

# Secret Sharing

$(2, n)$ : tutti hanno 's'

$(t, n)$ ,  $1 \leq t \leq n$

s rivelato con  
t shares su n

$(M, n)$

dati n parties, offrolo

Ri  $\wedge$  Party " $i$ ",  $1 \leq i \leq n-1$ ,

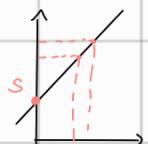
all'  $n$ -esimo party do lo share

$|S - \Sigma R_i|$ ; ho

perfect secrecy fino

a " $n-1$ " parties, con n  
rivelalo tutto.

## Shamir ( $t, n$ )



esempio:  $(2, n)$ ,  $t=2 \rightarrow \deg \text{Polynom.} = t-1$ , se  $t=2 \rightarrow$  retta

$f(x) = s + \alpha x$ , share  $(i, f(i))$ , per REBUILD ricrea la retta!

secret  $\uparrow$  rand

$$\text{dati } (x_1, y_1) \text{ e } (x_2, y_2) \rightarrow \frac{y - y_2}{y_2 - y_1} = \frac{x - x_2}{x_2 - x_1} \rightsquigarrow y = s$$

# Lagrange Interpolation

Con  $\deg(P) = t - 1$ , e dati  $t$  punti  $(x_1, y_1), \dots, (x_t, y_t)$  allora  $y = \sum_{i=1}^t y_i L_i(x)$

dove  $L_i(x) = \prod_{i \neq k} \frac{x - x_k}{x_i - x_k}$

- $\rightarrow 1, x = x_i$
- $\rightarrow \emptyset, x = x_k$

, ponendo  $x = \sigma$  trovo  $y = s$

La conoscenza di alcuni shares da' Leak Info,  $s = y_1 L_1 + ?L_2 + y_3 L_3 = 476 - 31$

NB: con modular arithmetic  $\rightarrow$  UNCONDITIONALLY SECURE! Sopra non l'ho usato!  
 $\uparrow$  se mod p  $\rightarrow$  uniformly distributed

Ho leaks perché il vero Shamir Scheme usa mod p con p appena più grande del

motivo intervallo di interese ( $(0, 99) \rightarrow P=101$ ), poiché sono UNCONDITIONALLY SECURE, in RSA

ero COMPUTATIONAL SECURE (large prime) in  $L_i = \sum \frac{\alpha}{B} \rightsquigarrow$  modular inverse

IDEAL  $\Leftrightarrow |\text{Share}| = |\text{Secret}|$ .

## Homomorphic

Due regretti  $S_F, S_G$ , con due polinomi per gli shares  $f(i), g(i)$ , vale che:

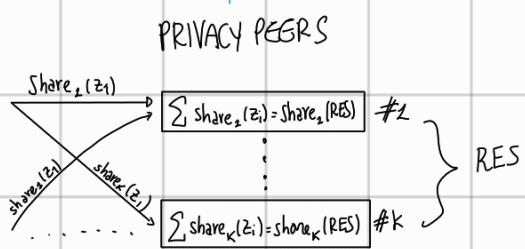
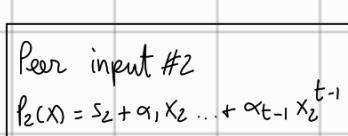
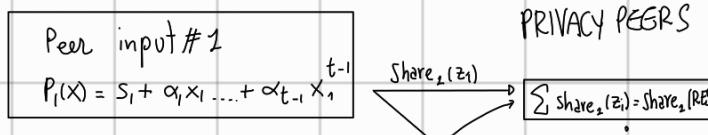
$\text{SUM(shares)} = \text{Shares}(S_F + S_G)$ , cioè  $f(i), g(i) \rightsquigarrow f(i) + g(i) \rightsquigarrow (f+g)(i)$

Esempio:  $\sum \text{donaz} = \infty \rightsquigarrow (\sum \text{rand} + \sum (\text{rand} + \text{donaz})) \bmod N = \infty$ , senza rivelare le singole donaz.

## Security Multiparty Computation

Opero su dati critptati, result in clear, va bene con LINEAR  $\alpha_i f_i + \alpha_j f_j + \dots$

Evita 3rd party (sa tutto), ma ho  $m$  privacy peers che ricevono gli shares



- INPUT PEER:  
 $m \geq 3$  (senza SUBTRACT)  
 puoi avere anche PRIVACY PEER
- PRIVACY PEER:  
 $k \geq 2$   
 threshold #peers:  $2 \leq t \leq K$   
 $(t, k)$

# Verifier Scheme

$P(x)$ , shares personali, NON controllabili! Risolvo con Feldman

- dealer (colui che distribuisce gli shares)

crea  $P(X) \rightarrow$  affida shares  $(x_i, y_i) \rightarrow$  **COMMITMENT**:  $C_0 = g^s \text{ mod } p, \dots, C_{t-1} = g^{x_{t-1}} \text{ mod } p,$

cioè realizzo bind  $\langle \text{shares}, \text{commitment} \rangle$

- Verifier (controllo che party "i" sia onesto)

Party "i" riceve  $(x_i, y_i) \rightarrow$  calcola  $[C_0 \cdot C_1^{x_1} \cdot \dots \cdot (C_{t-1})^{x_{t-1}}] \text{ mod } p = [g^{s + \dots + x_{t-1} \cdot x_i}] \text{ mod } p \stackrel{?}{=} g^{y_i}$

## Commit

• HIDING: COMMIT( $\alpha$ )  $\rightarrow$  no info su  $\alpha$ , • BINDING: Comm( $\alpha$ ) UNIVOCO ( $\forall \text{com}(\alpha) = \text{com}(\alpha')$ )

•  $C = g^x \text{ mod } p$  è commitment perché: 1) dato  $C$ , trovare  $x$  è computational Hiding

2) impossibile trovare  $x'$ :  $g^{x'} \text{ mod } p = c$  ( $1 < x' < p-1$ , altrimenti userei  $\phi(N)$ ), Perfect Binding

computational  
secure

$g^{\alpha} \text{ mod } p$  è HOMOMORPHIC:  $\text{comm}(\alpha + \beta) = \text{comm}(\alpha) \cdot \text{comm}(\beta) = g^{\alpha} \cdot g^{\beta} = g^{\alpha + \beta}$

$C_0 = g^s$  Leak Info, try until  $g^x = C_0$ . ↗ (Bind + Hide) Perfect

- $H(x)$  è tutto computazionale (Piccole range + collisione)



## Pedersen

definiti  $\langle g, h \rangle \rightsquigarrow \text{comm}(\alpha, \text{rand}) = g^{\alpha} \cdot h^{\text{rand}} \text{ mod } p$

