In 2024.11 Donal Trump declared America to be a Bitcoin country.
Possibly inspired by Elon Musk.

Anonymity in Blockchain

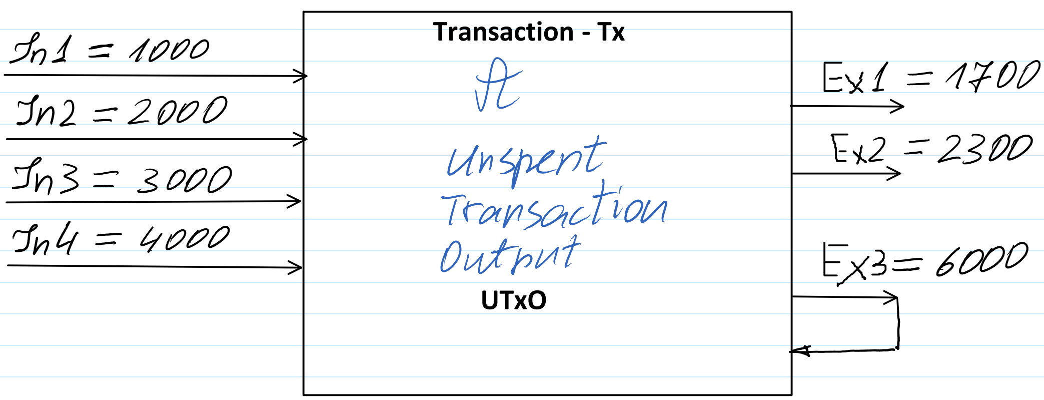
Let **Alice** opened her Bitcoin account with Bitcoin Address by generating her private key **PrK**=x and public key **PuK**=a.
We assume that **PuK**=a are linked to **Alice** Aaddress in Bitcoin.

In Bitcoin and other Blockchains the Address is computed as a function of user's public key:
Addr$_A$ = **F**(**PuK**) and consist of several dozens of decimal numbers.

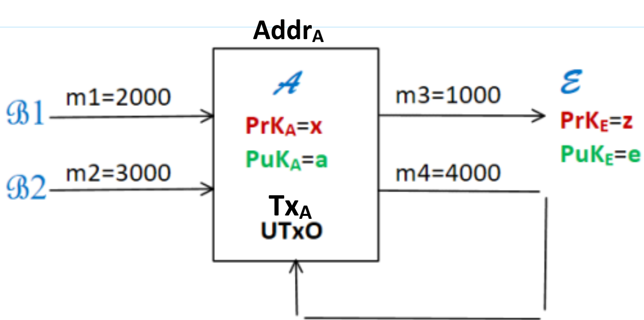
Cryptocurrency transaction

No.	Pajamos-Incomes	Išlaidos-Expenses	Likutis-Balance
In1.	Client1: 1000 Sat		1000 Sat
In2.	Client2: 2000 Sat	Out1. Firm 5: 1700 Sat	1300 Sat
In3.	Client3: 3000 Sat	Out2.t Firm 6: 2300 Sa	2000 Sat
In4.	Client4: 4000 Sat	Out3. Firm 7:	6000 Sat
Total	**10 000 Sat**	**4000 Sat**	**6000 Sat**

Sum of Inputs =
= Sum of Outputs
Divisibility

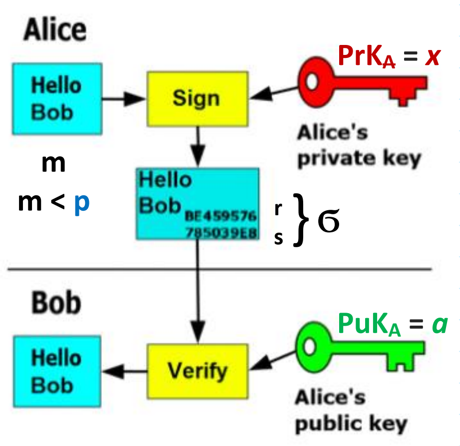


Transaction (Tx) information in simplified form consist of the following information:
1. The address of Tx creator.
2. The sums of Incomes and addresses of senders.
3. The sums of Expenses and addresses of receivers.



Schnorr Signature

In the case of Schnorr cryptosystem our simulation is performed with Public Parameters:
PP = (p, g); **p=268435019; g=2**; p=int64(268435019)
By having **PP** private key **PrK** and public key **PuK** are generated:
PrK = x <-- randi(p-1)
PuK = $a = g^x$ **mod** p.



