ENEE 459-C
Computer Security

Random number generation and intro to number theory

UNIVERSITY OF MARYLAND

Randomness is important!

- The keystream in the one-time pad
- The secret key used in ciphers

Pseudo-random Number Generator

- Pseudo-random number generator:
  - A polynomial-time computable function f (x) that expands a short random string x into a long string f (x) that appears random
- Not truly random in that:
  - Deterministic algorithm
  - Dependent on initial values
- Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin." – John von Neumann
- Objectives
  - Fast
  - Secure

Pseudo-random Number Generator

- Classical PRNGs
  - Linear Congruential Generator
- Cryptographically Secure PRNGs
  - Blum-Micali Generator

Linear Congruential Generator - Algorithm

- Based on the linear recurrence:
  $x_i = a\,x_{i-1} + b \mod m \qquad i \geq 1$

  Where

  x_0 is the seed or start value

  a is the multiplier

  b is the increment

  m is the modulus

  Output

  $(x_1, x_2, \ldots, x_k)$

  $y_i = x_i \mod 2$

  $Y = (y_1 y_2 \ldots y_k)$ ← pseudo-random sequence of K bits

Linear Congruential Generator - Example

- Let $x_n = 3 x_{n-1} + 5 \bmod 31 \quad n \geq 1$, and $x_0 = 2$
  - 3 and 31 are relatively prime, one-to-one (affine cipher)
  - 31 is prime, order is 30

- Then we have the 30 residues in a cycle:
  - 2, 11, 7, 26, 21, 6, 23, 12, 10, 4, 17, 25, 18, 28, 27, 24, 15, 19, 0, 5, 20, 3, 14, 16, 22, 9, 1, 8, 29, 30

- Pseudo-random sequences of 10 bits
  - when $x_0 = 2$
    01101010001
  - When $x_0 = 3$
    10001101001

Linear Congruential Generator - Security

- Fast, but insecure
  - Sensitive to the choice of parameters a, b, and m
  - Serial correlation between successive values
  - Short period, often $m=2^{32}$ or $m=2^{64}$

Linear Congruential Generator - Application

- Used commonly in compilers
  - Rand()
- Not suitable for high-quality randomness applications
- Not suitable for cryptographic applications
  - Use cryptographically secure pseudo-random number generators

Multiplicative group

- A set of elements where multiplication is defined
- It is a closed set
- Every element has an inverse
- Example:
  - $Z^*_7 = \{1,2,3,4,5,6\} \pmod 7$
  - $Z^*_{10} = \{1,3,7,9\} \pmod{10}$
- Find inverses

Order of a multiplicative group

- ***Order of a group: Number of elements contained in the group***
- ***What is the order of Z*p={1,2,…,p-1}***
- The multiplicative group for Z_n, denoted with Z^*_n, is the subset of elements of Z_n relatively prime with n
- The totient function of n, denoted with $\phi(n)$, is the size of Z^*_n
- For a generator of a group g, it is: $g^{\phi(n)} = 1 \bmod N$
- Also all elements in the group can be written as g^i
- If N = pq (p and q are primes), $\varphi(N) = (p-1)(q-1)$
- Also, if p prime phi(p)=p-1
- Difficult problem: Given N, find p and q or $\varphi(N)$
- Example

$$Z^*_{10} = \{ 1, 3, 7, 9 \} \qquad \phi(10) = 4$$

- If p is prime, we have

$$Z^*_p = \{1, 2, \ldots, (p-1)\} \qquad \phi(p) = p - 1$$

Fermat's Little Theorem

Theorem

Let p be a prime. For each nonzero x of Z_p, we have $x^{p-1} \bmod p = 1$

- Example ($p = 5$):

$1^4 \bmod 5 = 1$ $\quad$ $2^4 \bmod 5 = 16 \bmod 5 = 1$

$3^4 \bmod 5 = 81 \bmod 5 = 1$ $\quad$ $4^4 \bmod 5 = 256 \bmod 5 = 1$

Corollary

Let p be a prime. For each nonzero residue x of Z_p, the multiplicative inverse of x is $x^{p-2} \bmod p$

Proof

$$x(x^{p-2} \bmod p) \bmod p = xx^{p-2} \bmod p = x^{p-1} \bmod p = 1$$

Euler's Theorem

Euler's Theorem

For each element x of Z^*_n, we have $x^{\phi(n)} \bmod n = 1$

- Example ($n = 10$)

