ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Secure Pseudo Random Generators

Τσοτουλίδης Γεώργιος, Α.Ε.Μ.: 166

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Νοέμβριος 2023

TOC

Introduction

Introduction

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

Introduction

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

- Roll dice

Introduction

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

- Roll dice
- Flip coin

Introduction

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

- Roll dice
- Flip coin
- Shuffle playing cards

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

- Roll dice
- Flip coin
- Shuffle playing cards
- etc

Introduction

Random numbers or symbols

Random number generation is a process by which, often by means of a random number generator (RNG), a sequence of numbers or symbols that cannot be reasonably predicted.

Types of practical random data

- Roll dice
- Flip coin
- Shuffle playing cards
- etc

Work and ***time*** consuming

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise
- thermal noise

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise
- thermal noise
- external electromagnetic phenomena

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise
- thermal noise
- external electromagnetic phenomena
- quantum phenomena

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise
- thermal noise
- external electromagnetic phenomena
- quantum phenomena
- cosmic background radiation

True vs pseudo-random numbers

True random number generator

Measures some physical phenomenon that is expected to be random and then compensates for possible biases in the measurement process.

Example sources include measuring

- atmospheric noise
- thermal noise
- external electromagnetic phenomena
- quantum phenomena
- cosmic background radiation
- radioactive decay (ex. https://www.fourmilab.ch/hotbits/)

True vs pseudo-random numbers

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism		

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform		

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence		

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X	

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X periodic, deterministic	

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X periodic, deterministic	$\surd$

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X periodic, deterministic	$\surd$
Efficiency		

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X periodic, deterministic	$\surd$
Efficiency	$\surd$	

True vs pseudo-random numbers

What is a PRNG

A pseudorandom number generator (PRNG), is an *algorithm* for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers.

Characteristic	Pseudo	True
Mechanism	Mathematical	Physical & Mathematical
Uniform	$\surd$	$\surd$
Independence	X periodic, deterministic	$\surd$
Efficiency	$\surd$	X

Linear Congruential Generator - LCG

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$X_0 = 1, a = 2, c = 3, m = 5$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5$$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$$

Linear Congruential Generator - LCG

$X_{n+1} = (a \cdot X_n + c) \mod m$
$X_0, a, c < m$
$values \in \mathbb{Z}^+$

Example:
$X_0 = 1, a = 2, c = 3, m = 5$
$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$
$X_2 = (2 \cdot 0 + 3) \mod 5$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$$
$$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$$
$$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$$
$$X_3 = (2 \cdot 3 + 3) \mod 5$$

Linear Congruential Generator - LCG

$X_{n+1} = (a \cdot X_n + c) \mod m$
$X_0, a, c < m$
$values \in \mathbb{Z}^+$

Example:
$X_0 = 1, a = 2, c = 3, m = 5$
$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$
$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$
$X_3 = (2 \cdot 3 + 3) \mod 5 \Rightarrow X_3 = 9 \mod 5 = 4$

Linear Congruential Generator - LCG

$X_{n+1} = (a \cdot X_n + c) \mod m$
$X_0, a, c < m$
$values \in \mathbb{Z}^+$

Example:
$X_0 = 1, a = 2, c = 3, m = 5$
$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$
$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$
$X_3 = (2 \cdot 3 + 3) \mod 5 \Rightarrow X_3 = 9 \mod 5 = 4$
$X_4 = (2 \cdot 4 + 3) \mod 5$

Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$$
$$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$$
$$X_3 = (2 \cdot 3 + 3) \mod 5 \Rightarrow X_3 = 9 \mod 5 = 4$$
$$X_4 = (2 \cdot 4 + 3) \mod 5 \Rightarrow X_4 = 11 \mod 5 = 1$$

