OSS: per la crittografia "non segreta" (asimmetrica / a chiave pubblica) in cui ALICE e BOB non dispongono di un metodo sicuro per comunicare prima della trasmissione del messaggio cifrato → NON POSSONO CONCORDARE UNA CHIAVE IN MODO SEGRETO

servono dei problemi tipo DLOG, DHP "particolari"

1) che siano difficili da risolvere senza informazioni aggiuntive
2) che siano calcolabili/risolvibili con un dato extra.

trapdoor problems (problemi/funzioni a botola)

Vogliamo per DH un gruppo in cui calcolare gli esponenziali è facile; calcolare DLOG è difficile

1) COME CALCOLARE GLI ESPONENZIALI? $n \geq 0$

$$\mathbb{Z}_n \to \langle g \rangle \qquad o(g) = n$$

$$[-t] = [n-t]$$

el. $\in \mathbb{N}$

$$exp(g,t) := \begin{cases} \text{se } t = 0 \to 1 \\ \text{se } t > 0 \to g \cdot exp(g, t-1) \end{cases}$$

si esegue il prodotto di g per se stesso t volte

equivalente (ma non eff.).

```
r = 1
for (i = 0; i <= t - 1; i++)
    r = g * r
restituire r
```

$g^8 = \overbrace{g \cdot g \cdot g \cdot g \cdot g \cdot g \cdot g \cdot g}^{8 \text{ operazioni}} = \overbrace{\left(\left(g^2\right)^2\right)^2}^{3 \text{ operazioni}}$

t operazioni $\quad \log_2 t$ operazioni

$g^7 = g^6 \cdot g = (g^3)^2 \cdot g = \underbrace{(g^2 \cdot g)^2 g}_{4 \text{ op.}}$

g^n

```
def pow(self, x, n):
    l = self.msb(n)
    C = 1
    for i in range(0, l + 1):
        C = self.prod(C, C)
        if ((n >> (l - i)) & 0x1) == 1:
            C = self.prod(C, x)
    return C
```

bit più significativo di n.

$C = C^2$

$C = C \cdot g$

$0b\,1\,0\,0\,1\,0$ (bit: 5 4 3 2 1 0)

$g^{\quad} =$

se l = bit più significativo = 1

ciclo 0 $(i=0)$

$c \leftarrow c^2$

$c \leftarrow g$

$0b\,1\,0\,0\,0$

$1+4+8+32$

$c = g^{\ell}$

$\exp(g, 0b\,1\,0\,1\,1\,0\,1) \quad 45$ (bit: 6 5 4 3 2 1 0)

$\ell = 6$

$c \quad g^0$

$c \leftarrow g$

$c \leftarrow c^2 \cdot g^2$ $\quad i=2$

$\downarrow$

g^2

$i=0 \quad g \longrightarrow g^2$

$i=1 \quad g^2 \longrightarrow g^2$

$i=2 \quad g^2 \longrightarrow g^4 \cdot g = g^5$

$i=3 \quad g^5 \longrightarrow g^{10} \cdot g$

```
function msb(x)
        local i=0
        t=copy(x)
        while t>0
          i+=1
          t>>=1
        end
        i
end

function bad_pow(n,x)
        n==0 && return 1
        x*bad_pow(n-1,x)
end

function better_pow(n,x)
        n==0 && return 1
        local c=1
        for i in msb(n):-1:0
          c *= c
          (((n>>i) & 1)==1) && ( c*= x)
        end
        c
end
```

$i = 4 \quad g^{11} \rightarrow g^{22}$

$i = 5 \quad g^{27} \rightarrow g^{44} \cdot g = g^{45}$

≤ 10 moltiplicazioni

esiste un algoritmo di complessità $O(\log n)$ per elevare un elemento di gruppo alla potenza n-esima $\rightarrow$ in effetti ci servono al più $2 \log_2 n$ operazioni di gruppo $\rightarrow$ $x \rightarrow x^2$ oppure $x \rightarrow x \cdot g$!

Che gruppi ciclici considerare?

OSS: Sia G un gruppo e supponiamo $|G| = p$ con p numero primo.

$\Rightarrow \forall g \in G$, $\langle g \rangle$ ha ordine che divide p.

in particolare se $g \neq 1 \Rightarrow o(g) = |\langle g \rangle| \neq 1$

e quindi necessariamente $o(g) = p$.