• Homomorphic:  $\text{comm}(\alpha + \beta, R_1 + R_2) = g^{\alpha + \beta} \cdot h^{R_1 + R_2} \text{ mod } p = (g^{\alpha} \cdot h^{R_1}) \cdot (g^{\beta} \cdot h^{R_2}) = \text{comm}(\alpha, R_1) \cdot \text{comm}(\beta, R_2)$

Perfect Hiding:  $\forall \alpha \neq \alpha' \exists! R: g^{\alpha'} h^R = g^{\alpha} h^R$  (avendo monnaia ha  $\langle \alpha, R \rangle \neq \langle \alpha', R \rangle$ , vede SOLO il valore finale)

Comput. Bind: Sender non dovrebbe trovare  $g^{\alpha} h^R = g^{\alpha} h^{\tilde{R}}$ , ma  $g^{\alpha} \sqrt[n]{h^R} = g^{\alpha} \cdot g^{\frac{R}{n} \log g} = g^{\alpha} \cdot g^{\frac{\tilde{R}}{n} \log g} \Rightarrow \alpha + \tilde{R}w = \alpha' + \tilde{R}w \rightsquigarrow \tilde{R}' = \frac{\alpha - \alpha' + \tilde{R}w}{w} \text{ mod } ?$   
 serve lui, cioè non nota  
 ↳ riesce solo se conosce  $w = \log g$

• **Dealer**: genera  $\begin{cases} P(x) = s + a_1x + \dots + a_{t-1}x^{t-1} \\ Q(x) = r + b_1x + \dots + b_{t-1}x^{t-1} \end{cases}$   $\rightarrow \forall i \text{ share } (x_i, y_i, z_i) \rightarrow \text{commotto } c_0 = g^s \cdot h^r \dots c_{t-1} = g^{a_{t-1}} \cdot h^{b_{t-1}}$

• **Verifier**: party  $i$  vede se  $c_0 \cdot c_1^{x_i} \cdot \dots \cdot c_{t-1}^{x_i^{t-1}} = g^{s+a_1x_i+\dots+a_{t-1}x_i^{t-1}} \cdot h^{r+b_1x_i+\dots+b_{t-1}x_i^{t-1}} \stackrel{?}{=} g^{y_i} \cdot h^{z_i} \pmod{p}$

**Importante**: • l'esponente deve essere mod PRIME  $\rightarrow$  strong primes  $p=2q+1$

•  $\mathbb{Z}_{11}^*$   $\sim p=11 \sim q=5$ ,  $G=\{g^0, g^1, \dots, g^{10}\}$ , se  $g \in \mathbb{Z}_{11} - \{1, 2, 5\}$ ,  $g$  puo' generare tutti i numeri da 2 a 10 (**e' generator**) oppure un sottogruppo di cardinalita' "2" o "5"  
 $(\phi(11)=10=2 \cdot 5)$

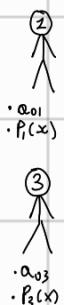
•  $g \in QR \Leftrightarrow \exists x: x^2 \pmod{p} = g$ , vero se  $g^{\frac{p-1}{2}} \pmod{p} = 1$ , voglio  $g \in QR$  perche' |ordine subgroup| è PRIME,  
quindi  $g \in QR$  **NON e' generator**, poiche' crea sottogruppo!

• concludendo:  $\begin{array}{l} \left. \begin{array}{l} \text{shares } y_i \\ \text{Lagran } l_i \\ (\sum l_i \cdot y_i = s) \end{array} \right\} \pmod{q} \\ ; \end{array} \quad \begin{array}{l} \text{calcolo } [c_0 \cdot c_1^{x_i} \cdot c_2^{x_i^2} \dots] \pmod{p} \\ \text{calcolo } c_i \text{ (es: } g^s \pmod{p}) \end{array}$



**Distributed Key Generator**: Pk NOTA, PrvKey manda a tutti

esempio (3,4)



- 1) Ogni party  $i$  crea  $a_{0,i}$ , crea  $P_i(x) = a_{0i} + \dots + a_{zi}x^z$
- 2) Ogni party  $i$  calcola  $P_j(j \neq i)$  da dare a  $j$ , per se ricevere  $P_i(i)$
- 3) Ogni party  $i$  crea  $y_i = \sum_{k=1}^4 P_k(i)$  (homomorphic)
- 4) Ogni party  $i$  pubblica  $g^{a_{0i}}$  (NO cheat)
- 5) Se è ignoto  $= P_1(x) + P_2(x) + P_3(x) + P_4(x) = (a_{01} + a_{02} + a_{03} + a_{04}) + \dots = P(x)$   
Ma ogni share ha  $P(i) = P_1(i) + P_2(i) + P_3(i) + P_4(i) = y_i$
- 6) Ognuno può calcolare  $g^{a_{01}} \cdot g^{a_{02}} \cdot \dots = g^s$ ; con 3 shares  $y_i$  e interpolazione ricostruisce! (s rivelabile in  
secondo momento, quando necessario!)

# Threshold ENC (secret sharing + group oriented crypto)

- Ci interessa Threshold Encryption, usando **EL GAMAL**, cioè:
  - were DH Keys (prima per Key Agreement) per Encryption Protocol (sfruttando DLOG)

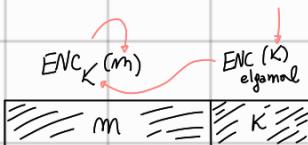
• DLOG e' scalabile, ed ElGamal usa DH per **asymmetric cipher**, come?

Definisco:  $\begin{cases} p \text{ large prime}, s \text{ private key (receiver)}, r \text{ random (moto al sender)} \\ g \text{ group generator}, h = g^s \text{ Pub Key} \end{cases}$  , operazioni mod p

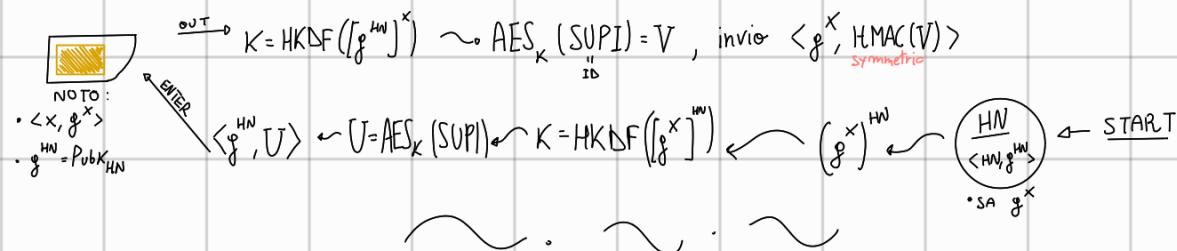
Allora:

$$\text{ENC}(g^r, m \cdot h^r) ; \text{DEC: NOTO } g^r, m \cdot h^r, s, p \rightarrow M = m \cdot \frac{h^r}{(g^r)^s} = m \cdot \frac{(g^s)^r}{(g^r)^s} \text{ modp}$$

ESEMPIO ASYM. HYBRID :



Oggi, 5G: Hybrid ENC 5G: Elliptic Curve Integrated Encryption Scheme



Threshold El Gamal

DEC  $\Leftrightarrow$  t su m party in GRUPPO vogliono decifrarlo ( $t, m$ ). Nessuno ha PrivKey! Come lo realizzo?

