De Cifris Trends in *Cryptographic Protocols*

University of Trento and *De Componendis Cifris*

October 2023

Lecture 6

Claudio Orlandi, Aarhus University, Denmark


A gentle introduction to:
Threshold Cryptography
&
Secure Multiparty Computation

Claudio Orlandi

AARHUS UNIVERSITY

DANMARKS FRIE FORSKNINGSFOND
INDEPENDENT RESEARCH FUND DENMARK


European Research Council
Established by the European Commission

COBRA
CONCORDIUM BLOCKCHAIN RESEARCH CENTER AARHUS

Example: Online Poker

2♠, 5♠ ,2♥,5 ♥,J♦

Q♠,Q♣,7♣,3♥,2♦,

10 ♠,9♣, 8♣,7♦,6♦

3♠, 4♠,7♥,Q ♦,10♦

Poker with Pirates

2♠, 5♠ ,2♥,5 ♥,J♦

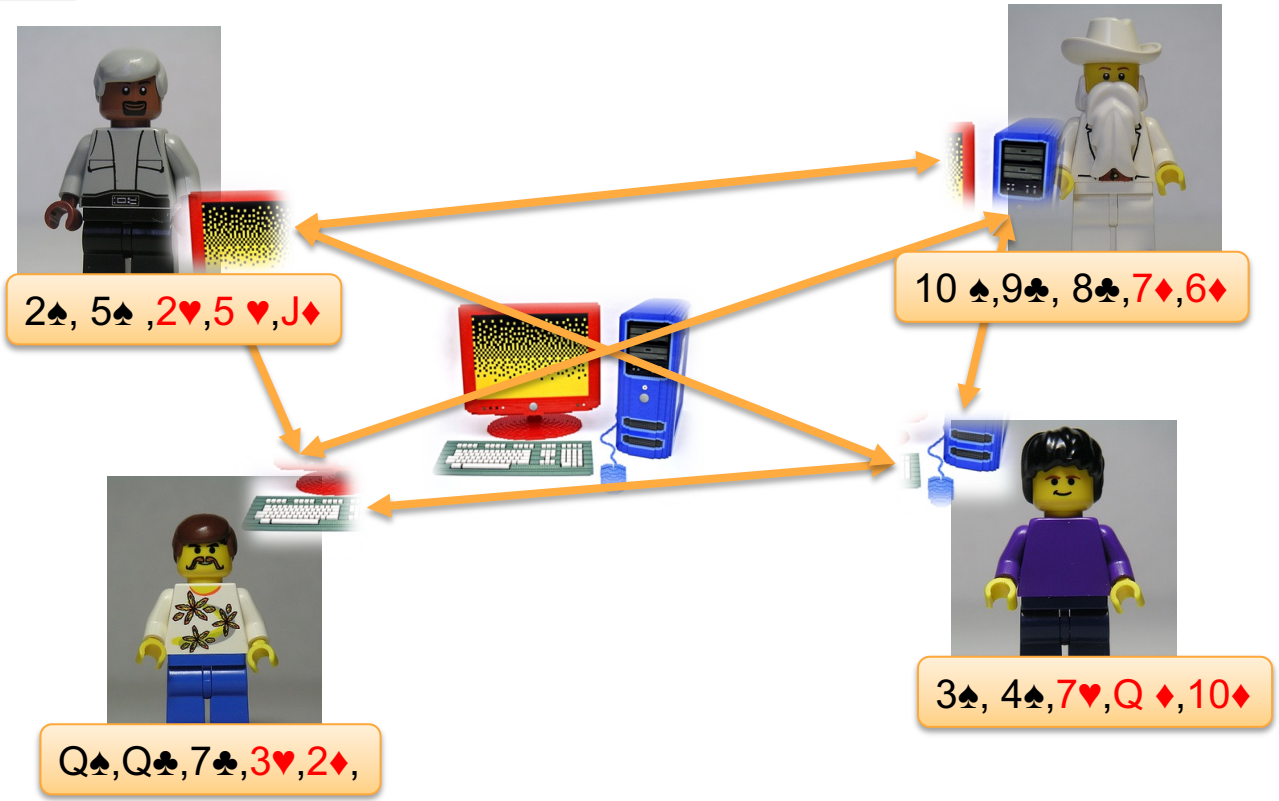