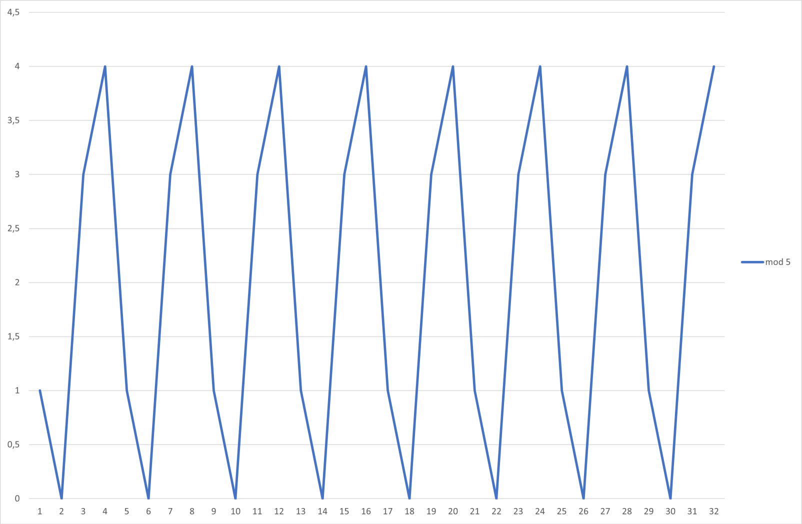
Linear Congruential Generator - LCG

$$X_{n+1} = (a \cdot X_n + c) \mod m$$
$$X_0, a, c < m$$
$$values \in \mathbb{Z}^+$$

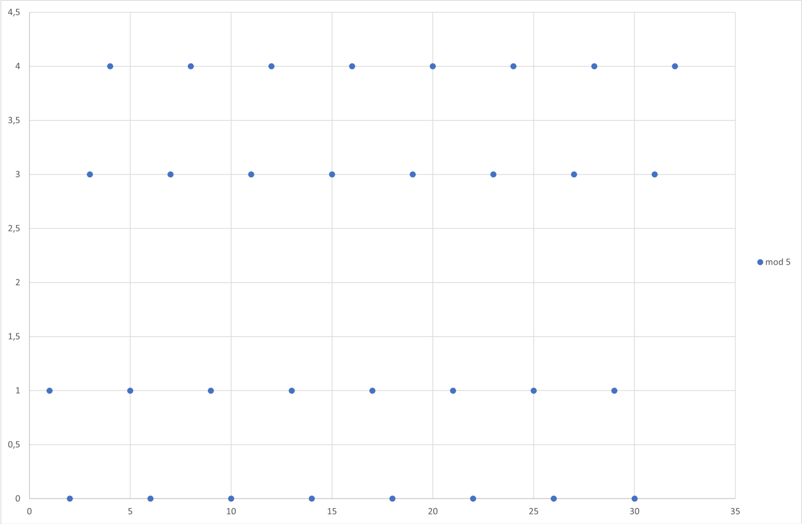
Example:

$$X_0 = 1, a = 2, c = 3, m = 5$$
$$X_1 = (2 \cdot 1 + 3) \mod 5 \Rightarrow X_1 = 5 \mod 5 = 0$$
$$X_2 = (2 \cdot 0 + 3) \mod 5 \Rightarrow X_2 = 3 \mod 5 = 3$$
$$X_3 = (2 \cdot 3 + 3) \mod 5 \Rightarrow X_3 = 9 \mod 5 = 4$$
$$X_4 = (2 \cdot 4 + 3) \mod 5 \Rightarrow X_4 = 11 \mod 5 = 1 \equiv X_0$$

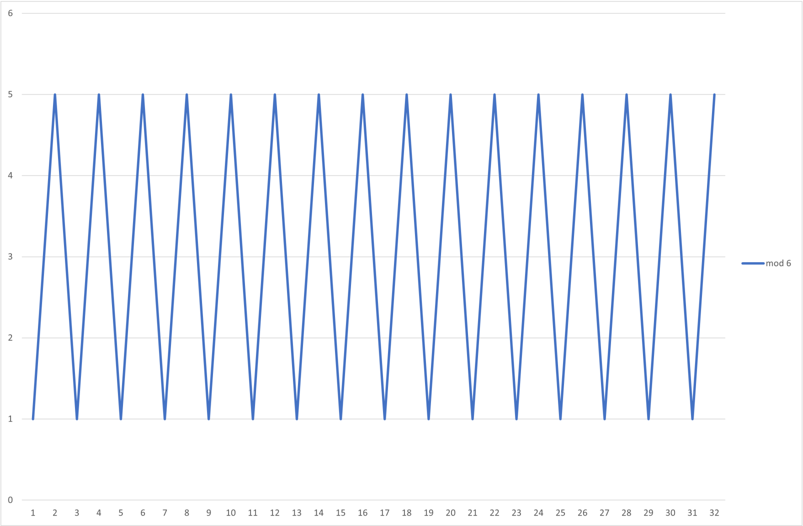
Linear Congruential Generator - LCG - Graph



