

## protocolli

- un **protocollo** è costituito da una serie di passi (step) e coinvolge **due o più** persone (parti, entità) allo scopo di svolgere un incarico
- proprietà
  - ogni persona coinvolta deve conoscere il protocollo in anticipo
  - ogni partecipante deve eseguire le disposizioni del protocollo
  - il protocollo non deve essere ambiguo
  - il protocollo deve essere completo
- un **protocollo crittografico** è un protocollo che utilizza la crittografia

## protocolli

- le parti si chiamano Alice, Bob, Carol, Davide,...
- a volte c'è un super-user chiamato Trent
- abbiamo già visto esempi di protocolli
  - scambio della chiave
  - scelta di un crittosistema e trasmissione di un messaggio cifrato
  - firma digitale

## secret spitting – divisione di un segreto

- solo il presidente della ACME conosce la ricetta della ACME-cola™
- se comunica la ricetta a un suo collaboratore, questi la può rivendere alla concorrenza
- si vuole un metodo che consenta di **dividere** l'informazione fra due o più collaboratori
- in modo che l'informazione che si dà a un singolo non consenta di risalire alla ricetta completa
- ma solo la conoscenza di **ogni** “parte” di informazione permetta di ricostruire il segreto

## secret splitting

- Trent vuole dividere un segreto fra Alice e Bob
- il segreto è un messaggio  $M$  – che pensiamo come una stringa binaria di lunghezza  $l$
- Trent genera in maniera casuale un'altra stringa binaria  $r$  di lunghezza  $l$
- calcola lo XOR fra  $M$  e  $r$ :

$$s = M \oplus r$$

- comunica  $r$  ad Alice e  $s$  a Bob
- Alice e Bob **insieme** possono ricostruire  $M = s \oplus r$
- ma la conoscenza solo di  $s$  o solo di  $r$  non dà alcuna informazione su  $M$
- di fatto, Trent cifra il messaggio con un one-time pad e dà il testo cifrato a Bob e la chiave ad Alice; abbiamo provato che l'one-time pad è a segretezza perfetta

## secret splitting

- questo schema si generalizza immediatamente
- se Trent deve dividere il segreto  $M$  fra  $k$  “collaboratori”  $A_1, A_2, \dots, A_k$
- genera  $k - 1$  stringhe binarie  $r_1, r_2, \dots, r_{k-1}$  di lunghezza  $l$
- calcola  $s = M \oplus r_1 \oplus r_2 \dots \oplus r_{k-1}$
- comunica  $r_1$  a  $A_1$ ,  $r_2$  a  $A_2$ ,  $\dots$   $r_{k-1}$  a  $A_{k-1}$  e  $s$  a  $A_k$
- in questo protocollo, l'unico modo per ricostruire il segreto è avere accesso a **tutte** le informazioni parziali

## secret sharing – condivisione di un segreto

- bisogna progettare uno schema per il lancio di un missile nucleare
- dieci ufficiali della base hanno le chiavi di lancio
- si vuole uno schema in cui un ufficiale (improvvisamente impazzito) non possa lanciare il missile da solo
- e neanche due o tre ufficiali possano lanciare il missile ma quattro sì
- o anche bisogna progettare uno schema in cui, per autorizzare un lancio, servano
  - due generali
  - oppure un generale e due colonnelli
  - oppure quattro colonnelli

## threshold scheme – schema a soglia

- problemi di questo tipo sono risolti da un  $(m, n)$ -threshold scheme (schema a soglia  $(m, n)$ )
- è uno schema in cui il segreto viene diviso in  $n$  parti, chiamate **ombre** o “shares”
- in modo tale che bastino  $m$  ombre per ricostruire il segreto ( $m < n$ )
- nell'esempio precedente si ha  $m = 4$ ,  $n = 10$ .
- in uno schema in cui alcuni utenti sono più “importanti” di altri, gli utenti più “importanti” ricevono più di un'ombra
- nell' esempio, si ha uno schema a soglia con  $m = 4$  in cui ogni colonnello ha un'ombra, ogni generale ne ha due