Q♠,Q♣,7♣,3♥,2♦,

10 ♠,9♣, 8♣,7♦,6♦

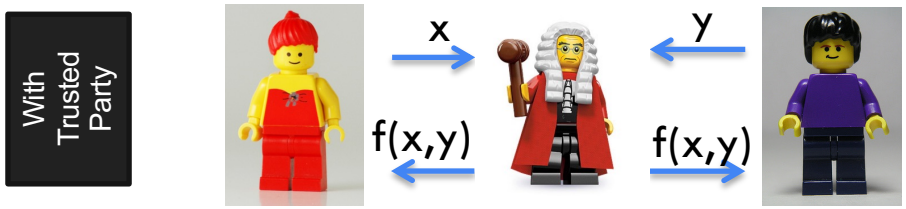
A♠,A♣,A♥,A♦,K♦

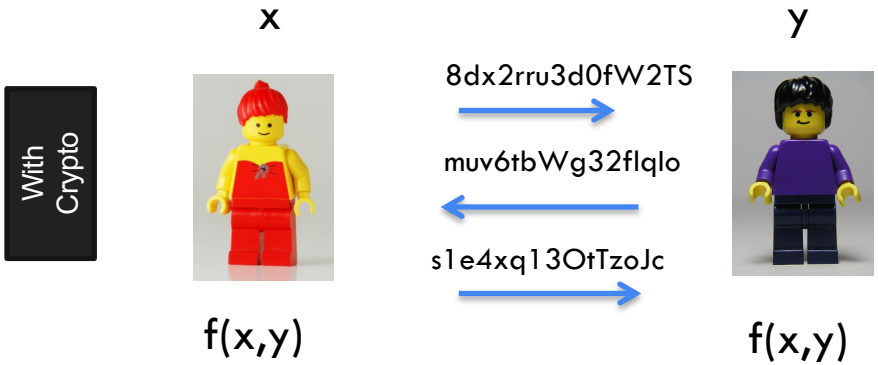
4

Secure Multiparty Poker



Secure Multiparty Computation (MPC)

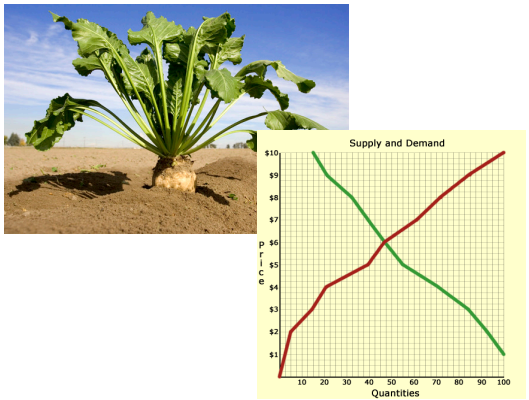




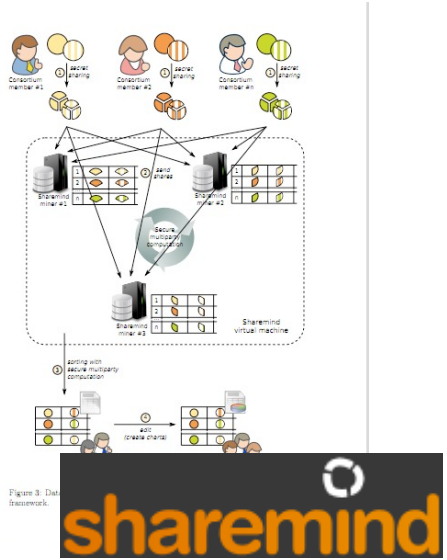
- *Privacy*
- *Correctness*
- *…*

MPC in Practice, Examples

"MPC Goes Live", Jan 2008
Secure Auction for Agriculture Production Rights in Denmark



Estonian study on student dropout



Boston women workforce council,
study on wage gap

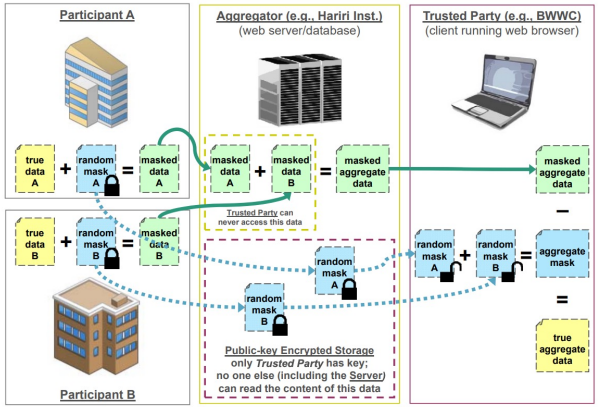


Figure 1: Illustration of a deployment of the protocol implementation for two participants.