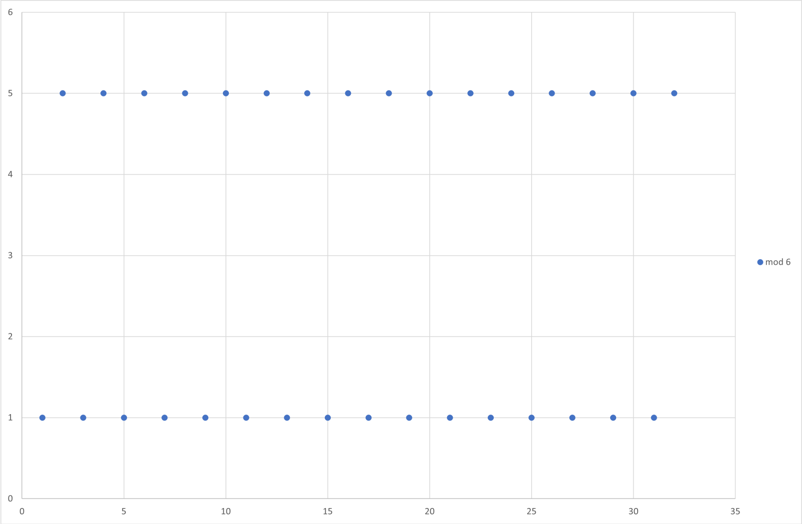
Linear Congruential Generator - LCG - Graph



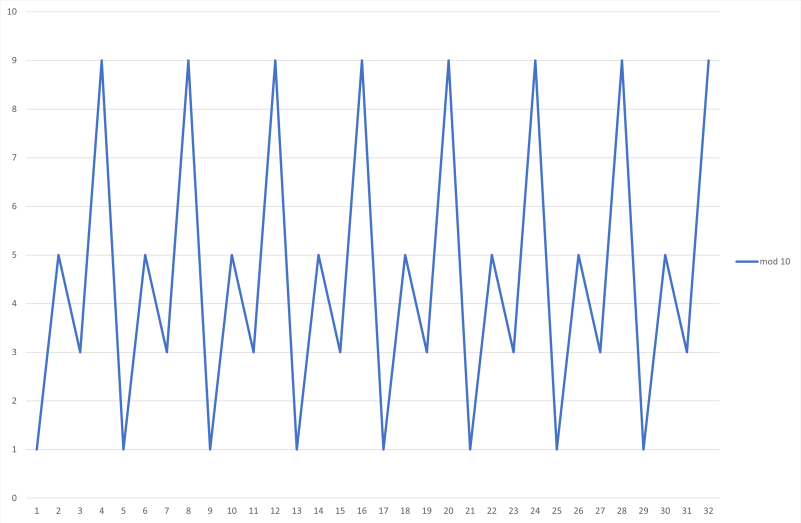
Linear Congruential Generator - LCG - Graph



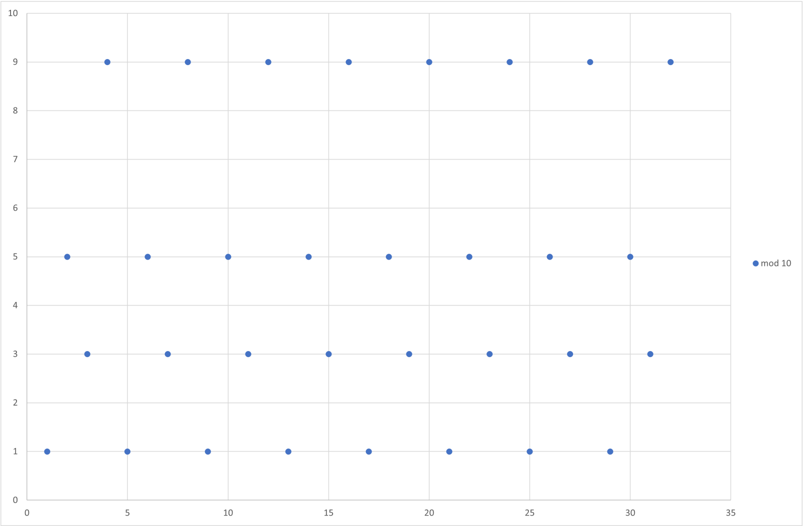
Linear Congruential Generator - LCG - Graph



