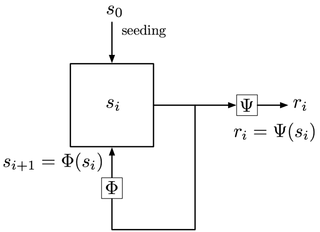
Deterministic Random Number Generators

Çetin Kaya Koç

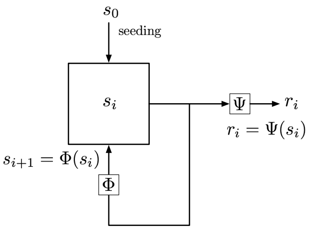
koc@cs.ucsb.edu



DRNGs

- A pure DRNG starts with a seed (s_0) value and using a **seeding algorithm** computes the first internal state s_1 from s_0
- Once s_0 is made available the subsequent internal states s_{i+1} are being produced using the **state transition function** $\Phi(s_i)$
- The output is the random number r_i which is computed using the **output functions** $\Psi(s_i)$ from the internal state s_i

$$\begin{aligned} s_1 &= \text{seeding}(s_0) \\ r_i &= \Psi(s_i) \\ s_{i+1} &= \Phi(s_i) \end{aligned}$$



DRNG Examples

- Linear Feedback Shift Registers (LFSRs)
- Cellular Automata (CA)
- Linear Congruential Generators (LCGs)
- Block cipher and Hash function based methods
- Number-theoretical methods: RSA, Rabin, Blum-Blum-Shub
- Elliptic curve methods: LCG, Power Generator, Naor-Reingold

DRNG Advantages

DRNGs as random number generators have many advantages:

- low cost
- no dedicated hardware is required
- implementations can be done in software
- identical seed values imply identical random numbers which is a necessary condition for using them as stream ciphers

DRNG Disadvantages

However, there are disadvantages:

- For pure DRNGs, the output is completely determined by the seed
- Output sequences of pure DRNGs cannot be truly independent
- They may behave as output sequence of an ideal RNG at most with respect to certain aspects
- The internal state has to be protected even if the device is not active

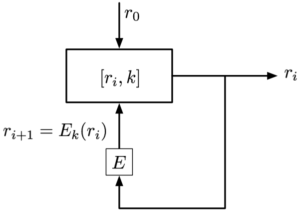
Security Requirements of DRNGs

- LFSRs: usually meet Requirement R1, but not R2
- CA: meet R1 and R2 on certain conditions
- LCGs: usually meet Requirement R1, may meet R2
- Block cipher methods: meet R1 and R2 on certain conditions
- Hash function methods: meet R1 and R2 on certain conditions
- Number-theoretical methods: meet R1 and R2 on certain conditions
- Elliptic curve methods: meet R1 and R2 on certain conditions

- Simple structures, such as LFSRs, are useful for fast and efficient implementations but they may have serious security shortcomings

Block Cipher DRNGs

- The seed r_0
- The initial state $[r_0, k]$
- The key k is to be kept secret
- Internal state is the pair $s_i = [r_i, k]$
- The output r_i
- Next state $s_{i+1} = [E_k(r_i), k]$



Block Cipher DRNGs

- Block cipher based DRNGs meet R1: ciphertext from a strong cipher should not have any statistical weakness
- Block cipher based DRNGs also meet R2: only if the encryption and decryption functions are secure against chosen-plaintext attacks
- This "security proof" is typical for DRNGs: tracing back to the recognized properties of well-known cryptographic primitives
- If the key k is unknown, the computation of the next state $s_{i+1} = [r_{i+1}, k]$ from the previous state $s_i = [r_i, k]$ is equivalent to breaking of the block cipher since $r_{i+1} = E_k(r_i)$

Block Cipher DRNGs

- Similarly, if the key k is unknown, the computation of the previous state $s_i = [r_i, k]$ from the next state $s_{i+1} = [r_{i+1}, k]$ is equivalent to breaking of the block cipher since $r_i = D_k(r_{i+1})$
- Therefore, unpredictability, or Requirement R2, depends on the cryptographic strength of the block cipher
- If the block cipher is secure, then Requirement R2 is satisfied
- For AES and Triple-DES encryption functions, we may assume that this type of DRNG meets R2
- In the 80s, the same conclusion was justified for Single-DES, but this conclusion is no longer valid!