Threshold Cryptography: Security and Availability in Cryptocurrencies

DEC. 23, 2013

Bloomberg TV Anchor Gives Bitcoin As a Christmas Gift, Immediately Has It Stolen

By Kevin Roose

The shortest-lived gift in Christmas history. Photo: Bloomberg TV

MONEY

Man accidentally threw away $127 million in bitcoin and officials won't allow a search

Published Wed, Dec 20 2017•12:30 PM EST • Updated Wed, Dec 20 2017•3:24 PM EST

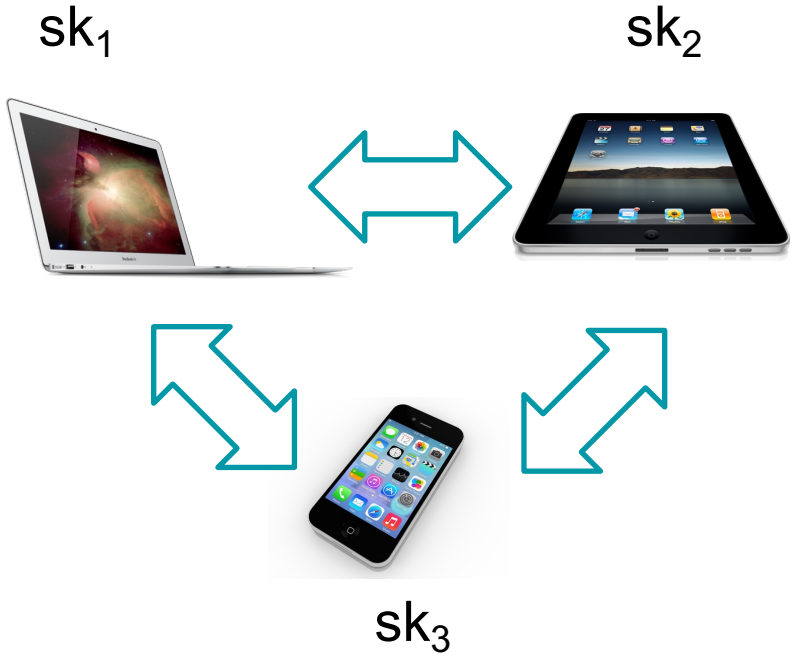
Shawn M. Carter
@SHAWNCARTERM

Share

Trending Now

1. These 20 companies around the world are on a hiring spree for work-from-home jobs
2. 17-year-old discovers planet 6.9 times larger than Earth on third day of internship with NASA
3. I tried 'Kakeibo': The Japanese art of saving money—and it completely changed how I spend

Threshold Cryptography: the idea



- The secret key is “shared” across e.g., three devices
- One share gives no information about the key
- Two shares are enough to compute a signature on any message
- Foundations from the 80s-90s, renewed interest due to Blockchains
- Can be seen as a special case of MPC where the shares are the secret inputs and signing/decrypting is the joint function.

Threshold Signatures: Industry

(Linear) 3-out-of-3 Secret Sharing

- Secret x in some range (0…M-1)
- Pick random x_1, x_2, x_3 from (0…M-1) such that

$$x_1 + x_2 + x_3 \bmod M = x$$



- We write [x] for short

- x can only be recovered with all 3 shares
- Good for privacy
- Bad for availability

(Linear) 3-out-of-3 Secret Sharing

- Given [x], [y] it is very easy to compute [z]=[x+y]

x_1	+	x_2	+	x_3	mod M	=	x
+		+		+			
y_1	+	y_2	+	y_3	mod M	=	y
=		=		=			
z_1	+	z_2	+	z_3	mod M	=	x+y



(Replicated) 2-out-of-3 Secret Sharing

- Secret x in some range (0…M-1)
- Pick random x_1, x_2, x_3 from (0…M-1) such that $x_1+x_2+x_3$ m od M = x