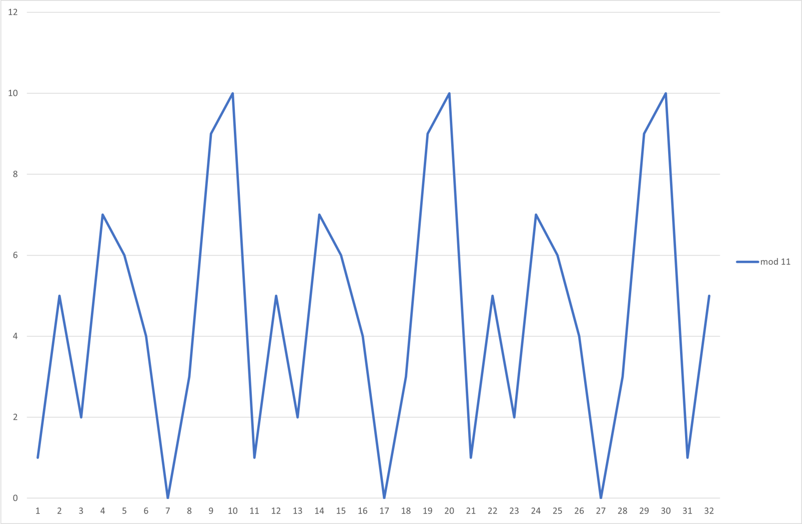
Linear Congruential Generator - LCG - Graph