  $3^{\phi(10)} \bmod 10 = 3^4 \bmod 10 = 81 \bmod 10 = 1$

  $7^{\phi(10)} \bmod 10 = 7^4 \bmod 10 = 2401 \bmod 10 = 1$

  $9^{\phi(10)} \bmod 10 = 9^4 \bmod 10 = 6561 \bmod 10 = 1$

Computing in the exponent

- For the multiplicative group Z^*_n, we can compute in the exponent modulo $\phi(n)$
- Corollary: For Z^*_p, we can compute in the exponent modulo **p-1**
- Example

  $Z^*_{10} = \{ 1, 3, 7, 9 \}$ $\phi(10) = 4$

  3 ^ 1590 mod 10 = 3 ^ (1590 mod 4) mod 10 = 3^2 mod 10 = 9

  How about 2^8 mod 10?
- Example for p=19

  $Z^*_p = \{1, 2, \ldots, (p-1)\}$ $\phi(p) = p - 1$

  15^39 mod 19 = 15^(39 mod 18) mod 19 = 15^3 mod 19= 12

Cryptographically Secure

- Passing the next-bit test
  - Given the first k bits of a string generated by PRBG, there is no polynomial-time algorithm that can correctly predict the next $(k+1)^{th}$ bit with probability significantly greater than ½
  - Next-bit unpredictable

Blum-Micali Generator - Security

- Blum-Micali Generator is provably secure
  - It is difficult to predict the next bit in the sequence given the previous bits, assuming it is difficult to invert the discrete logarithm function (by reduction)

Blum-Micali Generator - Concept

- Discrete logarithm
  - Let p be an odd prime, then $(Z_p^*, \cdot)$ is a cyclic group with order p-1
  - Let g be a generator of the group, then $|\langle g \rangle| = p-1$, and for any element a in the group , we have $g^k = a \bmod p$ for some integer k
  - If we know k, it is easy to compute a
  - However, the inverse is hard to compute, that is, if we know a, it is hard to compute $k = \log_g a$

- Example
  - $(Z_{17}^*, \cdot)$ is a cyclic group with order 16, 3 is the generator of the group and $3^{16} = 1 \bmod 17$
  - Let k=4, $3^4 = 13 \bmod 17$, which is easy to compute
  - The inverse: $3^k = 13 \bmod 17$, what is k? what about large p?

Blum-Micali Generator - Algorithm

- Based on the discrete logarithm one-way function:
  - Let p be an odd prime, then $(Z_p^*, \cdot)$ is a cyclic group
  - Let g be a generator of the group, then for any element a, we have $g^k = a \bmod p$ for some k
  - Let x_0 be a seed

$$x_i = g^{x_{i-1}} \bmod p \qquad i \geq 1$$

Output

$(x_1, x_2, \ldots, x_k)$

$y_i = 1$ if $x_i \geq (p-1)/2$

$y_i = 0$ otherwise

$Y = (y_1 y_2 \ldots y_k)$ ← pseudo-random sequence of K bits

Euclid's GCD Algorithm

- Euclid's algorithm for computing the GCD repeatedly applies the formula

  $\gcd(a, b) = \gcd(b, a \bmod b)$

- Example
  - $\gcd(412, 260) = 4$

Algorithm *EuclidGCD*(*a*, *b*)
  Input integers *a* and *b*
  Output gcd(*a*, *b*)

  if *b* = 0
    return *a*
  else
    return *EuclidGCD*(*b*, *a* mod *b*)

a	412	260	152	108	44	20	4
b	260	152	108	44	20	4	0

Proof of correctness

```
Algorithm EuclidGCD(a, b)
   Input integers a and b
   Output gcd(a, b)

   if b = 0
      return a
   else
      return EuclidGCD(b, a mod b)
```

- We need to prove that GCD(**a**,**b**)=GCD(**b**,**a** mod **b**)
- FACTS
  - Every divisor of **a** and **b** is a divisor of **b** and (**a** mod **b**): This is because (**a** mod **b**) can be written as the sum of **a** and a multiple of **b**, i.e., **a** mod **b** = **a** + k**b**, for some integer k.
  - Similarly, every divisor of **b** and (**a** mod **b**) is a divisor of **a** and **b**: This is because **a** can be written as the sum of (**a** mod **b**) and a multiple of **b**, i.e., **a** = k**b** + (**a** mod **b**), for some integer k.
  - Therefore the set of all divisors of **a** and **b** is **the same** with the set of all divisors of **b** and (**a** mod **b**). Thus the greatest should also be the same.