• distribuendo shares, e ricontrollando 's'? 's' rivelato, funzionerebbe solo una volta! Osservo che:  $m_1 = \frac{c_1}{(g^r)^s}$ ,  $m_2 = \frac{c_2}{(g^r)^s}$ , in base ai valori x i metà.

NUOVA idea: Shares esponenziali:  $A^x = A^{x_1+x_2} = A^{x_1} \cdot A^{x_2} \rightsquigarrow A^s = A^{\sum y_i \cdot x_i}$ , allora  $\forall i$  affido  $(g^r)^{y_i \cdot x_i (0)}$  (personale), **threshold**:  $(g^r)^{\sum y_i \cdot x_i} = (g^r)^s$

ognuno ha il suo share, nessuno vede gli altri, se UNKNOWN, ho infatti calcolato  $(g^r)^s$ . Con  $(g^r, m \cdot h^r)$   $m = \frac{m \cdot h^r}{(g^r)^s} = \frac{m \cdot g^{rs}}{g^{rs}}$

Con threshold
 

- Lagrange Interpol.
- Shamir ( $t, n$ )
- modular inverse
- Robusto

# Threshold Signature in RSA

La firma (attestato di validità) posta  $\Leftrightarrow t \leq m \leq (t, n)$  fanno  $\text{SIGN}(\text{msg})$ . NON VOGLIO "size of t"

\* COME? Definisco large prime  $N = p \cdot q$ ,  $\phi(N)$ ;  $e \cdot d = 1 \pmod{\phi(N)}$

► MAKE SIGN  $[m, H(m) \pmod{N}]$  poiché è una firma, uso 'd', metà solo a me!

$$\text{VERIFY SIGN: } H(m) = [H(m)^d]^e$$

► OSSERVAZIONI: RSA usa interi, se ho  $M^{\frac{1}{x}} = M$  modular inverse, si calcola con  $\phi(N)$ , che non ho!

Dealer calcola shares da  $f(x) = (d + \alpha_1 x + \dots + \alpha_{t-1} x^{t-1}) \pmod{\phi(N)}$ .

I parties ricombinano  $H(m)^{\sum_i y_i L_i x_i} \pmod{N} = H(m)^d \pmod{N}$ , ma attenzione:  $L_i x_i = \frac{\alpha_i}{B_i} \pmod{\phi(N)}$ , come lo calcolo?

Serve  $\phi(N)$ , dovrei usare factoring  $\rightarrow$  rompo RSA! (si basa su calcolare  $\frac{1}{e}$  solo con  $\phi(N)$ )

Dovendo evitare gli inversi!  $L_i(x) = \frac{\dots}{i(L-i)!} \sim \frac{L!}{i(L-i)!} = L! \cdot L_{i(x)} \stackrel{i(x) \text{ intero}}{=} \prod_{j=1}^{i(x)} j \sim \text{NO inverso} \rightarrow \text{non serve } \phi(N)!$

$H(m)^{\sum_i y_i L_i(x)} \pmod{N} = H(m)^{\frac{1}{e} d} \pmod{N}$ , come tolgo  $L!$ ? calcolo Inverso? tornerei come prima!

## RSA Common Modulus attack

Alice e Bob, PubK diverse, usano stesso  $\text{mod } N$  e stesso  $\text{msg } N$ :  $\left( \begin{array}{l} \text{Alice: } M^{e_1} \pmod{N} \\ \text{Bob: } M^{e_2} \pmod{N} \end{array} \right) \vee (\gcd(e_1, e_2) = 1)$  allora:

$\exists r, s : e_1 r + e_2 s = 1$  (ext. eucl. alg)  $\sim (M^{e_1})^r \cdot (M^{e_2})^s \pmod{N} = M^{e_1 r + e_2 s} = M^2$ , senza PrivKey

Y do sfruttato a mio vantaggio:  $\left[ \begin{array}{l} [H(m)^d]^r \\ [H(m)^d]^s \end{array} \right] \stackrel{\text{se coprimi}}{\sim} \exists r, s : [H(m)^d]^{r+s} = [H(m)]^d$ , quello che volevo!

Funziona sicuramente se c'è e larger prime than  $L!$

NOTA: questa è semplificazione di Shoup, che usa QR + safe primes  $\rightarrow p = 2p' + 1$ ,  $q = 2q' + 1 \rightarrow \phi(N) = 4p'q'$   $\rightarrow$  NO PROBLEMI DI INVERSO

# Resistenza a furto dati (McKenzie)

"resilient device": dispositivo utile solo dal proprietario, include una secret key per fare tutto, ed è connesso, quando usato, con un "protection server" (NON certificato) che conferma che il device è in possesso del proprietario, e quindi manda SK.

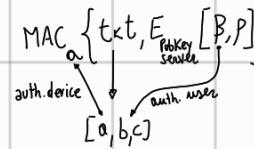
- Modalità basic: uso TICKETS, contenente info per autenticazione user (non stored su server, non ha SK diretta!)

INIZIO (immetto 'P')

	Presente nel Device	Rimosso dal device
iniziali	$P_k_{device}$ , rand 'a', rand 'v'	$SK_{device}$ , $PK_{server}$ (pubblico)
calcolo		$b = H(P)$ (no dict. attack) $c = SK \oplus PRF(v, P)$
req	$Tkt = E_{PK_{server}}[a, b, c]$	

RIPRENDI KEY

immetto P, calcolo  $\beta = H(P)$  e rand p. Invio al server MAC  $\{txt, E_{PK_{server}}[B, P]\}$   
 allora invio  $P \oplus C$  al client, che fa:  
 $(P \oplus C) \oplus P = C = SK \oplus PRF(v, P) \oplus PRF(v, P) = SK$



RUBO SERVER + PW:  $\rightsquigarrow v$  in device  $\rightsquigarrow$  NO  $PRF(v, PW)$  ✓

RUBO DEVICE, NO PW:  $\rightsquigarrow$  dictionary attack ONLINE ✓

RUBO DEVICE + SERVER:  $\rightsquigarrow$  dictionary OFFLINE (attacco  $H(PW)$ ) ✓

• rubo  $P_c + PW$  ✎: SK nel dispositivo

Quanto è resistente?