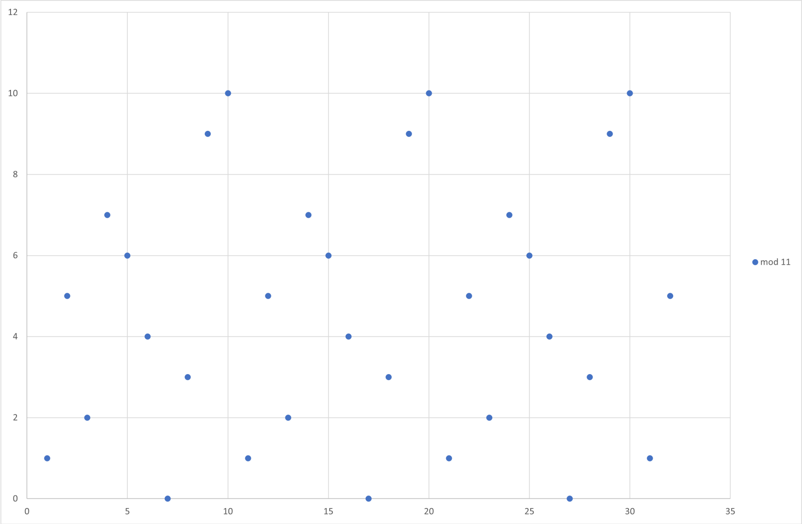


Linear Congruential Generator - LCG - Graph

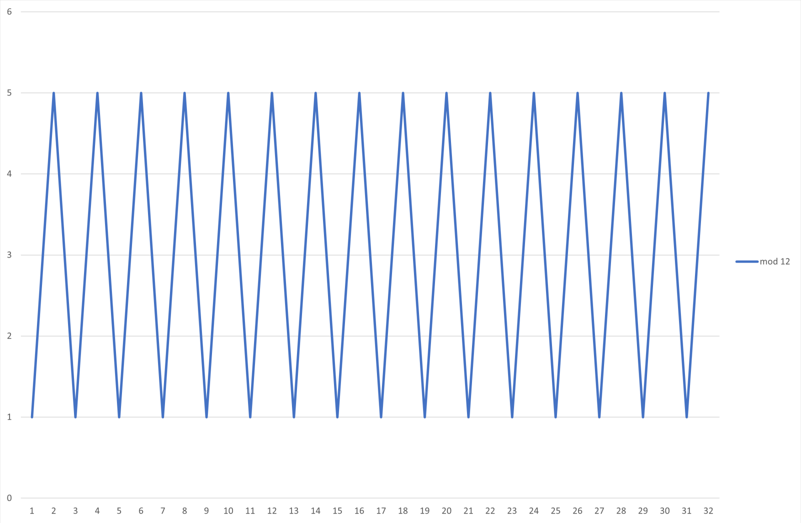


Linear Congruential Generator - LCG - Graph

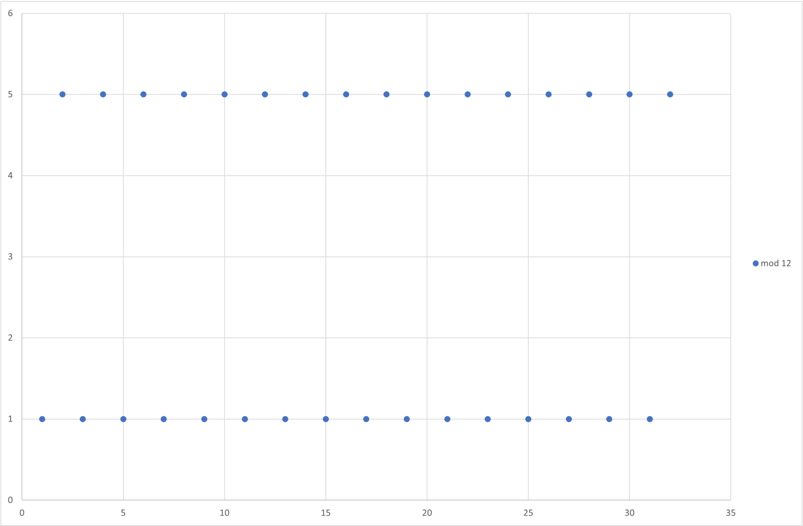



12
10
8
6
4
2
0
0
5
10
15
20
25
30
35
mod 11

Linear Congruential Generator - LCG - Graph



Linear Congruential Generator - LCG - Graph



Linear Congruential Generator - LCG - Table

Source	modulus m	multiplier a	increment c	output bits of seed in *rand()* or *Random(L)*
ZX81	$2^{16}+1$	75	74	
Numerical Recipes from the "quick and dirty generators" list, Chapter 7.1, Eq. 7.1.6 parameters from Knuth and H. W. Lewis	2^{32}	1664525	1013904223	
Borland C/C++	2^{32}	22695477	1	bits 30..16 in *rand()*, 30..0 in *lrand()*
glibc (used by GCC)[17]	2^{31}	1103515245	12345	bits 30..0
ANSI C: Watcom, Digital Mars, CodeWarrior, IBM VisualAge C/C++[18] C90, C99, C11: Suggestion in the ISO/IEC 9899,[19] C17	2^{31}	1103515245	12345	bits 30..16
Borland Delphi, Virtual Pascal	2^{32}	134775813	1	bits 63..32 of *(seed × L)*
Turbo Pascal	2^{32}	134775813 (8088405_{16})	1	
Microsoft Visual/Quick C/C++	2^{32}	214013 ($343FD_{16}$)	2531011 ($269EC3_{16}$)	bits 30..16
Microsoft Visual Basic (6 and earlier)[20]	2^{24}	1140671485 ($43FD43FD_{16}$)	12820163 ($C39EC3_{16}$)	
RtlUniform from Native API[21]	$2^{31}-1$	2147483629 ($7FFFFFED_{16}$)	2147483587 ($7FFFFFC3_{16}$)	
Apple CarbonLib, C++11's `minstd_rand0` ,[22] MATLAB's v4 legacy generator mcg16807[23]	$2^{31}-1$	16807	0	see MINSTD
C++11's `minstd_rand` [22]	$2^{31}-1$	48271	0	see MINSTD
MMIX by Donald Knuth	2^{64}	6364136223846793005	1442695040888963407	
Newlib	2^{64}	6364136223846793005	1	bits 62..32 (46..32 for 16-bit int)
Musl	2^{64}	6364136223846793005	1	bits 63..33
VMS's **MTH$RANDOM**,[24] old versions of glibc	2^{32}	69069 ($10DCD_{16}$)	1	

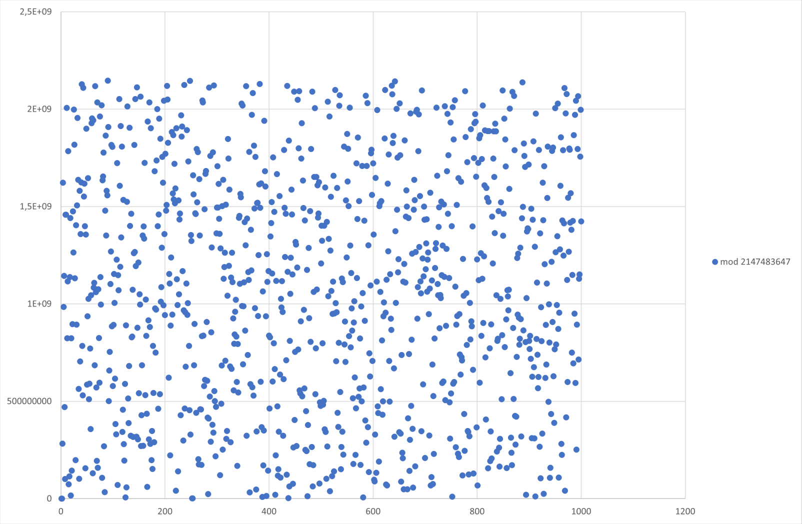
Figure: Parameters of LCGs in common use. Source: Wikipedia

Linear Congruential Generator - LCG - Graph

Linear Congruential Generator - LCG - Graph

$X_0 = 1, a = 16807, c = 0, m = 2^{31} - 1$

$X_0 = 1, a = 16807, c = 0, m = 2^{31} - 1$



Secure Pseudo Random Number Generator

Secure Pseudo Random Number Generator

What is a SPRNG

A cryptographically secure pseudorandom number generator (CSPRNG) or cryptographic pseudorandom number generator (CPRNG) is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography.