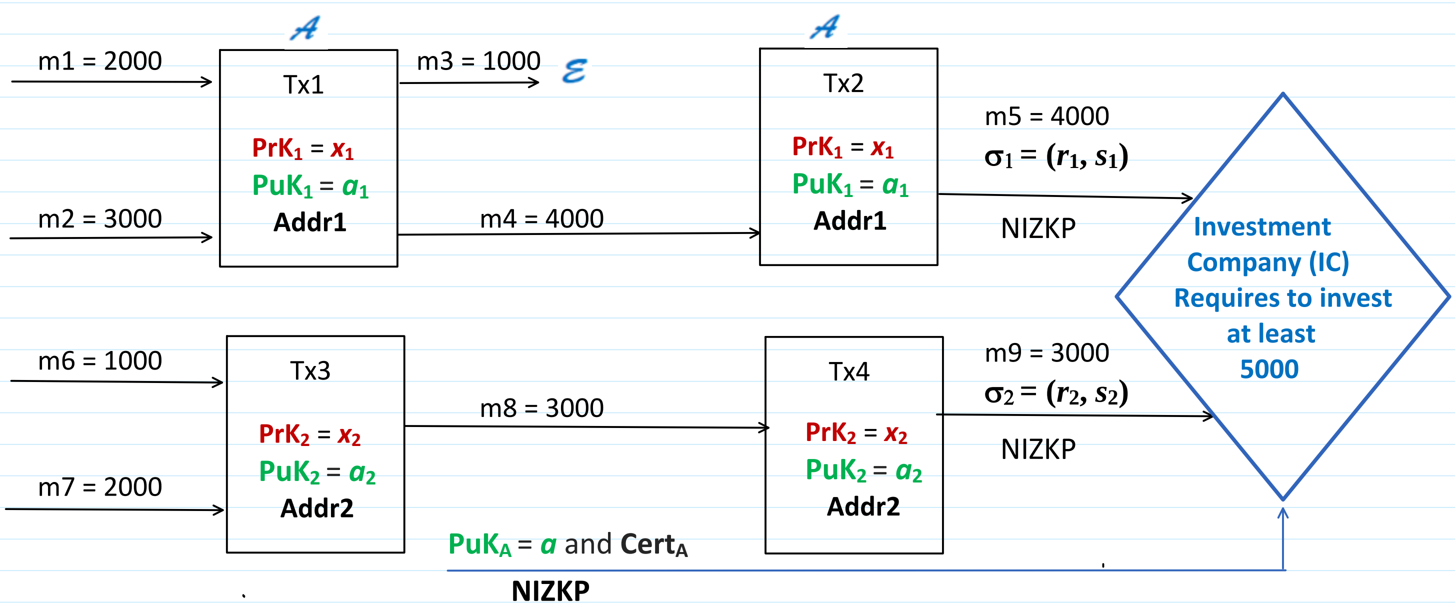
u <-- randi(p-1).
$r = g^u$ **mod** p.
h = H($M||r$). >> con=concat(M,r)
>> h=hd28(con)
$s = u + xh$ **mod** (p-1). (*) >> s=mod(u+x*h,p-1)
Alice's signature on h is $\sigma = (r, s)$.

Notice that it is infeasible to find x from (*), when
s and h are given, since there is 1 equation (*) and 2 unknowns u and x.

Signature is valid if: $\underbrace{g^s \text{ mod } p}_{V1} = \underbrace{ra^h \text{ mod } p}_{V2}$. (Eq.1)

But **Alice** do not want that all her incomes belonging to her Address were known and therefore and she prefers to be anonymous to the **Net**.
Then she creates a set of Addresses by generating a set of private keys {$PrK_i = x_i$} and a set of public keys {$PuK_i = a_i$}, where i=1, 2, ..., N.

But! There are the situations when **Alice** must prove some subjects that she possesses some amount of money distributed among a lot of her accounts and transactions with different addresses.
For example, she could pretend to tax concessions - mokesčiu lengvatos (according to the law) and she must prove to certain Investment Company that she possesses sufficient amount of money.
In this case she must prove that she controls some accounts with this sufficient amount of money for investment.
In this case **Alice** can prove that her transactions are authentic (i.e. are created by her) by proving that **PuK**=a belongs to her, e.g. using Certificate issued by Certificate Authority for **PuK**=a, but at the same time she remains **anonymous** for other part of the **Net**.