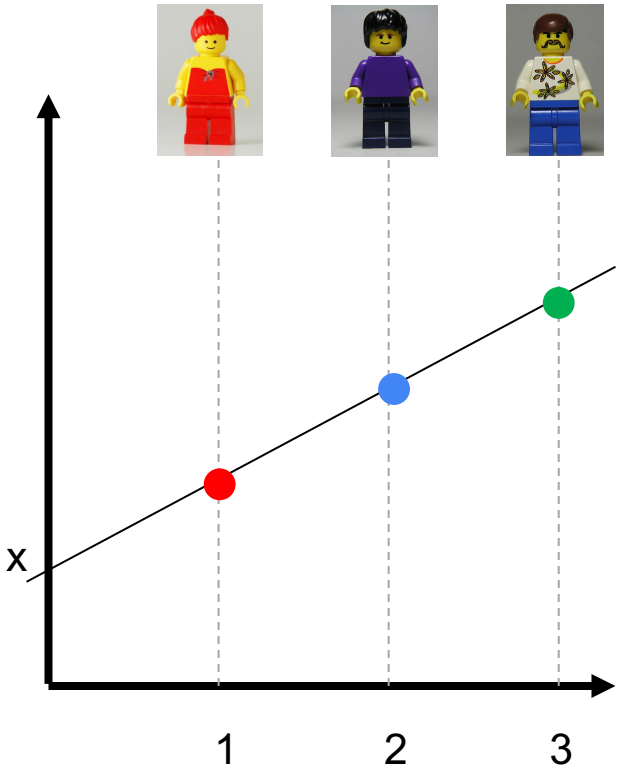
(x_1,x_2) (x_2,x_3) (x_3,x_1)



- Still linear!
- Scales very poorly with the number of parties:
  - 3-out-of-5: 6 shares per party!
  - 5-out-of-11: 210 shares per party!
  - m-out-of-n: (n-1)!/((m-1)!(n-k)!)

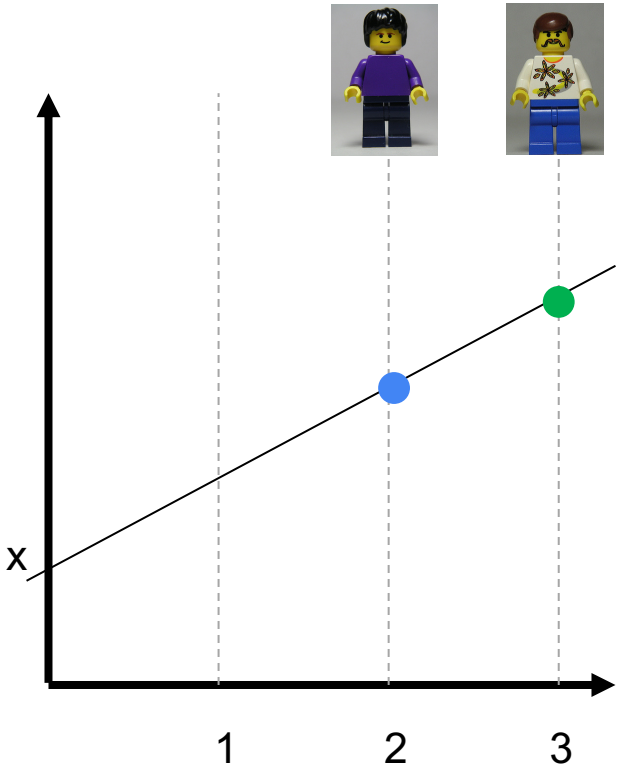
Shamir Secret-Sharing

- Secret x in (0…q-1), q prime.
- Choose random degree-d polynomial f s.t. x=f(0)
- Party i gets a single share f(i)
  - Independent of threshold and number of parties!
- In the example:
  - $f(\alpha) = x + r\,\alpha \bmod q$ (for some random r)
  - P_1 gets f(1), P_2 gets f(2), P_3 gets f(3)



