6.857 Computer and Network Security September 25, 2001

Lecture Notes 6 : More Number Theory, El Gamal

Lecturer: Ron Rivest *Scribe: Bailey/Cholankeril/Kwon/Zitser*

1 Outline:

- Erratum
- GCD & Modular Inverses
- Order & Generators & Discrete Log Problem
- El Gamal Signatures

2 Erratum

In the previous lecture, there was a small error in the definition of a Carmichael number. The corrected definition is as follows:

Definition (corrected): An integer $n > 1$ is a Carmichael number if,
$a^{n-1} \equiv 1 (\text{mod } n)$
$\forall_a, 1 \leq a < n$, $\underline{\text{s.t. } gcd(a,n) = 1}$

Question: Are there any proofs on the density of Carmichael numbers?
Answer: Yes. There are some bounds on the density of Carmichael numbers. These numbers are very rare, annoying obstacles.

3 GCD, Modular Inverses

Definition 1 *$d \mid a$ ("d divides a") if $\exists k$ s.t. $a = kd$*

Fact 1 *$\forall d, d \mid 0$. This includes $0 \mid 0$. If $a \neq 0$, then $0 \nmid a$*

Definition 2 *A divisor of an integer a is any $d \geq 0$ s.t. $d \mid a$*

Definition 3 *If d is a divisor of a and also of b, then d is a common divisor of a and b.*

Example 1 *7 is a common divisor of 14 and 77.*

Definition 4 *The greatest common divisor, $gcd(a, b)$, of two integers a and b is the largest of their common divisors. (But $gcd(0, 0) = 0$ by definition.)*

$$\begin{aligned} gcd(0,5) &= 5 \\ gcd(24,30) &= 6 \\ gcd(4,7) &= 1 \end{aligned}$$

Question: How are GCD's defined when negative numbers are involved?
Answer: They are defined the same way they are defined for positive numbers.

Question: And what are the divisors of a negative number?
Answer: By the definition of divisibility, $a \mid n$ implies $a \mid -n$, so negative numbers are considered to be divisible by the same numbers their positive counterparts are divisible by.

Definition 5 *Integers a and b are relatively prime if $gcd(a, b) = 1$.*

Fact 2 *If p is prime and $1 \leq a < p$, then $gcd(a, p) = 1$.*

Fact 3 *It is easy to compute $gcd(a, b)$. This is surprising because you might think that in order to compute the GCD of a and b you would need to figure out their divisors, i.e. solve the factoring problem. But, as you will see, we don't need to figure out the divisors of a and b to find their GCD.*

3.1 Euclid's Algorithm

Euclid's Algorithm is probably one of the world's oldest computing algorithms. It allows us to easily calculate the greatest common divisor of any two integers a and b. The algorithm is illustrated below:

Assume $a \geq 0, b \geq 0$

$$gcd(a,b) = \begin{cases} a & \text{if } b = 0 \\ gcd(b,\ a \bmod b) & \text{otherwise} \end{cases}$$

Example 2 *Using Euclid's Algorithm, find the greatest common divisor of 12 and 33.*

$$\begin{aligned} gcd(12,33) &= gcd(33,12) \\ &= gcd(12,9) \\ &= gcd(9,3) \\ &= gcd(3,0) \\ &= 3 \end{aligned}$$

Question: Why does Euclid's algorithm always terminate?
Answer: $a \bmod b$ is always less than b. Hence, on each recursive call, the second argument is strictly less than it was on the previous call.

Theorem 1 *The time to compute $gcd(a, b)$ is $O(log\ b)$.*

Proof: See CLRS, Chapter 31. ■

Intuitive Proof:

In a typical scenario, $gcd(b, a \bmod b)$ is about $b/2$. If we imagine b to be to be expressed in bits, this is equivalent to taking one bit off of b. So the order of execution will be roughly $\log b$. The actual worst case is for a pair of fibonnaci numbers; they decrease by the golden ration on each iteration.

Theorem 2 $(\forall a, b), \exists x, y$ *s.t.* $ax + by = gcd(a, b)$ *where* x, y *are integers.*

Proof: By example, (Euclid's Extended Algorithm)

$gcd(12, 33)$	$33 = 1 * 33$
$gcd(33, 12)$	$12 = 1 * 12$
$gcd(12, 9)$	$9 = 1 * 33 - 2 * 12$
$gcd(9, 3)$	$3 = 3 * 12 - 1 * 33$

3 and -1 are the values of x and y that satisfy the statement: $\forall_{a,b}$ it is true that $gcd(a, b) = ax + by$ for some pair of integers x, y.

Corollary 1 *It is easy to find such x and y.* The method used to find x and y s.t. $ax+by = gcd(a, b)$ is called *Euclid's Extended Algorithm.*

Corollary 2 *Given prime p and a where $1 \leq a < p$, it is easy to find an x s.t. $ax \equiv 1(mod\ p)$ [i.e. $x = a^{-1}(mod\ p)$]. Or equivalently, $ax + py = 1$*

Fact 4 *The above works even if p is not prime, as long as $gcd(a, p) = 1$.*

4 Orders & Generators & DLP

By Fermat's Theorem, $a^{p-1} \equiv 1(\text{mod } p)$ if p is prime and $a \not\equiv 0(\text{mod } p)$.

Definition 6 *The least positive x s.t. $a^x \equiv 1(mod\ p)$ is called the order of a, mod p.*

Theorem 3 - Lagrange
The order of any element a, modulo p (where p is prime and $a \not\equiv 0 mod\ p$) is a divisor of $p - 1$.

