# Mathematical Puzzles, Games and Other Diversions

## Day 22

Derrick Chung

April 15, 2021

# Public Key Cryptography

- Every cryptosystem we've seen so far requires that the two parties have a **shared** key.
- But how do you share a key in public between two parties without letting anybody else in on it?
- Going back to the locked box problem, do you remember that Alice wants to send Bob a special package?
  The package isn't the message.
  It's actually their shared key.

Public key cryptography gives you the ability for two parties to communicate securely even if all their communications are public.

# Modular Arithmetic Revisited

## Recall

We showed previously:

Let $a$ be relatively prime to $p$, i.e. $a \not\equiv 0 \pmod{p}$.

Then $a, 2a, 3a, \ldots, (p-1)a$ are also not divisible by $p$ and distinct from each other $\pmod{p}$.

## Corollaries

- This means that mod $p$, the list $\{a, 2a, \ldots, (p-1)\}$ is actually just a permutation or rearrangement of $\{1, 2, \ldots (p-1)\}$.
- We can also conclude that $a$ must have an inverse modulo $p$, since one of $\{a, 2a, \ldots, (p-1)\}$ must be equal to 1.

We'll use both of those facts to prove a very important theorem in mathematics.

# Modular Arithmetic Revisited (cont.)

## Fermat's Little Theorem

If $a$ is not divisible by a prime number $p$, then $a^{p-1} \equiv 1 \pmod{p}$.

## Proof

Let's assume $a$ is a number that's not divisible by $p$.

Consider the sequence $a, 2a, \ldots, (p-1)a$ and its product mod $p$.

We get: $a \times 2a \times 3a \times \cdots \times (p-1)a = a^{p-1}(p-1)!$

But that sequence (mod $p$) is a re-ordering of $1, 2, 3, \ldots, (p-1)$.
So the product also equals $1 \times 2 \times \cdots \times (p-1) = (p-1)! \pmod{p}$

This gives : $a^{p-1}(p-1)! \equiv (p-1)! \pmod{p}$

Now we can cancel the $(p-1)!$ from both sides to get the result.
But this is NOT automatic, since we can only multiply, not divide.

Since every factor of $(p-1)!$ is relatively prime to $p$, so is the product. So we can just multiply both sides by its inverse. $\square$

# Diffie-Hellman Key Exchange

## Basic Description

1. Alice and Bob publicly agree to use a prime $p$ and a residue $g \not\equiv 0 \pmod{p}$.
2. Alice chooses a secret integer $a$, then sends $A = g^a \pmod{p}$ to Bob.
3. Bob chooses a secret integer $b$, then sends $B = g^b \pmod{p}$ to Alice.
4. Alice computes $s = B^a = (g^b)^a \pmod{p}$.
5. Bob computes $s = A^b = (g^a)^b \pmod{p}$.

Alice and Bob now have a shared key $s$ that nobody else knows.

By exponentiation rules, $(g^b)^a = (g^a)^b = g^{ab} \pmod{p}$.

# Diffie Hellman (cont.)

For an interceptor Eve to figure out the secret key, she'd basically have to determine the value of $a$, given $A = g^a \pmod{p}$.
This is known as the discrete logarithm problem.

So long as the original numbers are chosen carefully, this is a very difficult problem (we think).

The security of this cryptosystem hinges on the fact that we don't have an efficient way to solve it.

## Question

What does this system have to with Fermat's Little Theorem?

Answer: Diffie-Hellman requires prime numbers to work properly.

Fermat's Little Theorem helps us find them.

# Generating Prime Numbers

## The Fermat Primality Test

Fermat's Little Theorem tells us that if $a$ is not divisible by a prime number $p$, then $a^{p-1} \equiv 1 \pmod{p}$.

So to test if $p$ is prime, we pick a random number $a$ with $2 < a < p - 2$ and put it to the power of $p - 1$.

If we pick enough random values of $a$ and $a^{p-1} \equiv 1 \pmod{p}$ every time, we say that $p$ is probably prime. Otherwise it's composite.

IT's fast but imperfect, so is used in conjunction with others tests.

## Final Notes

- Almost every public key cryptosystem used today requires large prime numbers to operate well.
- They all rely on the complexity of a particular math problem.
- Public Key Cryptosystems are often usually layered with private key systems for increased security.