## schema a soglia di Shamir

- è uno schema a soglia che usa l'interpolazione di Lagrange in un campo finito  $\mathbb{Z}_p$
- $p$  è un primo maggiore del massimo numero di ombre; il segreto è un elemento  $M$  di  $\mathbb{Z}_p$
- per creare un  $(m, n)$ -schema a soglia, Trent genera un polinomio  $f(x)$  di grado  $m - 1$  a coefficienti in  $\mathbb{Z}_p$  in cui il termine noto è il segreto  $M$
- e  $n$  elementi  $x_1, x_2 \dots x_n$  in  $\mathbb{Z}_p$
- le  $n$  ombre si ottengono calcolando il valore che il polinomio assume nei punti:  $k_i = f(x_i)$
- l' $i$ -esimo partecipante allo schema conosce  $m$ , il primo  $p$  e la coppia  $(x_i, k_i = f(x_i))$
- con almeno  $m$  ombre si riesce a determinare il polinomio – e quindi a ricostruire il segreto!

## schema di Shamir – esempio

- sia  $M = 11$  – vogliamo realizzare uno schema (3, 5)
- Trent sceglie  $p = 13$  e genera il polinomio di grado due  $f(x) = 7x^2 + 8x + 11 \pmod{13}$
- le cinque ombre si ottengono calcolando  $f$  per esempio in 1, 2, 3, 4, 5  
 $f(1) = 26 = 0$ ,  $f(2) = 55 = 3$ ,  $f(3) = 98 = 7$ ,  
 $f(4) = 155 = 12$ ,  $f(5) = 226 = 5$
- per ricostruire il polinomio dobbiamo conoscere il primo  $p$  e almeno tre ombre – per esempio dalla conoscenza della seconda, terza e quinta si ha il sistema

$$\begin{cases} 2^2a + 2b + M = 3 \\ 3^2a + 3b + M = 7 \pmod{13} \\ 5^2a + 5b + M = 5 \end{cases}$$

nelle incognite  $a$ ,  $b$  e  $M$

- risolvendo il sistema, si ritrova  $a = 7$ ,  $b = 8$  e  $M=11$

## generalizzazioni di uno schema a soglia

- possiamo immaginare la presenza di due gruppi  $A$  e  $B$  (per esempio, ostili fra loro) e si vuole che il segreto possa essere ricostruito solo con il contributo di entrambi i gruppi
- è facile modificare lo schema di Shamir per affrontare questa situazione
- per esempio, si richiedono almeno tre ombre provenienti dal gruppo  $A$  e due dal gruppo  $B$
- si scrive il segreto  $M$  come un prodotto  $M_A \cdot M_B$  in  $\mathbb{Z}_p$
- si generano due polinomi in  $\mathbb{Z}_p$ ;  $f_A = a_Ax^2 + b_Ax + M_A$ ,  
 $f_B = a_Bx + M_B$
- ai membri del gruppo  $A$  si danno le ombre ottenute calcolando il valore del polinomio  $f_A$ , e a quelli del gruppo  $B$  le ombre ottenute calcolando il valore del polinomio  $f_B$
- per ricostruire il segreto è indispensabile la partecipazione di entrambi i gruppi

## PKC a chiave multipla

- nell’RSA, la chiave è  $(N, p, q, a, b) \mid ab \equiv 1 \pmod{\phi(N)}$
- $b$  è pubblica, e  $a$  è privata
- oppure sono entrambe segrete: Alice conosce solo  $b$ , Bob conosce solo  $a$  (e non conoscono la fattorizzazione di  $N$ )
- in questo modo solo Alice e Bob possono comunicare fra loro
- posso pensare a una generalizzazione dell’RSA con più di due chiavi
- $(N, p, q, a_1, a_2, \dots, a_k) \mid a_1 a_2 \dots a_k \equiv 1 \pmod{\phi(N)}$
- si ha sempre che  $x^{a_1 a_2 \dots a_k} \equiv x \pmod{N}$
- se per cifrare si usano ad esempio  $a_1$  e  $a_2$ :  $y = x^{a_1 a_2} \pmod{N}$
- per decifrare bisogna conoscere  $a_3, \dots, a_k$
- $y^{a_3 \dots a_k} = x^{a_1 a_2 \dots a_k} \equiv x \pmod{N}$

## secret broadcasting

- la PKC a chiave multipla si può usare in uno schema di secret broadcasting (trasmissione/diffusione di un segreto)
- si ha un messaggio che viene trasmesso a  $n$  utenti
- ma si vuole che solo un sottoinsieme di  $k$  di essi sia in grado di decifrarlo
- questo sottoinsieme cambia a ogni trasmissione