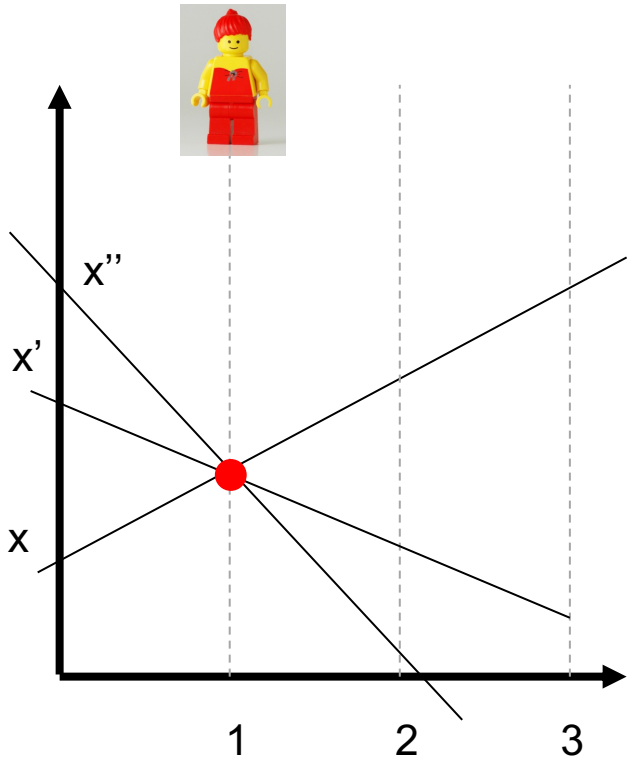
Shamir Secret-Sharing

- Since f(α) has degree d
- Any set of d+1 parties can reconstruct x
- Any set of at most d parties has no information about x



Shamir Secret-Sharing

- Since f(α) has degree d
- Any set of d+1 parties can reconstruct x
- Any set of at most d parties has no information about x



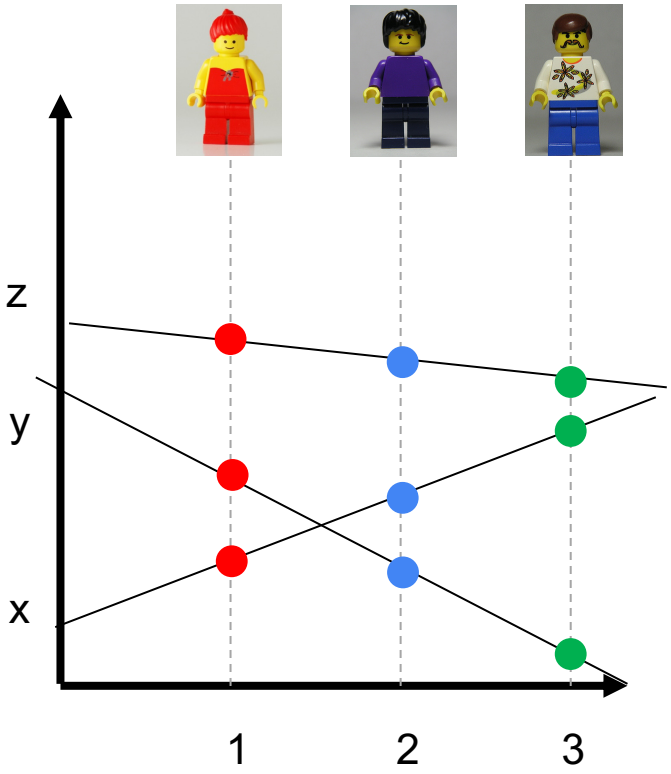
Shamir Secret-Sharing

- Addition can still be performed locally

- In the example:
- $f(\alpha) = x + r\,\alpha \bmod q$ (for random r)
- $g(\alpha) = y + s\,\alpha \bmod q$ (for random s)

- P_1 computes $h(1)=f(1)+g(1) \bmod q$
- P_2 computes $h(2)=f(2)+g(2) \bmod q$
- P_3 computes $h(3)=h(3)+g(3) \bmod q$

- h has degree d, and $h(0)=x+y \bmod q$




BLS Signatures

Ingredients

- A bilinear map e(xP,yQ)=e(P,Q)^(xy)
- All groups are of prime order q and are written using the additive notation.
- A cryptographic hash function H mapping messages into the first group.

