

Cryptography and Network Security

Pseudo-random Number

Xiang-Yang Li

Pseudo-random Bit Generator

✍ Several applications

✍ Key generation

✍ Some encryption algorithms, or one-time pad

✍ Let $l > k$ be integers

✍ Function $f: Z_2^k \rightarrow Z_2^l$ computable in poly-time

✍ Then f called (k, l) -pseudo-random bit generator

✍ The input $s_0 \in Z_2^k$ is called the seed

✍ Output $f(s_0)$ is called the pseudo-random string

Desired Properties

✍ Three important properties:

✍ Unbiased (uniform distribution):

- ✍ All values of whatever sample size is collected are equiprobable

✍ Unpredictable (independence):

- ✍ It is impossible to predict what the next output will be, given all the previous outputs, but not the internal "hidden" state.

✍ Unreproducible:

- ✍ Two of the same generators, given the same starting conditions, will produce different outputs.

Desired Properties

- ✍ Usually when a person says
 - ✍ A "good" pseudo-random number generator
 - ✍ they mean it is unbiased.
 - ✍ A "true" PRNG
 - ✍ they usually mean it's irreproducible
 - ✍ A "cryptographically strong" PRNG
 - ✍ they mean it's unpredictable
 - ✍ Very rarely they mean it's all three

More Properties



Long period

-  The generator should be of long period

Fast computation

-  The generator should be reasonably fast

Security








-  The generator should be secure
-  What is security level of PRNG?

Security

- ✍ A pseudo-random generator is secure
 - ✍ If there is no polynomial-time algorithm that on the first m output-sequences can predict the $m+1^{th}$ bit with probability greater than 0.5
 - ✍ It is also called the next-bit test

Linear Congruential Generator

Protocol








-  Let M be an integer and a, b less than M
-  Let k be number of bits of M
-  Integer l is between $k+1$ and $M-1$
-  Let s_0 be a seed less than M
-  Define $s_i = as_{i-1} + b \pmod M$
-  Then the i th random bit is $s_i \pmod 2$
-  It is not proved to be secure

Parameter Setting

- ✍ Not all a , b are good and m should be large
- ✍ For example, m is a large prime number
- ✍ For fast computation, usually $m=2^{31}-1$
 - ✍ And b is set to 0 often
- ✍ For this m , there are less than 100 integers a
 - ✍ It generates all numbers less than m
 - ✍ The generated sequences appear to be random
- ✍ One such $a=7^5=16807$
 - ✍ Used in IBM 360 family of computers

RSA Generator

Protocol

-  Let p, q be two $k/2$ bits primes and define $n=pq$
-  Integer b : $\gcd(b, \phi(n))=1$
-  Public: n, b ; Private p, q
-  A seed s_0 with k bits
-  Sequence $s_{i+1}=s_i^b \pmod n$
-  Then the i th random bit is $s_i \pmod 2$
-  It is proved to be secure!








BBS Generator

✍ Blum-Blum-Shub Generator

- ✍ Let p, q be two $k/2$ bits primes and define $n=pq$
- ✍ Here $p=q=3 \pmod 4$
- ✍ Let $QR(n)$ be all quadratic residues modulo n
- ✍ Public: n ; Private p, q
- ✍ A seed s_0 with k bits from $QR(n)$
- ✍ Sequence $s_{i+1}=s_i^2 \pmod n$
- ✍ Then the i th random bit is $s_i \pmod 2$
- ✍ It is proved to be secure!

Discrete Logarithm Generator

Protocol

-  Let p be a k -bit prime,
-  Let g be primitive element modulo p
-  A seed s_0 is any non-zero integer less than p
-  Define $s_{i+1} = g^{s_i} \bmod p$
-  Then the i th random bit is
 -  1 if s_i is larger than $p/2$
 -  0 if s_i is less than $p/2$

DES Based Generator

✍ ANSI X9.17 PRNG (used by PGP,..)

✍ Inputs: two pseudo-random inputs

✍ one is a 64-bit representation of date and time

✍ The other is 64-bit seed values

✍ Keys: three 3DES encryptions using same keys

✍ Output:

✍ a 64-bit pseudorandom number and

✍ A 64-bit seed value for next-round use

ANSI X9.17

