





# Numeri casuali



- ❑ Importanti per molte primitive crittografiche 
- ❑  un avversario non deve determinarli o indovinarli se non con una bassa probabilità



# Generazione Deterministica?

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

John von Neumann

[1951]




# Randomness by obscurity

- ❑ 2 studenti di Berkeley nel settembre 1995
- ❑ reverse-compilation della routine di Netscape 1.1
- ❑ time, pid, ppid mischiati erano il seme di un generatore per scegliere chiavi e challenge





# Caratteristiche del sistema

- ❑ variabili di sistema, ad esempio il tempo del clock oppure numeri di serie (come l'indirizzo Ethernet) 
- ❑ il numero di file su dischi od in particolare directory
- ❑ lo spazio libero su ogni disco
- ❑ le informazioni presenti nei vari buffer, nelle code per l'I/O, oppure nei driver per video
- ❑ il numero di task nella coda di schedulazione del sistema operativo, le loro ID, le loro grandezze
- ❑ oppure lo stato della memoria centrale
- ❑ informazioni definite dall'utente, come grandezza e posizione delle finestre, colori utilizzati, nomi dei file




# Caratteristiche del sistema

- ❑ variabili di sistema, ad esempio il tempo del clock oppure numeri di serie (come l'indirizzo Ethernet) 
  - ❑ il numero di file su dischi od in particolare directory
  - ❑ lo spazio libero su ogni disco
  - ❑ le informazioni presenti nei vari buffer, nelle code per l'I/O, oppure nei driver per video
  - ❑ il numero di task nella coda di schedulazione del sistema operativo, le loro ID, le loro grandezze
  - ❑ oppure lo stato della memoria centrale
  - ❑ informazioni definite dall'utente, come grandezza e posizione delle finestre, colori utilizzati, nomi dei file
- però solo pochi bit casuali ...non molto segreti 



# Eventi esterni per la generazione

- ❑ contenuto delle battiture su tastiera 
- ❑ intervalli di tempo tra la digitazione dei tasti
- ❑ misure di tempi e posizione dei movimenti del cursore e/o del mouse

informazione raccolta durante una sessione di lavoro (altrimenti si può chiedere di battere tasti a caso oppure di muovere il mouse per un pò)

- ❑ tempo di arrivo dei pacchetti su rete



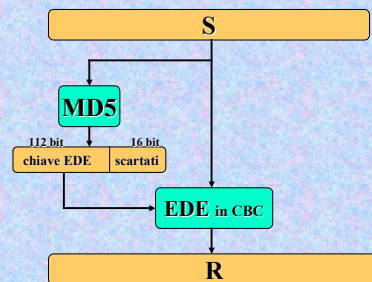
## Mischiare più sorgenti

- ❑ In genere si ottengono solo pochi bit casuali
- ❑ Usare più sorgenti casuali, non correlate tra di loro
- ❑ **Mischiare** i bit ottenuti tramite una opportuna funzione
  - evita che ci siano bit facili da indovinare
  - funzione complessa e non lineare di tutti i bit input
  - ad es., DES, triplo DES, SHS, MD4, MD5, ...

però nessuna funzione deterministica aumenta la casualità



## Mischiamento: un esempio



## Generazione pseudo-casuale

- ❑  $X_0$  valore iniziale o *seme*
- ❑ Generazione deterministica della sequenza

$$X_{i+1} = f(i, X_0, X_1, \dots, X_i) \quad i = 0, 1, 2, \dots$$

$X_1 X_2 X_3 X_4 \dots$



## Generazione pseudo-casuale

- ❑ Generatore a congruenze lineari
- ❑ Cifratura di un contatore
- ❑ j-bit Output feedback
- ❑ ANSI X9.17
- ❑ Generatore basato su RSA



## Generatore a congruenze lineari

[Lehmer 1951]

Ad oggi la tecnica più usata per la generazione pseudo-casuale

$$X_{i+1} \leftarrow (a \cdot X_i + b) \bmod m$$

$X_0$  *seme*



## Generatore a congruenze lineari

$$X_{i+1} \leftarrow (a \cdot X_i + b) \bmod m$$

$X_0$  *seme*


Che valori scegliere per a, b, m ?



