{aligned}$$

This implies $i \equiv 0 \bmod p$, but since $1 \leq i \leq p-1$, this is impossible.

We've shown that the elements of S' are distinct and they are not 0. The only remaining possible values are $\{1, \ldots, p-1\}$ and since there are $p-1$ elements in S' then $S' = \{1, \ldots, p-1\}$, and hence $S' = S$ as we claimed. □

16.3 Exercises

Exercise 1: Compute $3^{100} \bmod 7$, using Fermat's Little Theorem to simplify the calculations.

Exercise 2: Compute $40^{10001} \bmod 101$

16.4 Application of Fermat's Little Theorem

We've proved Fermat's Little Theorem. Notice the following consequence of Fermat's Little Theorem, which will foreshadow our cryptographic application. Consider a number b where $\gcd(b, p-1) = 1$ so b and $p-1$ are relatively prime. Why are we talking about $p-1$ now, because it's the "magic" exponent in Fermat's Little Theorem.

Since $\gcd(b, p-1) = 1$ then $b^{-1} \bmod (p-1)$ exists, so compute it and let $c \equiv b^{-1} \bmod (p-1)$. Since $bc \equiv 1 \bmod (p-1)$, then $bc = k(p-1)+1$ for some integer k.

Now consider some number m (think of having a message that we converted to binary and that's m). Then we have the following:

$$(m^b)^c \equiv m^{bc} \equiv m^{k(p-1)+1} \equiv (m^{p-1})^k \times m \equiv (1)^k \times m \equiv m \pmod{p},$$

where we used the fact that $m^{p-1} \equiv 1 \bmod p$ by Fermat's Little Theorem.

So if we have a message m that we encrypt using exponent b and then send $y \equiv m^b \bmod p$ then the receiver will decrypt using exponent c and they end up back with the original message $m \equiv y^c \bmod p$. The problem is that if the prime p is publicly available, then everyone can compute $c \equiv b^{-1} \bmod (p-1)$ and thus decrypt the encrypted message. So we're on the right track but we need to conceal the decryption key better; to do this we use a generalization of Fermat's Little Theorem known as Euler's Theorem.

16.5 Euler's Theorem

We will need the following generalization of Fermat's Little Theorem known as Euler's Theorem.

In the proof of Fermat's Little Theorem we considered the set $S = \{1, \ldots, p-1\}$. Notice that every $i \in S$ is relatively prime to p, and hence $i^{-1} \bmod p$ exists which we used in the later proof.

Consider an arbitrary integer $N \geq 1$. Let $\phi(N)$ denote the number of integers between 1 and $N-1$ which are relatively prime to N:

$$\phi(N) = |\{i : 1 \leq i \leq N-1, \gcd(i, N) = 1\}|.$$

This function ϕ is called Euler's totient function. For prime p, note $\phi(p) = p-1$.

Consider primes p and q where $p \neq q$. Let $N = pq$. Then,

$$\phi(pq) = (p-1)(q-1).$$

To see this, consider the set $\{1, \ldots, pq\}$. There are q multiples of p in that set, namely $p, 2p, \ldots, qp$, all of which are not relatively prime to pq since p is a common factor. Similarly, q has p multiples in this set. But we counted the last term pq twice. Hence the number of remaining numbers, all of which are relatively prime to pq, is

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

We can now state Euler's Theorem.

Theorem 16.3 (Euler's Theorem)**.** *For integers a and N with $N \geq 1$ and* $\gcd(a, N) = 1$,

$$a^{\phi(N)} \equiv 1 \pmod{N}.$$

Proof. The proof is almost identical to that of Fermat's Little Theorem.

Let

$$S = \{\, i : 1 \le i \le N-1,\ \gcd(i,N) = 1 \,\}.$$

Thus S contains all integers between 1 and $N-1$ that are relatively prime to N, and $|S| = \phi(N)$.

Define

$$S' = aS \bmod N = \{ai \bmod N : i \in S\}.$$

Since $\gcd(a,N) = 1$, the inverse $a^{-1} \bmod N$ exists. Using this inverse, exactly as in the proof of Claim 16.2 which appears in the proof of Fermat's Little Theorem, we can show that the elements of S' are distinct and are all relatively prime to N; consequently, $S' = S$.

Therefore the product of the elements of the two sets is the same:

$$\prod_{i\in S} i \equiv \prod_{i'\in S'} i' \equiv \prod_{i\in S} ai \equiv a^{|S|} \prod_{i\in S} i \pmod{N}.$$

Since every $i \in S$ is relatively prime to N, each has an inverse modulo N. Multiplying both sides by these inverses allows us to cancel the product. Then using that $|S| = \phi(N)$ we obtain:

$$a^{\phi(N)} \equiv 1 \pmod{N}.$$

□

16.6 Encryption/Decryption Main Idea

Consider a pair of primes p and q where $p \neq q$. Let $N = pq$. Recall, $\phi(N) = (p-1)(q-1)$.