- Modalità (2,2) secret sharing

idea RSA:  $N = p \cdot q$ , shares:  $\begin{cases} d = \text{rand int} \\ d_1 = \text{rand int} \\ d_2 = d - d_1 \bmod \phi(N) \end{cases}$

$\rightsquigarrow H(m) \cdot H(m) \stackrel{d_1}{\equiv} \stackrel{d_2}{\equiv} \stackrel{d}{\equiv} H(m) \bmod N$ , ho moltiplicato i tag costanti usando i due shares! • NO SHARING

• NO LAGRANGE

INIZIO (immetto

	Presente nel Device	Rimosso dal device
iniziali	rand 'v', 'a', N, e, t for disabling	$d, \phi(N), PK_{server}$ (pubblico)
calcolo	$d_1 = PRF(v, P)$ $d_2 = d - d_1 \bmod \phi(N)$ $u = H(t)$ $b = H(P)$	
req	$Tkt = E_{PK_{server}}[a, b, u, d_2, N]$	$C = SK_{device} \oplus PRF(v, P)$

RIPRENDI KEY

immetto P, calcolo  $\beta = H(P)$  e rand p. Invio al server MAC  $\{txt, E_{PK_{server}}[H(m), B, P]\}$   
 il quale invia  $P \oplus [H(m)]^{d_2}$  al client, che fa:

$(P \oplus [H(m)]^{d_2}) \oplus P \rightsquigarrow [H(m)]^{d_2} \cdot [H(m)]^{d_1} \rightsquigarrow H(m)^d$ , se rubo dev+pw ho  $d_1$ , ma non  $d_2$ , informo con  $u = H(t)$  key disabling il server.

# Linear Secret Sharing e Control Matrix

Vorrei riadattare Secret Sharing ( $(3,4) \rightarrow (2,3)$ ), analizzo Shamir:

$$Y_i = S + a_1 x_i + \dots + a_{t-1} x_i^{t-1} = [S, a_1, \dots, a_{t-1}] \cdot [1, x_i, \dots, x_i^{t-1}], \text{ puo' essere matrice } Ax = b?$$

$$\begin{matrix} p_1 \\ \vdots \\ p_4 \end{matrix} \begin{bmatrix} 1 & x_1 & x_1^2 \\ \vdots & \vdots & \vdots \\ 1 & x_4 & x_4^2 \end{bmatrix} \cdot \begin{bmatrix} S \\ r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} = \begin{bmatrix} Y_1 = S + r_1 x_1 + r_2 x_1^2 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} \begin{matrix} \text{share}_1 \\ \vdots \\ \text{share}_2 \end{matrix}$$

$$A \cdot x = b$$

ho un solo Linear System senza Lagrange, A puo' essere ARBITRARIA (LSSS)

Per ricostituire il segreto (3 shares)

$$\begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} S \\ r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} \rightsquigarrow \begin{pmatrix} S \\ r_1 \\ r_2 \end{pmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix}^{-1} \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix}$$

## 3D SPAN

Immagino schema  $(3,4)$ , voglio ricostituire  $s$ !

$$\bullet \quad \left[ C_1(1 x_1 x_1^2) + C_2(1 x_2 x_2^2) + C_3(1 x_3 x_3^2) \right] \begin{pmatrix} S \\ r_1 \\ r_2 \end{pmatrix} = S \iff \star = (100) \rightarrow (100) \begin{pmatrix} S \\ r_1 \\ r_2 \end{pmatrix} = S \quad !, \text{ l'equazione equivale:}$$

$$C_1(1 x_1 x_1^2) \begin{pmatrix} S \\ r_1 \\ r_2 \end{pmatrix} + C_2(1 x_2 x_2^2) \begin{pmatrix} S \\ r_1 \\ r_2 \end{pmatrix} + \dots = C_1 \cdot Y_1 + C_2 \cdot Y_2 + C_3 \cdot Y_3 = S, \text{ allora ciò che devo fare è:}$$

$$1) \text{ trovo } C_1, C_2, C_3 : \star = (1, 0, 0) \rightsquigarrow \sum C_i \cdot Y_i = S$$

Trivial Secret  $(m,n)$  è LSSS, ed è homomorphic:

$$\text{dati } \begin{cases} Ax_a = y_a \text{ dove } x_a = (s_a, r_{1a}, r_{2a}, \dots), y_a = (\text{share}_{1a}, \text{share}_{2a}, \dots) \\ Ax_b = y_b \text{ dove } x_b = (s_b, r_{1b}, r_{2b}, \dots), y_b = (\text{share}_{1b}, \text{share}_{2b}, \dots) \end{cases} \rightarrow \overbrace{y_a + y_b}^{\text{sum share}} = A(x_a + x_b) = A(s_a + s_b, \text{rand}, \text{rand}, \dots) \text{ share of the sum}$$

Negli schemi precedenti  $(t,m)$  parlo di POLICY, NO threshold (alcuni partecipanti più forti di altri, creano sottospazio!)

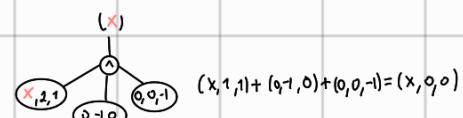
LSSS supporta tutti i predici booleani tranne negazione, ma gli schemi risultanti possono

NON essere ideali (Pi ha  $\geq 2$  shares), dipende da come viene scritta la policy!

## • REGOLE

- AND GATE : dim+1, espando : valore vett. iniziale =  $\sum$  vettori espansi :

(x)



$$(x, 1, 1) + (0, -1, 0) + (0, 0, -1) = (x, 0, 0)$$

- OR GATE : copio vett. precedente :



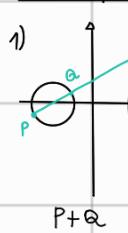
NB: Se ho 3 gruppi  $(A \wedge B) \vee (C \wedge D) \vee (E \wedge F)$  e faccio espansione in  $(A \wedge B)$ , sulle altre parentesi faccio PADDING = dim "i" aggiungendo zero.