Secure Pseudo Random Number Generator

What is a SPRNG

A cryptographically secure pseudorandom number generator (CSPRNG) or cryptographic pseudorandom number generator (CPRNG) is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography.

Requirements to be Cryptographically Secure:

Secure Pseudo Random Number Generator

What is a SPRNG

A cryptographically secure pseudorandom number generator (CSPRNG) or cryptographic pseudorandom number generator (CPRNG) is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography.

Requirements to be Cryptographically Secure:

- Satisfy the next-bit test

Secure Pseudo Random Number Generator

What is a SPRNG

A cryptographically secure pseudorandom number generator (CSPRNG) or cryptographic pseudorandom number generator (CPRNG) is a pseudorandom number generator (PRNG) with properties that make it suitable for use in cryptography.

Requirements to be Cryptographically Secure:

- Satisfy the next-bit test
- Withstand state compromise extension attacks

Secure Pseudo Random Number Generator

Satisfy the next-bit test

That is, given the first k bits of a random sequence, there is no polynomial-time algorithm that can predict the $(k+1)th$ bit with probability of success non-negligibly better than 50%.

Secure Pseudo Random Number Generator

Satisfy the next-bit test

That is, given the first k bits of a random sequence, there is no polynomial-time algorithm that can predict the $(k+1)th$ bit with probability of success non-negligibly better than 50%.

Withstand state compromise extension attacks

In the event that part or all of its state has been revealed (or guessed correctly), it should be impossible to reconstruct the stream of random numbers prior to the revelation. Additionally, if there is an entropy input while running, it should be infeasible to use knowledge of the input's state to predict future conditions of the CSPRNG state.

Differences

Differences

- PRNG is only required to pass certain statistical test

Differences

- PRNG is only required to pass certain statistical test
- CSPRNG must pass all statistical tests that are restricted to polynomial time in the size of the seed

Designs

Designs

CSPRNG designs are divided into three classes

Designs

CSPRNG designs are divided into three classes

- those based on cryptographic primitives such as ciphers and cryptographic hashes

Designs

CSPRNG designs are divided into three classes

- those based on cryptographic primitives such as ciphers and cryptographic hashes

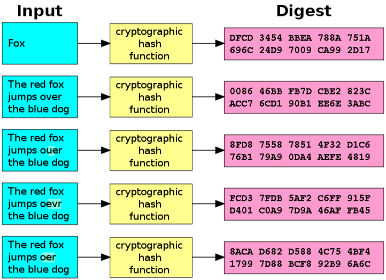


Figure: Hash encryption. Source: Wikipedia

Designs

CSPRNG designs are divided into three classes

- those based on cryptographic primitives such as ciphers and cryptographic hashes

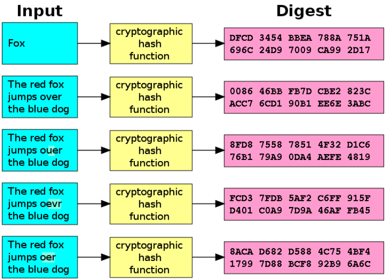


Figure: Hash encryption. Source: Wikipedia

- those based upon mathematical problems thought to be hard

Designs

CSPRNG designs are divided into three classes

- those based on cryptographic primitives such as ciphers and cryptographic hashes

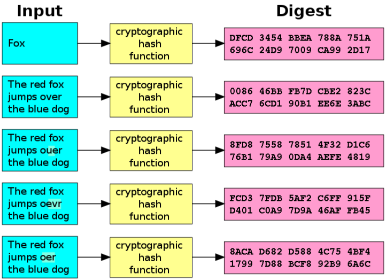


Figure: Hash encryption. Source: Wikipedia

- those based upon mathematical problems thought to be hard
- special-purpose designs

Standards

Standards

- FIPS $186-4$

Standards

- FIPS $186-4$
- NIST SP $800-90A$

Standards

- FIPS $186-4$
- NIST SP $800-90A$
  - $Hash\_DRBG$

Standards

- FIPS $186-4$
- NIST SP $800-90A$
  - *Hash_DRBG*
  - *HMAC_DRBG*

Standards

- FIPS $186-4$
- NIST SP $800-90A$
  - *Hash_DRBG*
  - *HMAC_DRBG*
  - *CTR_DRBG*

Standards

- FIPS $186-4$
- NIST SP $800-90A$
  - *Hash_DRBG*
  - *HMAC_DRBG*
  - *CTR_DRBG*
  - *Dual_EC_DRBG*

Standards

- FIPS $186-4$
- NIST SP $800-90A$
  - *Hash_DRBG*
  - *HMAC_DRBG*
  - *CTR_DRBG*
  - *Dual_EC_DRBG*
- NIST SP $800-90A$ Rev.1