→ IDEALMENTE SI VORREBBERO GRUPPI CICLICI DI ORDINE primo

1) perché ogni elemento $\neq 1$ è un generatore
2) perché non ci sono sottogruppi non banali

→ IN GENERALE SI USANO sottogruppi di ordine primo per i DLOG basati su curve ellittiche

→ altrimenti si hanno sottogruppi di ordine $p-1$

o di un primo p che divide $p^n - 1$ con p primo. $(n = 1)$

Sottogruppi moltiplicativi di campi finiti.

1) un campo $(\mathbb{K}, +, \cdot)$ è una struttura algebrica dove $(\mathbb{K}, +)$ gruppo commutativo, $(\mathbb{K} \setminus \{0\}, \cdot)$ gruppo commutativo e valgono le prop. distributive: $\forall \alpha, \beta, \gamma \in \mathbb{K}$: $(\alpha + \beta)\gamma = \alpha\gamma + \beta\gamma$

$\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma$

$|\mathbb{K}| < \infty \Rightarrow$ campo finito

$(\mathbb{Z}_2, +, \cdot)$ $(\mathbb{Z}_3, +, \cdot)$ $(\mathbb{Z}_4, +, \cdot)$ NON È UN CAMPO.

·	1	2
1	1	2
2	2	1

$2 \cdot 2 = 0$

	1	2	3
1	1	2	3
2	2	0	2
3	3	2	1

1) $(\mathbb{Z}_n, +, \cdot)$ è un campo finito $\Leftrightarrow$ n è un numero primo

2) Il gruppo moltiplicativo $(\mathbb{K}^*, \cdot) = (\mathbb{K} \setminus \{0\}, \cdot)$, $|\mathbb{K}| = p^n$ di un campo finito è sempre un gruppo ciclico $\rightarrow$ ed è un gruppo ciclico in cui, per p^n abbastanza grande il DLOG è difficile.

QUESTI GRUPPI CICLICI SONO USATI

OSS 1) Sia $\mathbb{K}$ un campo con $|\mathbb{K}| = n \Rightarrow n$ è potenza di un primo.

2) Sia p un primo $\Rightarrow \mathbb{Z}_p$ è un campo

→ ALGORITMO EUCLIDEO

DOBBIAMO DIMOSTRARE CHE $\forall [\alpha]_p \in \mathbb{Z}_p \setminus \{[0]\}$ esiste $[\beta]_p$ tale che $[\alpha]_p \cdot [\beta]_p = [\alpha\beta]_p = [1]_p$

cioè che ogni elemento non nullo è invertibile.

$\forall \alpha \in \mathbb{Z} \setminus p\mathbb{Z} \; \exists \beta \in \mathbb{Z} \setminus p\mathbb{Z}$ tali che

$$\alpha\beta \equiv 1 \mod p \Rightarrow (\alpha\beta \,\%\, p) = 1$$

cioè

$$\alpha\beta = 1 + kp \quad \text{per } k \in \mathbb{Z}$$

ovvero $\boxed{\alpha\beta + kp = 1}$ ammette soluzione in $\mathbb{Z}$

↗ RISOLVERE QUESTA EQ. DIOFANTEA → negli interi.

MASSIMO COMUN DIVISORE FRA 2 INTERI α, β

DEF: Siano α, β due interi si dice massimo comun divisore fra α e β ogni intero d tale che se $c|\alpha$ e $c|\beta$ $(c \in \mathbb{Z}) \Rightarrow c|d$

È un intero che è diviso da tutti i divisori comuni fra α e β.

oss se $x \mid y \Rightarrow |x| \leq |y|$ $\Rightarrow$ un MCD ha modulo massimo fra tutti i divisori comuni fra α e β.

Lemma: $\forall \alpha, \beta \in \mathbb{Z}$, $(\alpha, \beta) \neq (0,0)$, esistono 2 $MCD(\alpha,\beta)$ che differiscono a meno del segno.