Schnorr-Multi-Signature
Deanonymization in BlockChain

Group of signers (**GoS**) must sign on different transactions.
Let the **GoS** is: $\{S_1; S_2\}$.
All members of **GoS** have their private and public keys:

S_1;	S_2;
$\mathbf{PrK_1}=z_1$, $\mathbf{PuK_1}=a_1$;	$\mathbf{PrK_2}=z_2$, $\mathbf{PuK_2}=a_2$;
$u_1 \leftarrow$ randi$(p\text{-}1)$;	$u_2 \leftarrow$ randi$(p\text{-}1)$;
$r_1=g^{u1} \bmod p$;	$r_2=g^{u2} \bmod p$;
$h_1=\mathrm{H}(Tx2\|\|r_1)$;	$h_2=\mathrm{H}(Tx4\|\|r_2)$;
$s_1=u_1 + x_1h \bmod (p\text{-}1)$;	$s_2=u_2 + x_2h_2 \bmod (p\text{-}1)$;

$\sigma_1 = (r_1, s_1)$. $\sigma_2 = (r_2, s_2)$.

$\sigma_{12} = \sigma_1 * \sigma_2 = (r_1, s_1)*(r_2, s_2) = (R_{12}, S_{12})$.
$R_{12} = r_1*r_2 \bmod p = g^{u1} * g^{u2} \bmod p$.
$S_{12} = s_1+s_2 \bmod (p\text{-}1) = [(s_1=u_1 + x_1h_1) + (s_2=u_2 + x_2h_2)] \bmod (p\text{-}1)$.

GoS signature verification:

$$\underbrace{g^{S_{12}} \bmod p}_{V1} = \underbrace{R_{12} * (a_1)^{h1} * (a_2)^{h2} \bmod p}_{V2}. \quad \text{(Eq.2)}$$

$$\underbrace{g^s \bmod p}_{V1} = \underbrace{ra^h \bmod p}_{V2}. \quad \text{(Eq.1)}$$

Correctness:

$$g^{S_{12}} \bmod p = g^{(s1 + s2) \bmod (p-1)} \bmod p = g^{s1 \bmod (p-1)} * g^{s2 \bmod (p-1)} \bmod p = g^{(u1 + x1*h1) \bmod (p-1)} * g^{(u2 + x2*h2) \bmod (p-1)} \bmod p =$$

$$= r_1 * (a_1)^{h1} * r_2 * (a_2)^{h2} \bmod p =$$
$$= r_1 * r_2 * (a_1)^{h1} * (a_2)^{h2} \bmod p =$$
$$= R12 * a1^{h1} * a2^{h2} \bmod p.$$

Anonymity disclosing for Investment Company (IC)
Deanonymization for Authenticity

Alice has private key $\mathbf{PrK_A} = x$ and public key $\mathbf{PuK_A} = a = g^x \bmod p$.
Alice has a certificate $\mathbf{Cert_A}$ issued by Certificate Authority (CA) on her $\mathbf{PuK_A} = a$.
Alice must prove to IC that $\mathbf{PuK_1} = a_1$ and **Addr1** and $\mathbf{PuK_2} = a_2$ and **Addr2** belongs to her.
Then she must sign h-value: $H' = \mathrm{H}(a_1 || \mathbf{Addr1} || a_2 || \mathbf{Addr2})$ using her $\mathbf{PuK_A} = a$ and $\mathbf{Cert_A}$.

$u \leftarrow$ randi$(p\text{-}1)$.
$r = g^u \bmod p$.
$h = \mathrm{H}(H'||r)$.
$s = u+xh \bmod (p\text{-}1)$.

$$\xrightarrow{\sigma = (r, s)}$$
$\mathbf{PuK_A} = a$; $\mathbf{Cert_A}$
$\mathbf{Addr_A}$; **NIZKP**

IC verifies $\mathbf{Cert_A}$ and Alice signature
$$\underbrace{g^s \bmod p}_{V1} = \underbrace{ra^h \bmod p}_{V2}.$$

If verification passes then **IC** transfers the interest on investments to **Alice** account.
The material regarding **NIZKP** is included in schemes and explanation is presented below.

Additional material: of Non-Interactive Zero Knowledge Proof (**NIZKP**).