# ELLIPTIC CURVE, usate perché scalano bene. La difficoltà di DLOG

cambia in base ai gruppi. FORMULA:  $y^2 = x^3 + ax + b$  con  $4a^3 + 27b^2 \neq 0$

OPERAZIONE:  $P + Q = R \in \text{Curva}$

esempi



1)

$P+Q$

2)

$P+P$

3)

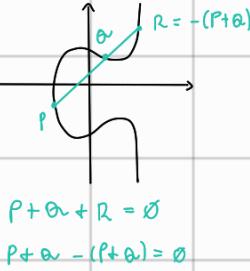
$P-P = \emptyset$

4)

$P+0 = P$ , per questo faccio FLIP, così  $P+0 = P$

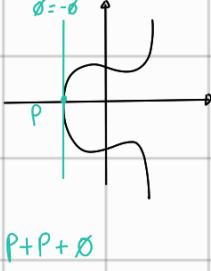
NB: Se ho 3 punti QUALSIASI in linea retta.  $A+B+C = \emptyset$

1)



$$P+Q = \emptyset$$

2)



$$P+P = \emptyset$$

- dati 2 punti, come trovo il terzo?

$P \neq Q$

La retta passante per  $(P, Q)$  è:  $y = y_1 + \frac{y_2 - y_1}{x_2 - x_1} (x - x_1)$ ; infatti il fascio di rette

passante per  $P$  è  $y = y_1 + m(x - x_1)$ , e scelgo la passante per  $Q$ ,  $\lambda - \frac{y_2 - y_1}{x_2 - x_1}$

Ho  $\begin{cases} y = y_1 + \lambda(x - x_1) \text{ eq. retta} \\ y^2 = x^3 + ax + b \sim x^3 - \lambda x^2 + o(x^2) \end{cases}$

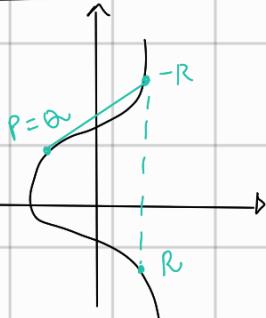
eq. passante per 3 punti:  $(x-x_1)(x-x_2)(x-x_3) = x^3 - (x_1+x_2+x_3)x^2 + o(x^2)$

$$\begin{aligned} \lambda^2 &= x_1^2 + x_2^2 + x_3^2 \\ x_3 &= \lambda^2 - x_1 - x_2 \end{aligned}$$

Noto  $x_3$ , trovo  $y_3 = \square [y_1 + \lambda(x_3 - x_1)]$

devo invertire!

P = Q



$$\left\{ \begin{array}{l} y = y_1 + \lambda(x - x_1) = y_1 + y'_1(x - x_1) \\ y^2 = x^3 + ax + b \end{array} \right. \xrightarrow{\text{deriva}} 2y \frac{dy}{dx} = 3x^2 + a \rightarrow \frac{dy}{dx} = y'_1 = \frac{3x_1^2 + a}{2y_1} = \lambda$$

Moto  $\lambda \rightarrow \left\{ \begin{array}{l} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = -[y_1 + \lambda(x_3 - x_1)] \end{array} \right.$

~ . ~ . ~ . ~ . ~

## Elliptic Curve su $\mathbb{Z}_p$ (int mod p)

Curva  $E_p(x, y) : \begin{cases} \bullet x, y \text{ integer mod } p \\ \bullet y^2 \text{ mod } p = (x^3 + ax + b) \text{ mod } p \end{cases}$ , #punti finita (se ho  $\#x = \#y = p \sim p^2$  punti)

ESEMPIO :  $(x^3 + x + 1) \text{ mod } 5$

al max  $5 \cdot 5 = 25$  punti +  $\emptyset = 26$

Copie  $(i, j)$ ,  $i \in [0, 4]$ ,  $j \in [0, 4]$  :  $j^2 \text{ mod } 5 = (i^3 + i + 1) \text{ mod } 5$ , come trovo i punti?

Provo con  $(x=1, y=3)$  :  $x^3 + x + 1 \Big|_{x=1} = 3$ , poi provo :  $\exists y : y^2 = 3$ , NO!  $\rightarrow (1, 3) \notin$  curve!

Provo con  $(x=0, y=1)$  :  $x^3 + x + 1 \Big|_{x=0} = 1$ ,  $\exists y : y^2 = 1$ , si,  $y=1 \rightarrow (0, 1) \in$  curve!

Penso trovarli senza andare a tentativi? Sì, partendo da  $P=0,1$  faccio  $\overset{?}{[k]}P$  fino a che trovo  $\emptyset$

ESEMPIO :  $2P = (0, 1) + (0, 1)$  (caso  $P=0$ )

• trovo  $\lambda = y'_1 \sim y^2 dy = x^3 + \overset{1}{a}x + \overset{1}{b} dx \rightarrow \frac{dy}{dx} = \frac{3x^2 + 1}{2y} \Big|_{\substack{x=0 \\ y=1}} = \frac{1}{2} \text{ mod } 5 = 3 \text{ mod } 5$

• trovo  $x_3 = \lambda^2 - x_1 - x_2 = 9 \text{ mod } 5 = 4 \text{ mod } 5$

• trovo  $y_3 = -[y_1 + \lambda(x_3 - x_1)] \text{ mod } 5 = -[1 + 3(4 - 0)] \text{ mod } 5 = -13 \text{ mod } 5 = 2 \text{ mod } 5 \rightarrow 2P = (4, 2)$

ESEMPIO :  $3P = (4, 2) + (0, 1)$  ( $P \neq 0$ )

• trovo  $\lambda = m = \frac{y_2 - y_1}{x_2 - x_1} \text{ mod } 5 = \frac{1}{4} \text{ mod } 5 \rightarrow 4 \cdot x \text{ mod } 5 \overset{?}{=} 1 \xrightarrow{x=4} 4 \text{ mod } 5 = \lambda$

• trovo  $x_3 = \lambda^2 - x_1 - x_2 \text{ mod } 5 = (16 - 0 - 4) \text{ mod } 5 = 12 \text{ mod } 5 = 2 = x_3$

• trovo  $y_3 = -[y_1 + \lambda(x_3 - x_1)] \text{ mod } 5 = -[2 + 4(2 - 4)] \text{ mod } 5 = 6 \text{ mod } 5 = 1$

$\rightarrow 3P = (2, 1)$

ESEMPIO:  $9P = (0,4) + (0,1)$  ( $P \neq \alpha$ )

$$\lambda = m = \frac{y_2 - y_1}{x_2 - x_1} \bmod 5 \quad \not\sim \quad 9P = \emptyset \sim |E(\mathbb{Z}_5)| = 9, \text{ sliegt da mod } 5$$

Considerazioni su EC group:  $(E(\mathbb{Z}_p), +)$  è gruppo abeliano



Nel campo crypto permette di avere:

→ dati  $[k]$ , point  $P \rightsquigarrow$  easy  $[k]P$ ; → dati point  $P$  e point  $[k]P \rightsquigarrow$  hard  $[k]$   
inoltre dipendente dalla curva scelta!