Let m be an integer which corresponds to a message which we'd like to send encrypted. Suppose we have an integer e where $\gcd(e,(p-1)(q-1)) = 1$. Again, $(p-1)(q-1)$ appears because it is the "magic" exponent $\phi(pq)$ in Euler's Theorem. Since e and $(p-1)(q-1)$ are relatively prime, we can compute the inverse:

$$d \equiv e^{-1} \bmod (p-1)(q-1)$$

To encrypt the message m, we compute

$$y \equiv m^e \bmod N,$$

where $N = pq$. Then to decrypt we compute:

$$y^d \bmod N.$$

Since $de \equiv 1 \bmod (p-1)(q-1)$ then $de = k(p-1)(q-1)+1$ for some integer k. Euler's Theorem tells us that $m^{(p-1)(q-1)} \equiv 1 \bmod N$, and hence we have the following:

$$y^d \equiv (m^e)^d \equiv m^{de} \equiv m^{k(p-1)(q-1)+1} \equiv (m^{(p-1)(q-1)})^k \times m \equiv 1^k \times m \equiv m \pmod{N},$$

where the above assumes that $\gcd(m,N) = 1$ so that Euler's Theorem applies. In the case where $\gcd(m,N) > 1$ then the same result holds that $y^d \equiv m \bmod N$, but proving it requires additional work in that case.

In summary, if we take a message m and raise it to the power e and let $y \equiv m^e \bmod N$. Then we raise y to the power d then we get back the original message $m \equiv y^d \bmod N$. This idea forms the mathematical foundation of the RSA cryptosystem.

16.7 RSA Cryptography

The RSA public-key cryptography scheme was devised by Rivest, Shamir, and Adelman in 1979. In the public-key setting, no private communication is needed.

A person say Bob publishes a public key (N,e), and anyone who wants to send an encrypted message to Bob looks up his public key.

What is Bob's public key? Bob first chooses a pair of primes p and q where $p \neq q$. These primes are HUGE meaning they are about 2000 or 4000 bits long. Then Bob computes $N = pq$. Bob also computes an e where $\gcd(e, (p-1)(q-1)) = 1$. How does Bob find such an e? By trying $e = 3, 5, 7, 11, \ldots$ until they find one which is relatively prime to $(p-1)(q-1)$. Bob then publishes his public key (N, e).

Note the whole world knows N and e, but only Bob knows how to factor $N = pq$. Thus, only Bob can compute $d \equiv e^{-1} \bmod (p-1)(q-1)$, because only Bob knows $(p-1)(q-1) = N - p - q + 1$.

For someone, such as Alice, to send a message m to Bob, they look up Bob's public-key (N, e). Alice then computes $y \equiv m^e \bmod N$, and then sends the encrypted message y to Bob. Bob, upon receiving y, then computes $m \equiv y^d \bmod N$, which is the original message.

Suppose Eve is eavesdropping and sees the encrypted message y. Eve is unable to compute the original message m unless they can compute d, which requires obtaining $(p-1)(q-1)$, but that requires figuring out p and q from N. That's the security guarantee of the RSA protocol: decrypting an encrypted message requires factoring N into its prime factors p and q.

16.8 RSA Toy Example

Let's work through a small example of RSA. Choose

$$p = 11, \qquad q = 31.$$

Then

$$N = pq = 341$$

and

$$\phi(N) = (p-1)(q-1) = 10 \cdot 30 = 300.$$

Notice that

$$\gcd(3, 300) = 3 > 1 \qquad \text{and} \qquad \gcd(5, 300) = 5 > 1,$$

so neither $e = 3$ nor $e = 5$ can be used as the encryption exponent.

Instead, choose

$$e = 7,$$

since

$$\gcd(7, 300) = 1.$$

Next we compute the decryption exponent d such that

$$ed \equiv 1 \pmod{300}.$$

Since

$$7 \cdot 43 = 301 = 300 + 1,$$

we can take

$$d = 43.$$

Thus Bob's public key is

$$(N, e) = (341, 7),$$

and Bob's private decryption key is

$$d = 43.$$

Suppose Alice wants to send the message

$$m = 42.$$

To encrypt, Alice computes

$$y \equiv m^e \equiv 42^7 \equiv 323 \pmod{341},$$

where Alice computed y using the repeated squaring based algorithm (though $e = 7$ is small enough that it's not necessary in this case). Hence, Alice sends the encrypted message:

$$y = 323.$$

Bob receives $y = 323$ and decrypts by computing

$$y^d \equiv 323^{43} \pmod{341},$$

which Bob computes using the repeated squaring based algorithm (which is necessary in this case since d is large). Since

$$y^d \equiv 323^{43} \equiv 42 \pmod{341},$$

then Bob recovers the original message of 42.