The technique presented above has an essential flaw.
The anyone having **PrK**$_{Im}$ = x_{Im} and public key **PuK**$_{Im}$ = a_{Im} can impersonate the actual **Addr1** and **Addr2** holder and redirect the interest on investments to his/her account by creating new **Addr**$_{Im}$ = **F**(**PuK**$_{Im}$) by obtaining the certificate **Cert**$_{Im}$ on **PuK**$_{Im}$.
Then **Impersonator** having Tx2 and Tx4 data together with **PuK**$_1$ = a_1 and **PuK**$_2$ = a_2 can sign Tx2 and Tx4 with his/her **PrK**$_{Im}$ by computing $\sigma_{Im} = (r_{Im}, s_{Im})$.
Then **Impersonator** sends ($\sigma_{Im} = (r_{Im}, s_{Im})$, **PuK**$_{Im}$ = a_{Im}, **Cert**$_{Im}$ and **Addr**$_{Im}$) as actual **Addr1** and **Addr2** holder **Alice** did.
After **IC** verification the **Impersonator** is waiting when **IC** transfers the interest on investments to his/her account represented by **Addr**$_{Im}$.

The solution of this problem is the realization of Non-Interactive Zero Knowledge Proof (**NIZKP**) by Alice proving that she knows her generated **PrK**$_1$ = x_1 and **PrK**$_2$ = x_2.
The **NIZKP** is as an additional operation is included in the schemes above.
The details of **NIZKP** realization is left as an exercise.

Anonymity and authenticity simulation

```
>> p=int64(268435019)
p = 268435019
>> g=2;
```

```
>> x=int64(randi(p-1))
x = 257726155
>> a=mod_exp(g,x,p)
a = 32920391
>> AddrA=hd28('32920391')
AddrA = 126423499
```

```
>> x1=int64(randi(p-1))
x1 = 156758073
>> a1=mod_exp(g,x1,p)
a1 = 15617773
>> Addr1=hd28('15617773')
Addr1 = 32691790
```

```
>> x2=int64(randi(p-1))
x2 = 93240757
>> a2=mod_exp(g,x2,p)
a2 = 92735335
>> Addr2=hd28('92735335')
Addr2 = 186632019
```

```
Tx2='In21=4000||Ex21=4000||Addr1'

>> u1=int64(randi(p-1))
u1 = 50037375
>> r1=mod_exp(g,u1,p)
r1 = 32904517
>> con=concat(Tx2,r1)
con = In21=4000||Ex21=4000||Addr132904517
>> h1=hd28(con)
h1 = 64943318
>> s1=mod(u1+x1*h1,p-1)
s1 = 234649183
```

```
Tx4='In41=3000||Ex41=3000||Addr1'

>> u2=int64(randi(p-1))
u2 = 190308111
>> r2=mod_exp(g,u2,p)
r2 = 22463608
>> con=concat(Tx4,r2)
con = In41=3000||Ex41=3000||Addr122463608
>> h2=hd28(con)
h2 = 26322703
>> s2=mod(u2+x2*h2,p-1)
s2 = 61742096
```

```
>> R12=mod(r1*r2,p)
R12 = 92919544
>> S12=mod(s1+s2,p-1)
S12 = 27956261
>> g_S12=mod_exp(g,S12,p)
```

```
>> a1_h1=mod_exp(a1,h1,p)
a1_h1 = 168145239
>> a2_h2=mod_exp(a2,h2,p)
a2_h2 = 55254133
```

```
>> R12ma1_h1=mod(R12*a1_h1,p)
R12ma1_h1 = 241090947
>> R12ma1_h1ma2_h2=mod(R12ma1_h1*a2_h2,p)
R12ma1_h1ma2_h2 = 91640974 = 91640974 = V2
```

g_S12 = **91640974 = V1**

Schnorr-Multi-Signature is valid since V1 = V2 = **91640974**

Deanonymization against IC

```
> con1=concat(a1,Addr1)
con1 = 1561777332691790
>> con2=concat(a2,Addr2)
con2 = 92735335186632019
>> con12=concat(con1,con2)
con12 = 156177733269179092735335186632019
>> HH=hd28(con12)
HH = 150477396
```

```
>> u=int64(randi(p-1))
u = 218160208
>> r=mod_exp(g,u,p)
r = 76047239
>> conHHr=concat(HH,r)              %   H'||r
conHHr = 15047739676047239          % HH==H'
>> h=hd28(conHHr)
h = 114895503
>> s=mod(u+x*h,p-1)
s = 107897009
```

$\underbrace{g^s \bmod p}_{V1} = \underbrace{ra^h \bmod p}_{V2}$.   (Eq.1)

```
> g_s=mod_exp(g,s,p)
g_s = 18634187
>> V1=g_s
V1 = 18634187
```

```
>> a_h=mod_exp(a,h,p)
a_h = 202702734
>> V2=mod(r*a_h,p)
V2 = 18634187
```

Till this place