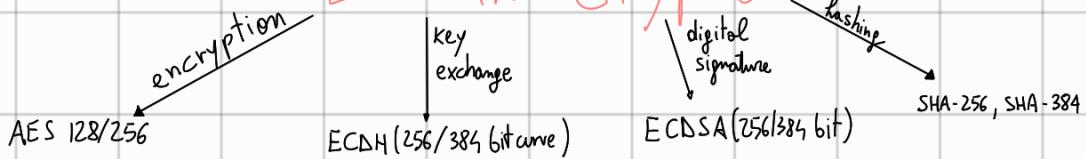
NB:  $[k]P \bmod P$  è ERRORE,  $[k]P$  è un punto, mentre modulo. Modulo applicato alle coordinate singole:  $x_i, y_i$

Inoltre, dato gruppo ordine  $\times$  (160 bit)  $\rightsquigarrow q = 2^{160}$ , le moltiplicazioni sono costose,  
ma con double & sum (= square & multiply) con efficienza  $1.5 \log_2 q$

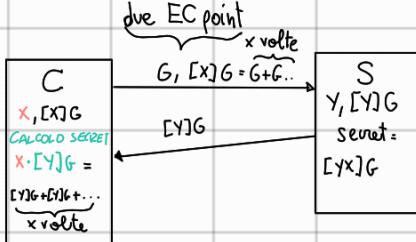
es:  $[1437]P \rightsquigarrow 1437_{10} = 10110011101_2 \rightsquigarrow \text{len} = 11 \text{ bit}$   
 $\rightsquigarrow \#`1` = 7$

	double	result
1	$1P$	$+1P = P$
0	$2P$	.
1	$4P$	$4P + 1P = 5P$
1	$8P$	$8P + 5P = 13P$
1	$16P$	$16P + 13P = 29P$
0	$32P$	.
:	⋮	⋮
	$(11-1)$ double	$(\#`1` - 1) \text{ result}$

# EC in Crypto



- ECDH : DH, con gruppi EC



- ECDSA: digital signature algorithm, "variante" di ELGAMAL (DLog based)

DSA senza EC:  $p=2q+1$ ;  $\text{PrivK}=d$ ;  $\text{PubK}=y=g^d \pmod p$

sign m: 1) nonce  $K \in \mathbb{Z}_q$ , NB:  $K < q$  prime  $\rightsquigarrow \exists K^{-1}$   
 (sender)

2)  $r = (g^K \pmod p) \pmod q$  } Nonce nascosto

3)  $s = \frac{H(m) + dr}{K} \pmod q$  } signature  $(r, s)$

verify: 2)  $m_1 = s^{-1} \cdot H(m) \pmod q = \frac{K \cdot H(m)}{H(m) + dr} \pmod q$  } manca 'd',  
 (receiver)

2)  $m_2 = s^{-1} r \pmod q = \frac{K \cdot r}{H(m) + dr} \pmod q$  } sta in PubKey!

3)  $\left[ g^{m_1} \cdot (g^d)^{m_2} \pmod p \right] \pmod q = \left[ g^{m_1 + d \cdot m_2} = g^{\frac{K(H(m) + dr)}{H(m) + dr}} \pmod p \right] \pmod q \stackrel{?}{=} r = (g^K \pmod p) \pmod q$

DSA con EC = ECDSA

"ingredienti":  $E(\mathbb{Z}_p)$ ,  $m$ ,  $P$ ,  $Q = [d]P$ ,  $[d]$

curve nascoste  
 prime group order  
 group generator  
 PubKey  
 Priv Key

SIGN :  $\text{mane } k \in [0, m-1] \rightsquigarrow [k]P = (x_1, y_1)$

$$\left[ \begin{array}{l} r = x_1 \bmod m \\ K^{-1} \bmod m \end{array} \right] \rightarrow \left[ \begin{array}{l} r = x_1 \bmod m \\ s = \frac{H(m) + dr}{K} \end{array} \right]$$

Verify (porto da 'm')

- calcolo  $\left\{ \begin{array}{l} \mu_1 = S^{-1} H(m) \\ \mu_2 = S^{-1} r \end{array} \right. = \left\{ \begin{array}{l} \frac{K H(m)}{H(m) + dr} \\ \frac{K r}{H(m) + dr} \end{array} \right.$  manca "d", pongo a EC
- $\mu_1 P + \mu_2 Q = [\mu_1 + d\mu_2] P = (x_0, y_0), \frac{K \cdot H(m)}{H(m) + dr} P + \frac{K \cdot dr}{H(m) + dr} P = [k]P$  mai rivelato!

allora il check è :  $r = x_1 \bmod m \stackrel{?}{=} x_0 \bmod m$  ?

What if ... K predetto, se valore di 's' e 'x<sub>1</sub>' → trovo "d"

K ripetuto ↳ ho  $\left\{ \begin{array}{l} S_1 = \frac{H(m_1) + dr}{K} \rightarrow K = \frac{H(m_1) + rd}{S_1} \bmod m \\ S_2 = \frac{H(m_2) + dr}{K} \rightarrow K = \frac{H(m_2) + rd}{S_2} \bmod m \end{array} \right. \right\}$  trovo "d"

# Bilinear Maps

- tool utile per il PAIRING (spostamento tra gruppi diversi, stabilendo relazioni)

definizione: Dati  $G_1, G_2, G_t$  <sup>= ISOMORFICI</sup> gruppi ciclici dello stesso ordine, ha bilinear map  $e: G_1 \times G_2 \rightarrow G_t$ , tale che:

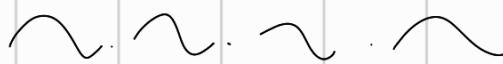
$$\begin{cases} u \text{ punto } \in G_1, [a], [b] \in \mathbb{Z} \\ v \text{ punto } \in G_2, \end{cases} \rightarrow e(u^a, v^b) = e(u, v)^{ab} \text{ (unica operazione), NON } e \text{ e' invertibile!}$$

Se uso  $\begin{cases} g_1 \text{ generatore di } G_1 \\ g_2 \text{ generatore di } G_2 \end{cases}$  ed  $e(g_1, g_2) \xrightarrow{\text{computabile}} g_t$  allora parlo di mappa ammmissibile. <sup>(caso di nostro interesse)</sup>

N.B.:  $e(aP, bQ) = e(P, Q)^{ab}$  e' altra notazione; NBB: normalmente  $G_1 = G_2 = G$