Multiplicative Inverses (1)

- The residues modulo a positive integer n are the set

  $$Z_n = \{0, 1, 2, \ldots, (n-1)\}$$

- Let x and y be two elements of Z_n such that

  $$xy \bmod n = 1$$

  We say that y is the multiplicative inverse of x in Z_n and we write $y = x^{-1}$

- Example:
  - Multiplicative inverses of the residues modulo 11

x	0	1	2	3	4	5	6	7	8	9	10
x^{-1}		1	6	4	3	9	2	8	7	5	10

Multiplicative Inverses (2)

Theorem

An element x of Z_n has a multiplicative inverse if and only if x and n are relatively prime

- Example
  - The elements of Z_{10} with a multiplicative inverse are $1, 3, 7, 9$

Corollary

If is p is prime, every nonzero residue in Z_p has a multiplicative inverse

Theorem

A variation of Euclid's GCD algorithm computes the multiplicative inverse of an element x of Z_n or determines that it does not exist

x	0	1	2	3	4	5	6	7	8	9
x^{-1}		1		7				3		9

Extended Euclidean algorithm

Theorem

Given positive integers a and b, let d be the smallest positive integer such that

$$d = ia + jb$$

for some integers i and j.

We have

$$d = \gcd(a,b)$$

- Example
  - $a = 21$
  - $b = 15$
  - $d = 3$
  - $i = 3, j = -4$
  - $3 = 3\cdot 21 + (-4)\cdot 15 = 63 - 60 = 3$

Algorithm *Extended-Euclid*(a, b)
    Input integers a and b
    Output $\gcd(a, b)$, i and j
        such that $ia+jb = \gcd(a,b)$
    if $b = 0$
        return (a,1,0)
    (d′, x′, y′) = *Extended-Euclid*(b, $a \bmod b$)
    (d, x, y) = (d′, y′, x′ - [a/b]y′)
    return (d, x, y)

Computing multiplicative inverses

- Compute the multiplicative inverse of a in Z_b
- Given two numbers **a** and **b**, there exist integers x and y such that
  - x**a + **y**b = gcd(a,b)**
- Can be computed efficiently with the Extended Euclidean algorithm
- To compute the multiplicative inverse of a in Z_b ,use the Extended Euclidean algorithm to compute x and y such that x**a + **y**b = 1**
- Then x the multiplicative inverse of a in Z_b

Powers

- Let p be a prime
- The sequences of successive powers of the elements of Z_p exhibit repeating subsequences
- The sizes of the repeating subsequences and the number of their repetitions are the divisors of $p - 1$
- Example ($p = 7$)

x	x^2	x^3	x^4	x^5	x^6
1	1	1	1	1	1
2	4	1	2	4	1
3	2	6	4	5	1
4	2	1	4	2	1
5	4	6	2	3	1
6	1	6	1	6	1

Review of Secret Key (Symmetric) Cryptography

- Confidentiality
  - block ciphers with encryption modes
- Integrity
  - Message authentication code (keyed hash functions)
- Limitation: sender and receiver must share the same key
  - Needs secure channel for key distribution
  - Impossible for two parties having no prior relationship
  - Needs many keys for n parties to communicate

Concept of Public Key Encryption

- Each party has a pair (K, K^{-1}) of keys:
  - K is the **public** key, and used for encryption
  - K^{-1} is the **private** key, and used for decryption
  - Satisfies $\mathbf{D}_{K^{-1}}[\mathbf{E}_K[M]] = M$
- Knowing the public-key K, it is computationally infeasible to compute the private key K^{-1}
  - Easy to check K, K^{-1} is a pair
- The public-key K may be made publicly available, e.g., in a publicly available directory
  - Many can encrypt, only one can decrypt
- Public-key systems aka *asymmetric* crypto systems

Public Key Cryptography Early History

- Proposed by Diffie and Hellman, documented in "New Directions in Cryptography" (1976)
  1. Public-key encryption schemes
  2. Key distribution systems
     - Diffie-Hellman key agreement protocol
  3. Digital signature

- Public-key encryption was proposed in 1970 in a classified paper by James Ellis
  - paper made public in 1997 by the British Governmental Communications Headquarters

- Concept of digital signature is still originally due to Diffie & Hellman

Public Key Encryption Algorithms

- Almost all public-key encryption algorithms use either number theory and modular arithmetic, or elliptic curves
- RSA
  - based on the hardness of factoring large numbers
- El Gamal
  - Based on the hardness of solving discrete logarithm
  - Use the same idea as Diffie-Hellman key agreement