ANSI X9 Standards

- A well-known block cipher based DRNG called X9.17 is standardized for the financial industry
- American National Standards Institute (ANSI) is broken down into committees, one being ANSI X9
- The committee ANSI X9 develops standards for the financial industry, more specifically for personal identification number (PIN) management, check processing, electronic transfer of funds, etc
- Within the committee of X9, there are subcommittees
- The actual documents are named as X9.9 and X9.1, etc

ANSI X9.17

- ANSI X9.17 defines a format for messages to establish new keys and replace old ones called CSM (cryptographic service messages)
- ANSI X9.17 also defines two-key triple-DES encryption as a method by which keys can be distributed
- ANSI X9.17 is gradually being supplemented by public-key techniques such as Diffie-Hellman encryption
- Triple DES with two keys is defined as the function

$$E_k(M) \equiv E_{k_1}(D_{k_2}(E_{k_1}(M))) \quad \text{such that} \quad k = k_1 \,||\, k_2$$

- The symbol $||$ stands for the concatenation operator

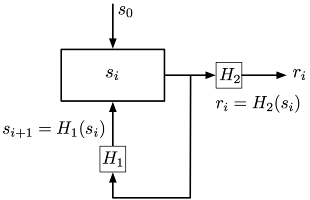
ANSI X9.17 DRNG

- **Input:** A secret 64-bit seed s, 112-bit key k, and an integer m
- **Step 1:** Compute $I = E_k(D)$
  D is the 64-bit representation of the date and time[1]
  D is used as the initializing variable (IV)
- **Step 2:** For i from 1 to m
  - $r_i \leftarrow E_k(I \oplus s)$
  - $s \leftarrow E_k(r_i \oplus I)$
- **Output:** Return $r_1, r_2, \ldots, r_m$

[1] The Unix Time is a 32-bit number

Hash Function DRNGs

- The seed s_0
- s_i is a vector of at least 256 bits
- H_1 and H_2 are two different hash functions
- The output $r_i = H_2(s_i)$
- Next state $s_{i+1} = H_1(s_i)$
- Fulfills R1 and R2



Hash Function DRNGs

- Since $s_{i+1} = H_1(s_i)$, the computation of the next state s_{i+1} from the previous state s_i is trivial
- However, the computation of the previous state s_i from the next state s_{i+1} requires the inversion of the one-way function H_1
- Similarly, the computation of the output r_i from the previous state s_i is also trivial
- However, the computation of the previous state s_i from the output r_i requires the inversion of the one-way function H_2

Security Requirement R4′

- Therefore, the discovery of the current state s_i is detrimental to the security of the hash function DRNG
- Similarly, the discovery of the current state $s_i = [r_i, k]$ would be detrimental to the security of the block cipher DRNG

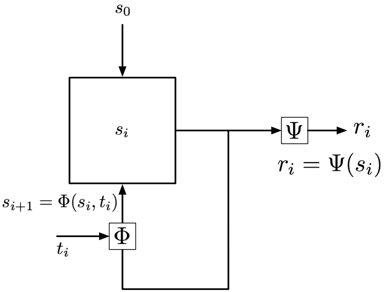
- For specific applications, Requirement R4′, named as **enhanced forward security**, is desirable
- *It should not be practically feasible to compute future random numbers from the internal state or to guess them with non-negligibly larger probability than without the knowledge of the internal state*

Security Requirement R4′

- **Attack Scenario:** An attacker is able to read or to manipulate the internal state of a DRNG without being noticed by the user/owner of the DRNG who uses the subsequent random numbers
- Pure DRNGs cannot fulfill Requirement R4′
- A hybrid DRNG may fulfill R4′ for the random numbers that are generated after the first update of its internal state with random data after the internal state has been compromised

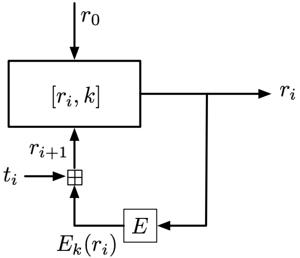
Hybrid DRNGs

- s_0: Seed
- r_i: Random number
- Φ: State transition function
- Ψ: Output function
- t_i: **Additional input**
- t_i: Non-deterministic data, for example, time, system data, random bits from a TRNG



Example: Block Cipher Hybrid DRNGs

- Strong block cipher, such as AES or Triple-DES
- Key k is kept secret
- t_i: **Additional input**
- $r_{i+1} = t_i \oplus E_k(r_i)$
- The algorithmic part guarantees R1 and R2
- Additional input with large entropy may ensure R3 and R4



Hybrid DRNGs

- Additional input may be provided
  - After each step
  - Occasionally
  - Upon external request of an application
- Additional input may have
  - Large entropy per bit, using a strong physical RNG
  - Low entropy, time etc

DRNGs vs TRNGs

- A pure DRNG fulfills Requirement R2 and possibly R3 if the state function and output function are sufficiently complex
- A pure DRNG provides practical (computational) security
- Its security assessment may change in the course of time

- A TRNG fulfills R2 if the entropy of the internal random numbers is sufficiently large
- This implies theoretical (time-invariant) security
- Secure forever :)

Usefulness of Hybrid RNGs

- In practice: coming up with high quality and well controlled noise sources is challenging
- Several proposals have failed over the years, due to entropy properties or due to the use of simple structures for algorithmic post-processing
- We are stuck between a rock and a hard place:
  - DRNG: Security properties may change over time
  - TRNG: Low-cost, high-quality, and high-bandwidth entropy sources that can be integrated on to our chips are scarce
- Hybrid DRNGs: low-cost and low-bandwidth entropy sources, coupled with cryptographically strong and high-bandwidth functions for DRNGs, whose design parameters can be selected and controlled