Proprietà di  $e$ :  $e(u \cdot v, g^s) = e(u, g^s) \cdot e(v, g^s)$  con  $\begin{cases} u = g^x \\ v = g^y \end{cases}$

$$e(u \cdot v, g^s) = e(g^x \cdot g^y, g^s) = e(g^{x+y}, g^s) = e(g, g)^{x+y s} = e(g^x, g) \cdot e(g^y, g) = e(g^x, g) \cdot e(g^y, g) \quad \checkmark$$



Decisional DH: dati  $g^a, g^b, g$  vedo  $\alpha \xrightarrow{\text{soi.}} g^{a+b}$  <sup>soi.</sup> rand non devo poterli distinguere, se riesco

Computational DH: dati  $g^a, g^b \xrightarrow[\text{non devo calcolare}]{\text{soi.}} g^{ab}$ , se ci riesco



Il Pairing aiuta con DDH?

$$\begin{cases} e(g^a, g^b) = g^{ab} \\ e(g, \alpha) = \xrightarrow{\text{soi.}} e(g, g^{ab}) = g^{ab} \\ \xrightarrow{\text{soi.}} e(g, g^{\text{rand}}) = g^{\text{rand}} \end{cases}$$

posso vedere se uguali,  
NON posso calcolare  $g^{ab}$ ,

PAIRING non riapplicabile nel dest. group

Il Pairing aiuta nelle EC a trovare ' $x$ '? (DLOG in E.C.)

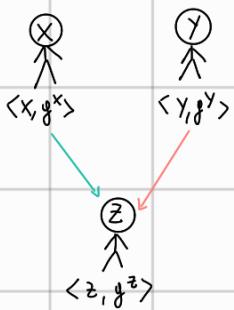
$$G^x = \text{RES}, x = ? \xrightarrow{\text{posso?}} \underbrace{G \times G}_{\text{EC group}} \xrightarrow{e} G_t \in (\mathbb{Z}_p^\times) \underline{\text{SI}}, e(\text{RES}, G) = e(G^x, G) = g_t^x$$

WEAKER GROUP

Ho usato man attack, passando il log. Problem da EC a campo finito

(" $e$ " e' funzione: mappa 2 punti  $\in$  EC  $\rightsquigarrow$  elementi im campo finito.)

# Third Party D.H.



• Posso avere  $g^{xyz}$ ? NO

• Scambio  $(x, z) \rightsquigarrow z$  riceve  $g^x \rightarrow (g^x)^z = g^{xz}$

• Scambio  $(y, z) \rightsquigarrow z$  riceve  $g^y$ , e poi? al massimo  $g^{yz}$ !

• Se usassi il Pairing?  $e(g^{xz}, g^y) = e(g, g)^{xyz} = g_t^{xyz}$  sembra OK!

• Con 4 persone?  $e(g_t^{xyz}, g^w) \neq g_t^{xyz}$   $\rightsquigarrow$  PAIRING utilizzabile solo una volta come "step finale"!



## Identity Based Encryption IBE (Boneh & Franklin)

GOAL: Vorrei Crypto System con PubK = nome ricordabile, scelto da me!

Normalmente PubK è derivato da PrivK (es:  $x \rightarrow g^x$  mod n im DH /  $(e, n)$  con 1cecm im RSA)

### INGREDIENTI:

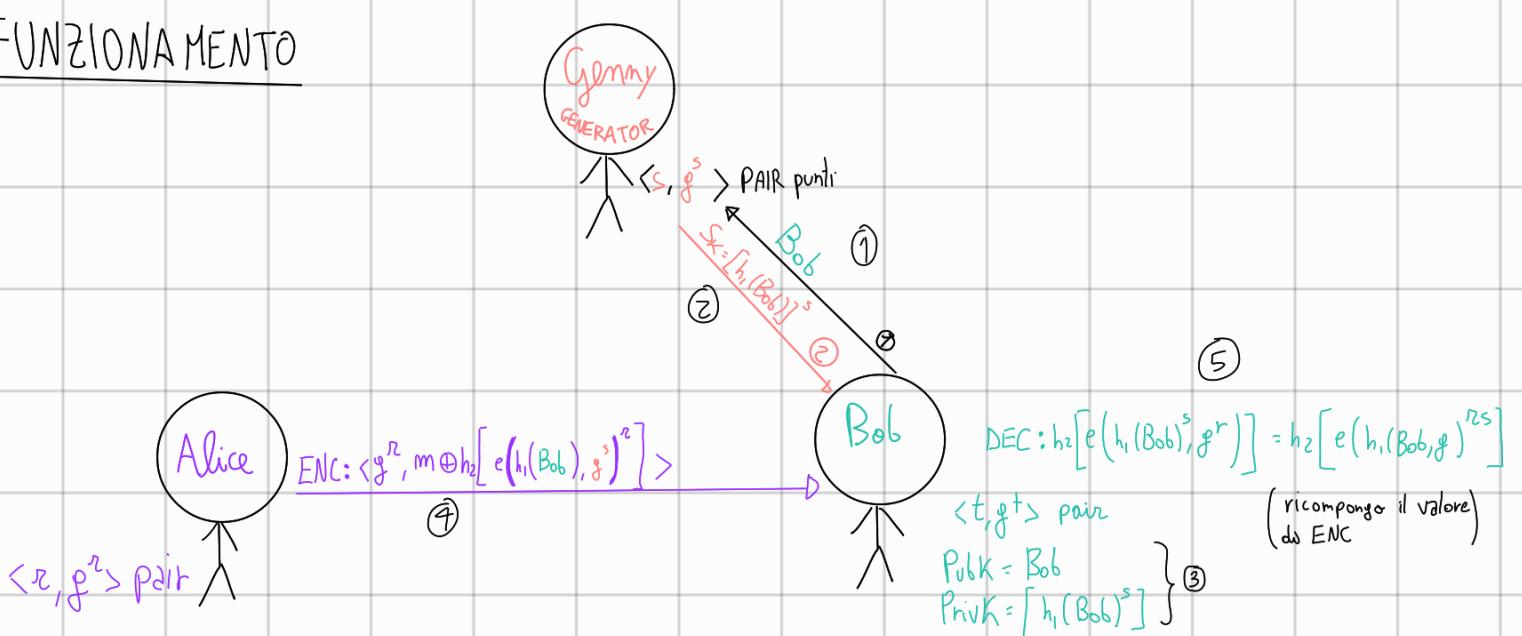
•  $G$  è EC prime order  $q$ , "g" generator

