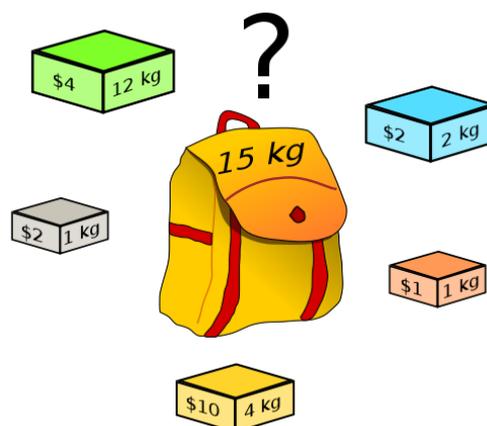


## idea della crittografia a chiave pubblica

- sviluppare un crittosistema in cui data la funzione di cifratura  $e_k$  sia **computazionalmente difficile** determinare  $d_k$
- Bob rende pubblica la **sua** funzione di cifratura  $e_k$
- Alice (e chiunque altro) può scrivere a Bob, cifrando il messaggio con la  $e_k$  senza bisogno di accordi preliminari
- Bob è l'unico che può decifrare il messaggio
- la funzione di cifratura  $e$  è una **funzione unidirezionale** (one-way function)
- informalmente, una funzione invertibile  $f : \mathcal{P} \rightarrow \mathcal{C}$  si dice unidirezionale se
  - dato  $x \in \mathcal{P}$ , il calcolo di  $f(x)$  è **facile**
  - per **quasi tutti** gli  $y \in \mathcal{C}$  il calcolo di  $f^{-1}(y)$  è **difficile**

## problema dello zaino (knapsack problem)

- un problema **NP**-completo è il **problema dello zaino**
- dato uno zaino che può contenere  $b$  unità
- e dati  $k$  oggetti di volume  $a_1, \dots, a_k$
- il problema è riempire lo zaino usando alcuni oggetti della lista



## problema dello zaino (knapsack problem)

- dobbiamo trovare una  $k$ -pla  $(e_1, \dots, e_k)$ ,  $e_i = 0, 1$
- tale che  $b = \sum_i e_i a_i$
- se possibile – oppure mostrare che una tale  $k$ -pla non esiste
- il problema è in **NP** – data una  $k$ -pla, è facile verificare se è una soluzione
- ma trovare una soluzione è difficile – procedendo per tentativi ho  $2^k$  possibilità da controllare
- e non si conosce un algoritmo veloce per risolverlo – il problema dello zaino è **NP**-completo

### esempio

- data la lista (323, 412, 33, 389, 544, 297, 360, 486)
- e dato  $b = 1228$
- un metodo possibile è un procedimento **greedy** (avido, goloso)
- a ogni passo, prendo l'oggetto più grande che entra ancora nello zaino
- in questo esempio, ho  
 $1228 = 544 + 684 = 544 + 486 + 198 = 544 + 486 + 33 + 165$
- e qui ci fermiamo
- invece si ha che  $1228 = 412 + 33 + 297 + 486$
- se avessimo una lista di 100 numeri di 50 cifre ognuno il problema diventa inavvicinabile

## sequenze supercrescenti

- ci sono particolare  $k$ -ple per cui il problema dello zaino è facile
- la sequenza  $(a_1, \dots, a_k)$  si dice supercrescente se ogni termine è maggiore della somma dei termini precedenti
- $\sum_{j=1}^{i-1} a_j < a_i$
- se ho una lista  $(a_1, \dots, a_k)$  supercrescente
- è facile vedere che il procedimento greedy visto prima risolve il problema dello zaino
- e che - se esiste - la soluzione è unica
- per esempio la successione  $1, 2, 4, 8, 16, \dots$  delle potenze di due è supercrescente
- il procedimento greedy è quello che si usa per scrivere un intero in base 2

## sequenze supercrescenti-esempio

- data la lista supercrescente  $(1, 3, 7, 15, 31, 63, 127, 255)$
- dato  $b = 328$
- $328 = 255 + 73 = 255 + 63 + 10 = 255 + 63 + 7 + 3$

## un crittosistema dal problema dello zaino

- la chiave pubblica di Bob è una lista  $(a_1, \dots, a_k)$  di interi positivi
- per cifrare, il messaggio dev'essere una stringa binaria di lunghezza  $k$
- $x = (e_1, \dots, e_k)$ ,  $e_i = 0, 1$  – Alice calcola  $\sum_i e_i a_i = b$
- e trasmette  $b$  a Bob
- nessun attaccante può decifrare – bisogna risolvere un problema dello zaino!
- in questo momento, non può decifrare neanche Bob...

## il crittosistema di Merkle–Hellman

- non possiamo usare una sequenza supercrescente come chiave pubblica!
- bisogna “mascherare” questa proprietà
- Bob sceglie una lista  $(a_1, \dots, a_k)$  supercrescente
- sceglie poi un intero  $n > \sum a_i$  e un intero  $u$  con  $(u, n) = 1$
- costruisce una nuova lista  $(a_1^*, \dots, a_k^*)$
- dove  $a_i^* = ua_i \pmod{n}$
- $(a_1^*, \dots, a_k^*)$  è la **chiave pubblica**
- $(a_1, \dots, a_k)$ ,  $n$  e  $u$  sono la **chiave privata**

- la cifratura funziona come già detto: Alice deve cifrare  $x = (e_1, \dots, e_k)$  – calcola  $\sum_i e_i a_i^* = b^*$  e trasmette  $b^*$  a Bob
- per decifrare, Bob calcola  $v$ , l'inverso di  $u$  modulo  $n$
- poi calcola  $vb^* \equiv b \pmod{n}$
- $b \equiv vb^* = v \sum e_i a_i^* \equiv v \sum e_i u a_i \pmod{n}$
- $v \sum e_i u a_i = uv \sum e_i a_i \equiv \sum e_i a_i \pmod{n}$
- ora sia  $b$  che  $\sum e_i a_i$  sono minori di  $n = \sum a_i$
- quindi  $b = \sum e_i a_i$
- Bob deve risolvere un caso facile del problema dello zaino per decifrare

## esempio

- data la lista supercrescente (1, 3, 7, 15, 31, 63, 127, 255)
- Bob sceglie  $n = 557$  maggiore della somma dei termini della lista e  $u = 323$  coprimo con 557
- moltiplicando per 323 e riducendo mod 557 ottiene (323, 412, 33, 389, 544, 297, 360, 486)
- la stringa binaria 01100101 viene codificata da Alice come  $412 + 33 + 297 + 486 = 1228$
- Bob riceve 1228 – sa che l'inverso di 323 mod 557 è 169
- calcola  $1228 \cdot 169 \pmod{557}$  e ottiene 328
- $328 = 255 + 73 = 255 + 63 + 10 = 255 + 63 + 7 + 3$
- riottiene quindi la stringa 01100101

## il crittosistema di Merkle–Hellman

- il crittosistema ora descritto è dovuto a Merkle e a Hellman (1978)
- è elegante – e molto più semplice dell’RSA
- sfortunatamente non è molto sicuro
- A. Shamir, A polynomial-time algorithm for breaking the basic Merkle - Hellman cryptosystem, IEEE Transactions on Information Theory (1984)
- il “mascheramento” non funziona molto bene