## Generatore a congruenze lineari

$$X_{i+1} \leftarrow (a \cdot X_i + b) \pmod m$$

$a = 1$   
 $b = 1$

$$X_{i+1} = X_i + 1 \pmod m$$


Randomness 12

## Generatore a congruenze lineari

$$X_{i+1} \leftarrow (a \cdot X_i + b) \pmod m$$

$a = 7$   
 $b = 0$   
 $m = 32$   
 $X_0 = 1$


$$X_{i+1} = 7 X_i \pmod{32}$$

7, 17, 23, 1, 7, 17, 23, ...  
**periodo 4**

$a = 5$   
 $b = 0$   
 $m = 32$   
 $X_0 = 1$

$$X_{i+1} = 5 X_i \pmod{32}$$

5, 25, 29, 17, 21, 9, 13, 1, ...  
**periodo 8**



Randomness 13

## Generatore a congruenze lineari

$$X_{i+1} \leftarrow 7^5 \cdot X_i \pmod{2^{31}-1}$$

$7^5 = 16.805$   
 originalmente usato nella famiglia IBM 360 [1969]

Numero primo  
 Conveniente per aritmetica a 32 bit

- Genera tutti i numeri  $1, 2, \dots, 2^{31}-2$
- Molto usato per simulazione e statistica

Randomness 14


## Generatore a congruenze lineari

$$X_{i+1} \leftarrow 7^5 \cdot X_i \pmod{2^{31}-1}$$

$7^5 = 16.805$   
 originalmente usato nella famiglia IBM 360 [1969]

Numero primo  
 Conveniente per aritmetica a 32 bit

- Genera tutti i numeri  $1, 2, \dots, 2^{31}-2$
- Molto usato per simulazione e statistica
- Buono per scopi crittografici?



Randomness 15


## Generatore a congruenze lineari

$$X_{i+1} \leftarrow 7^5 \cdot X_i \pmod{2^{31}-1}$$

$7^5 = 16.805$   
 originalmente usato nella famiglia IBM 360 [1969]

Numero primo  
 Conveniente per aritmetica a 32 bit

Se conoscessi un solo  $X_i$ ,  
 potrei calcolare tutti i valori  
 seguenti ed anche i successivi!



Randomness 16

## Generatore a congruenze lineari

$$X_{i+1} \leftarrow (a \cdot X_i + b) \pmod m$$

Dati  $X_0, X_1, X_2, X_3$  si possono calcolare  $a, b, m$

$$\begin{cases} X_1 = (a \cdot X_0 + b) \pmod m \\ X_2 = (a \cdot X_1 + b) \pmod m \\ X_3 = (a \cdot X_2 + b) \pmod m \end{cases}$$

Randomness 17

### Cifratura di un contatore

$c \leftarrow$  valore iniziale

$c \leftarrow c + 1$

$c$  diversi  $\Rightarrow X_i$  diversi

chiave (seme)  $\rightarrow$  **E**

$X_i$

Randomness 18

### Cifratura di un contatore

$c \leftarrow$  valore iniziale

$c \leftarrow c + 1$

$c$  diversi  $\Rightarrow X_i$  diversi

chiave (seme)  $\rightarrow$  **E**

$X_i$

Usò di un generatore, non un semplice contatore

Randomness 19

### j-bit Output feedback

shift di j bit

64-j bit | j bit

$k$   $\rightarrow$  **DES**

j bit | 64-j bit

$X_i \leftarrow$  output di 64 bit

Si inizia cifrando IV

Randomness 20

### ANSI X9.17

seme  $\rightarrow$   $k, k'$

data ed ora corrente  $DT_i$

seme  $\rightarrow$   $V_i$

**EDE**

$\oplus$

**EDE**

$\oplus$

**EDE**

$V_{i+1}$

$X_i$  output

Randomness 21

### Generatore basato su RSA

- $p, q \leftarrow$  primi
- $n \leftarrow p \cdot q$
- $e \leftarrow$  intero in  $[3, \varphi(n)[$  tale che  $\gcd(e, \varphi(n))=1$

$z_0$  ( $i=0$ )  $\rightarrow$   $z_i \leftarrow z_{i-1}^e \bmod n$

$z_i$   $\rightarrow$  bit meno significativo  $\rightarrow X_i$

$i \leftarrow i + 1$

Randomness 22

### Generatore basato su RSA

- $p, q \leftarrow$  primi
- $n \leftarrow p \cdot q$
- $e \leftarrow$  intero in  $[3, \varphi(n)[$  tale che  $\gcd(e, \varphi(n))=1$

$z_0$  ( $i=0$ )  $\rightarrow$   $z_i \leftarrow z_{i-1}^e \bmod n$

$z_i$   $\rightarrow$   $c \cdot \log \log n$  bit meno significativi  $\rightarrow X_i$

$i \leftarrow i + 1$

Randomness 23



## Generatore crittograficamente forte

Deve passare i seguenti test:

- ❑ Test del prossimo bit
- ❑ Test statistici

(I due test sono equivalenti)



## Generatore crittograficamente forte

### Test del prossimo bit

dati i primi  $r$  bit dell'output nessun algoritmo efficiente può predire l' $(r+1)$ -esimo bit con probabilità significativamente migliore di  $1/2$

Questi sono  
i primi  $r$  bit:  
1101001...0



Qual è il prossimo?



## Generatore crittograficamente forte

### Test statistici

nessun algoritmo efficiente distingue tra l'output di un generatore ed una sequenza realmente casuale con probabilità significativamente migliore di  $1/2$

Ecco  $r$  bit:  
1101001...0



Casuale o  
pseudo-casuale?