•  $e: G \times G \rightarrow G_t$ , "g" generator

$$\begin{cases} h_1: \{0,1\}^* \rightarrow G & (\text{str} \rightarrow \text{punto EC}) \approx \text{HASH} \\ h_2: G \rightarrow \{0,1\}^* & (\text{punto EC} \rightarrow \text{str}) \approx \text{SHA1} \end{cases}$$

• entità Generator che, dato ' $s$ '  $\rightarrow g^s$  PubKey  $\rightarrow \langle s, g^s \rangle$  pair di PKG

### FUNZIONAMENTO

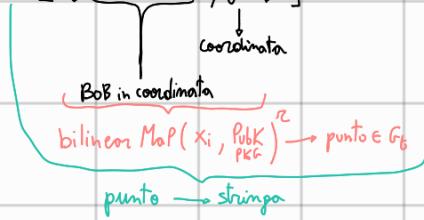


Poche' funziona? Normalmente forse  $e(g^s, g^x)^t = g^{rst}$ , ma NON ho t ~ fornisca già  $g^t = \text{Bob}$ , e uso un nuovo pair  $e([g^t]^s, g^r) = g^{rst}$

Nel dettaglio:

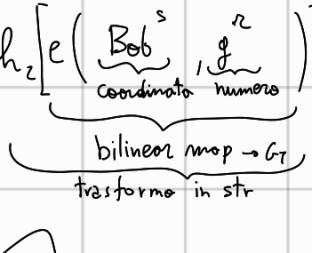
• PKG riceve 'Bob'  $\rightarrow h_1(\text{Bob})$  (diventa 'numero')  $\rightarrow \text{PrivK}_{\text{Bob}} = h_1(\text{Bob})^s$

• ENC: Alice sceglie  $r \rightarrow (g^r, m \oplus h_2[e(h_1(\text{Bob}), g^s)]^r)$



• DEC: Bob ha:  $\text{Bob}, \text{Bob}^s, g, g^r \rightarrow$  riceve  $(g^r, m \oplus str)$   $\rightarrow$  calcola  $str = h_2[e(\underbrace{\text{Bob}^s}_{\text{coordinate}}, \underbrace{g^r}_{\text{numero}})]$

$$\rightarrow m \oplus str \oplus str = m$$



## Ciphertext Policy Attribute Based Encryption (CP-ABE)

GOAL:  $\begin{cases} 1 \text{ PubKey} \\ 1 \text{ Master Private Key per generare } m \text{ private key ristrette (con "attributi")} \end{cases}$

Vorrei scalabilità, sicurezza, accessi flessibili, NON voglio una chiave  $\forall$  file, ma unica chiave con "attributi".

Vorrei evitare **collusion attack** (Se file richiede  $Sk_A \wedge Sk_C$ , Alice ha  $Sk_A$ , Bob ha  $Sk_C$ , non dovrebbero poter combinare le chiavi!)

### SOLUZIONE:

• aumento complessità di  $\begin{cases} \text{PubK} = (g, g^\beta, e(g, g)^\alpha) \\ \text{MSK} = (\beta, g^\alpha) \end{cases} \in \text{Authority}$

• Per le PrivKey:  $\begin{cases} \forall \text{ attributo "i" definisco stringa } X_i \text{ di } \emptyset \in 1 \sim (x_1, x_2, \dots, x_n) \\ \text{definisco random values } r_1, r_2, r_3, \dots, r_n \text{ associate a } X_i \end{cases}$

Allora  $Sk = \left( g^{(\alpha+r)/\beta}, \langle g^{r_{x_1}} H(x_1)^{r_{x_1}}, g^{r_{x_1}} \rangle, \dots, \langle g^{r_{x_n}} H(x_n)^{r_{x_n}}, g^{r_{x_n}} \rangle \right)$   
 dove "legame", NON COMBINABILI

# ALCUNI ESEMPI DI ESERCIZI

## ESEMPIO

- $p = 55$ , ho 5 partecipanti  $P_i$ ;  $P_1, P_3, P_5$  ricostruiscono regalo.

1) Lagrange Interpolation?

$$L_1 = \frac{x - x_3}{x_1 - x_3} \cdot \frac{x - x_5}{x_1 - x_5} = \frac{15}{(-2)(-4)} = [15 \cdot 8^{-1}] \bmod 55 = [15 \cdot 7] \bmod 55 = 50$$

$\uparrow$  modular inverse

$$L_3 = \frac{-1}{3-1} \cdot \frac{-5}{3-5} = \frac{5}{-4} \bmod 55 = (-14 \cdot 5) \bmod 55 = 40$$

$\uparrow$  inverse

$$L_5 = \frac{-x_1}{x_5 - x_1} \cdot \frac{-x_3}{x_5 - x_3} = \frac{3}{4 \cdot 2} \bmod 55 = 3 \cdot 8^{-1} \bmod 55 = [3 \cdot 7] \bmod 55 = 21$$

2) Ricostruisci regalo, con rhores:  $P_1 = 46, P_3 = 51, P_5 = 2$

$$S = (y_1 L_1 + y_2 L_2 + y_3 L_3) \bmod 55 = (50 \cdot 46 + 40 \cdot 51 + 2 \cdot 21) \bmod 55 = 37 \doteq S$$

3) Prova che è UNCONDITIONALLY sicure con  $P_3$  e  $P_5$ .

$$S = (D \cdot 50 + 40 \cdot 51 + 2 \cdot 21) \bmod 55 = (2082 + 50d) \bmod 55 = (47 + 50 \cdot D) \bmod 55$$

Ma  $D=0 \rightarrow S=47, D=1 \rightarrow S=42$ , molto di sì!

$\{47, 42, 37, 32, 27, 22, 17, 12, 7, 2, 52, \dots\}$  11 elementi, "solt" di 5  $\rightarrow$  fattore  $55 = 5 \cdot 11$

Con  $P$  primo non sarebbe successo!

## ESEMPIO

All' esempio del (3,4) scheme ricalcolo gli shares mod 101 (ho deciso  $0 \leq s \leq 100$ )

e trovo  $(1, 87), (2, 47), (3, 13), (4, 86)$ . Li non variano (dipendono da  $x_i$  piccolo = 1, 2, ...)

$$\lambda_1 = 3, \lambda_2 = -3, \lambda_3 = 1$$

$$s = [87 \cdot 3 + 47 \cdot (-3) + 13] \bmod 101 = 32, \text{ se share } (2, 47) \text{ nascosto?}$$

$$s = [274 - 3D] \bmod 101 = [72 - 3D] \bmod 101, \text{ in questo modo } s \text{ assume,}$$

al variare di  $D$ , TUTTI i valori nel range  $(0, 100)$ , **no leak!**