$d \mid \alpha$, $d \mid \beta$ e $\forall c$: $c \mid \alpha, c \mid \beta \Rightarrow c \mid d$ $\rightarrow$ pongo $c = d'$

$d' \mid \alpha$, $d' \mid \beta$ e $\forall c'$ $c' \mid \alpha, c' \mid \beta \Rightarrow c' \mid d'$ $\quad c' = d$.

$\Rightarrow d \mid d'$ e $d' \mid d \Rightarrow d = kd'$ e $d' = hd$

$\Rightarrow d = hkd \Rightarrow hk = 1$ ma questo in $\mathbb{Z}$ è possibile $\Leftrightarrow (h,k) = (1,1)$ oppure $(h,k) = (-1,-1)$ □

per convenzione $MCD(\alpha,\beta)$ è l'$MCD > 0$

Teorema (ALG EUCLIDEO)

1) $MCD(\alpha,\beta) = MCD(\beta,\alpha)$
2) $MCD(\alpha,0) = \alpha$
3) $MCD(\alpha,\beta) = MCD(\alpha \% \beta, \beta)$

DIM 3) Sia $d = MCD(\alpha,\beta)$ e supponiamo

$$\alpha = \beta q + r$$

$\Rightarrow d \mid \alpha$, $d \mid \beta \Rightarrow d \mid (\beta q + r)$, $d \mid \beta$

$\beta q + r = dh \qquad \beta = dk \Rightarrow$

$\Rightarrow r = dh - (dk)q = d(h - kq) \Rightarrow d \mid r$

OGNI DIVISORE COMUNE FRA α e β é divisore COMUNE FRA β ed r

VICEVERSA Se $d \mid r$ e $d \mid \beta \Rightarrow r = td \quad \beta = dk$

$\Rightarrow \alpha = \beta q + r = dkq + dt = d(kq + t)$

$\Rightarrow d$ é divisore comune fra α e β. □

ALGORITMO

1) $MCD(\alpha, \beta) = MCD(|\alpha|, |\beta|)$

2) Se $\alpha < \beta \Rightarrow$ ricor. $MCD(\beta, \alpha)$

3) Se $\beta = 0 \Rightarrow$ restituisco α

4) restituisco $MCD(\beta, \alpha \% \beta)$

N.B $0 \leq \alpha \% \beta < \beta$

Teorema: l'equazione diofantea

$$\alpha x + \beta y = \gamma$$

é RISOLUBILE $\Leftrightarrow MCD(\alpha, \beta) \mid \gamma$

$\Rightarrow$ IN TALE CASO L'ALGORITMO EUCLIDEO (ESTESO) FORNISCE UNA SOLUZIONE IN TEMPO POLINOMIALE.

D.M: Se $d \mid \alpha$ e $d \mid \beta \Rightarrow d \mid (\alpha x + \beta y) \ \forall x, y$

e quindi se l'eq. é risolubile $d \mid \gamma$ □

```
def Euclidean(u, v):
    print("(", u, ",", v, ")")
    if v > u:
        return Euclidean(v, u)
    while v != 0:
        r = u % v
        u = v
        v = r
        print("=(", u, ",", v, ")")
    return u
```

$u = qv + r$

DIMOSTRIAMO CHE DATI $\alpha, \beta \in \mathbb{Z}$

$\exists\, x, y$ tali che

$$\alpha x + \beta y = MCD(\alpha, \beta)$$

e quindi se $MCD(\alpha, \beta) \mid \gamma \Rightarrow \gamma = h\, MCD(\alpha, \beta)$

e $\bar{x} = hx \quad \bar{y} = hy$

risolvono $\alpha x + \beta y = \gamma$

$MCD(71, 13) \rightarrow 71 = 13 \cdot 5 + 6 \qquad 6 = 71 - 13 \cdot 5$

$MCD(13, 6) \rightarrow 13 = 6 \cdot 2 + 1 \rightarrow 1 = 13 - 6 \cdot 2$

$MCD(6, 1) \rightarrow 6 = 1 \cdot 6 + 0$

$MCD(1, 0) \rightarrow 1$

$1 = 13 - (71 - 13 \cdot 5) \cdot 2$

$\Rightarrow 1 = 13 \cdot (11) + 71 \cdot (-2)$

```
function my_gcd(x,y)
    x>y || return my_gcd(y,x)
    y==0 && return x
    @show (x,y)
    my_gcd(y,mod(x,y))
end
```