Standards

- FIPS $186 - 4$
- NIST SP $800 - 90A$
  - $Hash\_DRBG$
  - $HMAC\_DRBG$
  - CTR_DRBG
  - $Dual\_EC\_DRBG$
- NIST SP $800 - 90A$ Rev.1
- ANSI $X9.17 - 1985$ Appendix C

Standards

- FIPS 186 – 4
- NIST SP 800 – 90A
    - *Hash_DRBG*
    - *HMAC_DRBG*
    - *CTR_DRBG*
    - *Dual_EC_DRBG*
- NIST SP 800 – 90A Rev.1
- ANSI X9.17 – 1985 Appendix C
- ANSI X9.31 – 1998 Appendix A.2.4

Standards

- FIPS 186 – 4
- NIST SP 800 – 90A
  - *Hash_DRBG*
  - *HMAC_DRBG*
  - *CTR_DRBG*
  - *Dual_EC_DRBG*
- NIST SP 800 – 90A Rev.1
- ANSI X9.17 – 1985 Appendix C
- ANSI X9.31 – 1998 Appendix A.2.4
- ANSI X9.62 – 1998 Annex A.4, obsoleted by ANSI X9.62 – 2005, Annex D (*HMAC_DRBG*)

Cryptographic PRNGs

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks
- mouse moves

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks
- mouse moves
- network activity

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks
- mouse moves
- network activity
- system I/O interruptions

Cryptographic PRNGs

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks
- mouse moves
- network activity
- system I/O interruptions
- hard disk activity

Some classes of CSPRNGs include

- stream ciphers
- block ciphers running in counter or output feedback mode
- combination PRNGs
- special designs based on mathematical hardness

Entropy collection

- keyboard clicks
- mouse moves
- network activity
- system I/O interruptions
- hard disk activity
- etc

Secure Random Generators (CSPRNG)

Secure Random Generators (CSPRNG)

Usually a CSPRNG should start from an unpredictable random seed from the operating system, from a specialized hardware or from external source. Random numbers after the seed initialization are typically produces by a pseudo-random computation, but this does not compromise the security. Most algorithms often "reseed" the CSPRNG random generator when a new entropy comes, to make their work even more unpredictable.

Secure Random Generators (CSPRNG)

Usually a CSPRNG should start from an unpredictable random seed from the operating system, from a specialized hardware or from external source. Random numbers after the seed initialization are typically produces by a pseudo-random computation, but this does not compromise the security. Most algorithms often "reseed" the CSPRNG random generator when a new entropy comes, to make their work even more unpredictable.

Typically modern OS CSPRNG APIs combine the constantly collected entropy from the environment with the internal state of their built-in pseudo-random algorithm with continuous reseeding to guarantee maximal unpredictability of the generated randomness with high speed and non-blocking behavior in the same time.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.
- Encrypts a partition or drive where Windows is installed (pre-boot authentication).

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.
- Encrypts a partition or drive where Windows is installed (pre-boot authentication).
- Encryption is automatic, real-time(on-the-fly) and transparent.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.
- Encrypts a partition or drive where Windows is installed (pre-boot authentication).
- Encryption is automatic, real-time(on-the-fly) and transparent.
- Parallelization and pipelining allow data to be read and written as fast as if the drive was not encrypted.

Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.
- Encrypts a partition or drive where Windows is installed (pre-boot authentication).
- Encryption is automatic, real-time(on-the-fly) and transparent.
- Parallelization and pipelining allow data to be read and written as fast as if the drive was not encrypted.
- Encryption can be hardware-accelerated on modern processors.

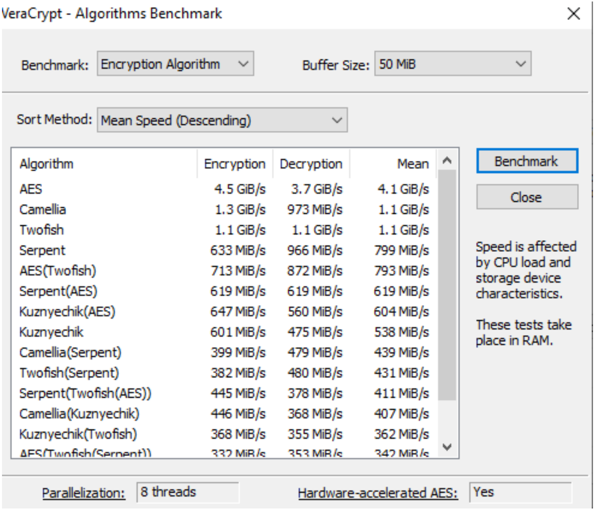
Veracrypt encryption program

VeraCrypt is a free open source disk encryption software for Windows, Mac OSX and Linux based on TrueCrypt 7.1a.