Proof: See CLRS, Chapter 31.

Example 3 *Calculate the orders of various elements modulo 7*
p = 7 and p-1 = 6. The divisors of 6 are 1, 2, 3, 6. So all of the numbers in Z_7^* must have order 1, 2, 3, or 6 modulo 7.

$a \in Z_p^*$	a^2	a^3	a^4	a^5	a^6	$order(a)$
6	$6^2 = 1$	$6^3 = 6$	1	6	1	2
2	4	1	2	4	1	3
3	2	6	4	5	1	6

Definition 7 *g is a generator of Z_p^* if the order of g mod p is equal to $p-1$.*

Question: Can generators of groups be even?
Answer: Yes. But there aren't any modulo 7. If you play around with primes other than 7, you should be able to find even generators.

Theorem 4 *If p is prime, then $\exists g$ s.t. g is a generator mod p.*

Fact 5 *If p is prime and g is a generator mod p, then for every y in Z_p^* (i.e. in $\{1, 2, \ldots, p-1\}$) $\exists$ a unique $x (0 \leq x < p-1)$ s.t. $g^x \equiv y (mod\ p)$*

Definition 8 *In the above theorem, x is called the discrete logarithm of y modulo p, base g*

Theorem 5 *If p is prime, then g is a generator mod p iff $g^{(p-1)/q} \not\equiv 1 (mod\ p)$ for every prime q dividing $p-1$*

Question: How many generators exist for Z_p^*?
Answer: Enough to sample and find one efficiently. It's like finding a prime. We need to test whether a candidate number is a generator.

Question: How do we find generators for numbers mod a large prime? Does this require knowing ALL of the prime factors of $p-1$?
Answer: Rather than trying to find all q for a prime p to determine the generator g, we can take a different approach and pick our prime p, s.t. the factorization of $p-1$ is known, allowing us to easily find the generator g.

Idea: Let factorization of $p-1$ be known (e.g., $p-1 = 2 * q$, where q is prime).
Pick g at random. Test $g^{(p-1)/2} \not\equiv 1 (\text{mod } p)$ &
$g^{(p-1)/q} \not\equiv 1 (\text{mod } p) \rightarrow g$ is a generator mod p,
otherwise pick another g.
There are lots of generators, so this works. Yields p, g where p is prime and g is a generator mod p.

4.1 Discrete Logarithm Problem

Given a prime p
a generator g mod p
a value $y \in Z_p^*$
Find
x s.t. $y = g^x (\text{mod } p)$

The discrete logarithm problem is believed to be computationally infeasible if p is large (e.g., 1024 bits) and $p-1$ has a large prime factor. It is as hard as trying to factor a 1024-bit number. This is useful for cryptography because we like to make the hard problem the adversary's problem.

Question: Are the discrete logarithm problem and the factoring problem equally hard in the sense that a problem of one type can be reduced to a problem of the other type?
Answer: No. They are closely related problems, but in the usual formulations no reductions exist. (But taking logs modulo a composite can help factor that composite.)

Question: Doesn't research in the area of discrete logarithms always contribute to solving the factoring problem, therefore making the discrete logarithm problem harder?
Answer: I'm not sure I understand this question. But these problems are closely related, and advances on one problem have usually been translatable into advances in the other.

$x \rightarrow f(x) = g^x (\text{mod } p)$ is a one-way function.

5 El Gamal Signature Scheme

Keygen: generate a prime p (1024 bits)
generator g of Z_p^*
$x \in_R \{0, 1, \ldots, p-2\}$
$y = g^x (\text{mod } p)$
$PK = (p, g, y)$
$SK = (x)$

Question: Is it okay if we take the first n primes, multiply them all together and add or subtract 1 to get a prime number?
Answer: Those primes are bad for cryptography. If all the prime factors of $p-1$ are relatively small, lots of cryptographic attacks are possible. Generally, primes p such that $p-1$ has a big prime factor are much better.

Sign(M)**:** (using SK & PK)
$m = h(M)$
h is a collision-resistant hash function
$k \in_R \{1, 2, \ldots, p-2\}$ s.t. $gcd(k, p-1) = 1$
($\in_R$ means choose at random $\rightarrow$ randomized signature scheme)
$r = g^k (\text{mod } p)$
$s = (m - rx)/k (\text{mod } p-1)$
output: $\sigma = (r, s)$

Note: k, r can be computed before the message is seen. In addition, you need a new k and r everytime you sign a message. Otherwise, it will not be secure.

Verify (M, σ, PK)**:**
Output "Ok" if $0 < r < p$

and $y^r r^s \equiv g^m (\text{mod } p)$, where $m = h(M)$
Otherwise, output "Not Ok"

Question: Why does that work?
Answer:
$g^{rx+ks} = g^{rx} g^{ks} \equiv g^m (\text{mod } p)$
$rx + ks \equiv m (\text{mod } p - 1)$
$s \equiv (m - rx)/k (\text{mod } p - 1)$ [if $gcd(k, p-1) = 1$].

Note: The security of the El Gamal signature scheme depends on DLP (otherwise an adversary could find x, and forge), but it is not equivalent to DLP.

Note: The El Gamal signature scheme can also be generalized to many other groups. e.g., elliptic curves, 2x2 matrices, etc.

Question: Is there a standard hash function for El Gamal?
Answer: It will work with any hash function, as long as both parties agree on which hash function is being used.