Main features:

- Creates a virtual encrypted disk within a file and mounts it as a real disk.
- Encrypts an entire partition or storage device such as USB flash drive or hard drive.
- Encrypts a partition or drive where Windows is installed (pre-boot authentication).
- Encryption is automatic, real-time(on-the-fly) and transparent.
- Parallelization and pipelining allow data to be read and written as fast as if the drive was not encrypted.
- Encryption can be hardware-accelerated on modern processors.
- Provides plausible deniability, in case an adversary forces you to reveal the password: Hidden volume (steganography) and hidden operating system.

Veracrypt encryption random number generation



Algorithm	Encryption	Decryption	Mean
AES	4.5 GiB/s	3.7 GiB/s	4.1 GiB/s
Camellia	1.3 GiB/s	973 MiB/s	1.1 GiB/s
Twofish	1.1 GiB/s	1.1 GiB/s	1.1 GiB/s
Serpent	633 MiB/s	966 MiB/s	799 MiB/s
AES(Twofish)	713 MiB/s	872 MiB/s	793 MiB/s
Serpent(AES)	619 MiB/s	619 MiB/s	619 MiB/s
Kuznyechik(AES)	647 MiB/s	560 MiB/s	604 MiB/s
Kuznyechik	601 MiB/s	475 MiB/s	538 MiB/s
Camellia(Serpent)	399 MiB/s	479 MiB/s	439 MiB/s
Twofish(Serpent)	382 MiB/s	480 MiB/s	431 MiB/s
Serpent(Twofish(AES))	445 MiB/s	378 MiB/s	411 MiB/s
Camellia(Kuznyechik)	446 MiB/s	368 MiB/s	407 MiB/s
Kuznyechik(Twofish)	368 MiB/s	355 MiB/s	362 MiB/s
AES(Twofish(Serpent))	332 MiB/s	353 MiB/s	342 MiB/s

Figure: Benchmark algorithms

Veracrypt encryption random number generation

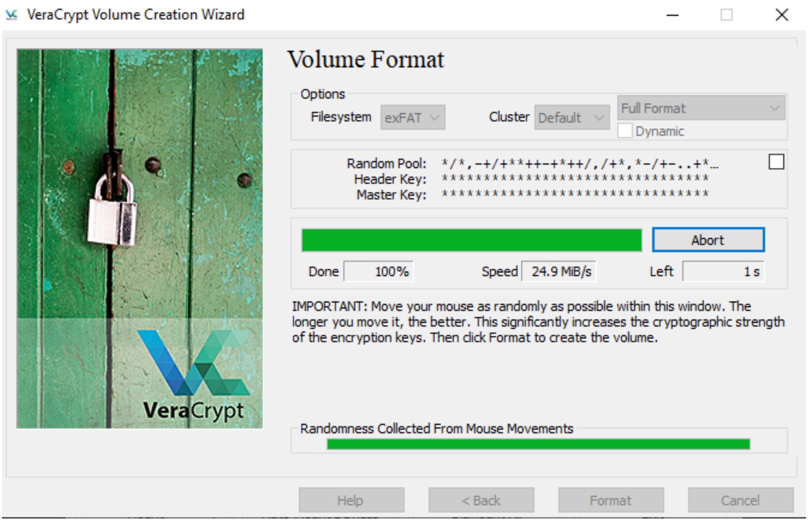


Figure: Random pool

Bibliography

- https://www.random.org/randomness/
- https://cryptobook.nakov.com/secure-random-generators/secure-random-generators-csprng
- https://textbook.cs161.org/crypto/prng.html
- https://veracrypt.fr/en/Random Number Generator.html
- https://www.youtube.com/watch?v=PtEivGPxwAI
- Wikipedia
  - Cryptographically secure pseudorandom number generator
  - Pseudorandom number generator
  - Linear congruential generator

Thank you - QA

Σας ευχαριστώ για την προσοχή και τον χρόνο σας.
Μην διστάσετε να κάνετε οποιαδήποτε ερώτηση.

Σας ευχαριστώ για την προσοχή και τον χρόνο σας.
Μην διστάσετε να κάνετε οποιαδήποτε ερώτηση.