KeyGen:

- sk : a random value in 0..q-1
- pk = sk*Q

Sign(sk,m)

- Output σ = sk*H(m)

Verify(pk, m, σ)

- Accept iff e(σ,Q)=e(H(m),pk)

Threshold BLS Signatures

Key Generation:

- $sk = sk_1 + sk_2 + sk_3$ (or [sk])
- pk = sk*Q

ThresholdSign(sk_i, m)

- Output $\sigma_i = sk_i * H(m)$ (or $[\sigma]=[sk]*H(m)$)

Combine(σ_1, σ_2, σ_3)

- Output $\sigma = \sigma_1 + \sigma_2 + \sigma_3$

Verify(pk, m, σ)

- Accept iff e(σ,Q)=e(H(m),pk)

Generalizes to:

- Any number of parties and threshold with any linear secret sharing schemes from earlier
- (With caveats) to any signature scheme in which the secret key is only used in a "linear" way (e.g., RSA, Schnorr, …)

ECDSA Signatures

- P is a point on an Elliptic Curve
- Q is the public key s.t. Q=sk*P

ECDSASign(sk,m)

1. Pick random k
2. $(r_x,r_y)=kP$
3. $s = 1/k * (H(m) + r_x*sk)$
4. Output (r_x,s)

We need to perform a non-linear computation of two secrets, sk and k!

Verification: compute $(t_x,t_y)= H(m)*P+r_x*Q)/s$, and accept if $t_x=r_x$


Threshold ECDSA with 2 multiplications

ECDSASign(sk,m)

1. Pick random k
2. $(r_x,r_y)=kP$
3. $s = 1/k * (H(m) + r_x*sk)$
4. Output (r_x,s)

ThresholdECDSASign([sk],m)

1. Generate random share [k]
2. $(r_x,r_y) \leftarrow$ Open([k]*P)
3. Generate random share [x]
4. Compute [y]=Mul([k],[x])
5. $y \leftarrow$ Open([y])
6. Compute [1/k]=[x]/y
7. Compute [sk/k]= Mul([sk],[1/k])
8. $s \leftarrow$ Open([1/k]*H(m)+ r_x*[sk/k])
9. Output (r_x,s)

Multiplication with Replicated Secret Sharing

- [z]=Mul([x],[y])

(x_1,x_2) (x_2,x_3) (x_3,x_1)



Goal: compute

$$z = (x_1+x_2+x_3)(y_1+y_2+x_3)$$

$$= \underbrace{x_1y_1 + x_2y_1 + x_3y_1}_{P_1} + \underbrace{x_1y_2 + x_2y_2 + x_3y_2}_{P_2} + \underbrace{x_1y_3 + x_2y_3 + x_3y_3}_{P_3}$$

Each party can locally compute some of the products (then one round of resharing for security)

Multiplication with Shamir Secret Sharing

[z]=Mul([x],[y]) :

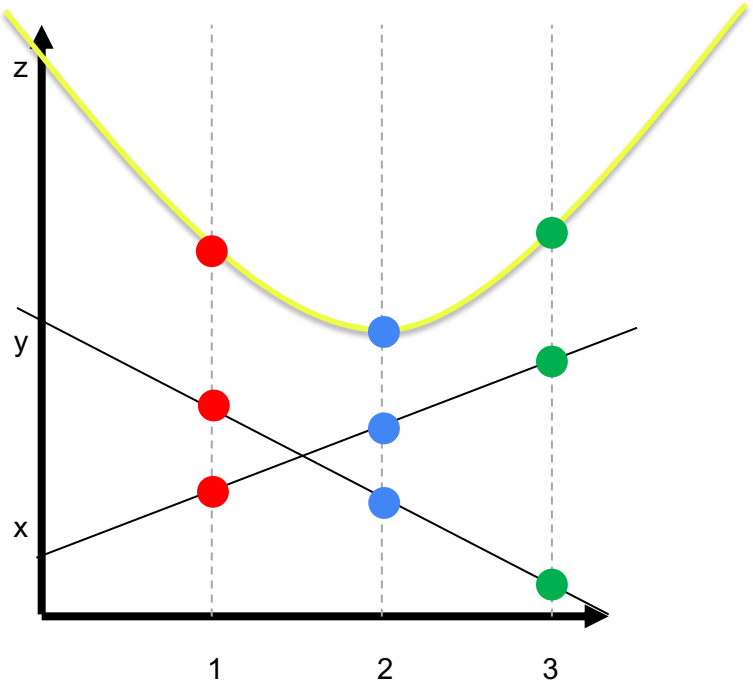
- x=f(0), y=g(0)
- $f(\alpha) = x + r*\alpha$
- $g(\alpha) = y + s*\alpha$

- P_1 computes $h(1)=f(1)*g(1)$
- P_2 computes $h(2)=f(2)*g(2)$
- P_3 computes $h(3)=f(3)*g(3)$

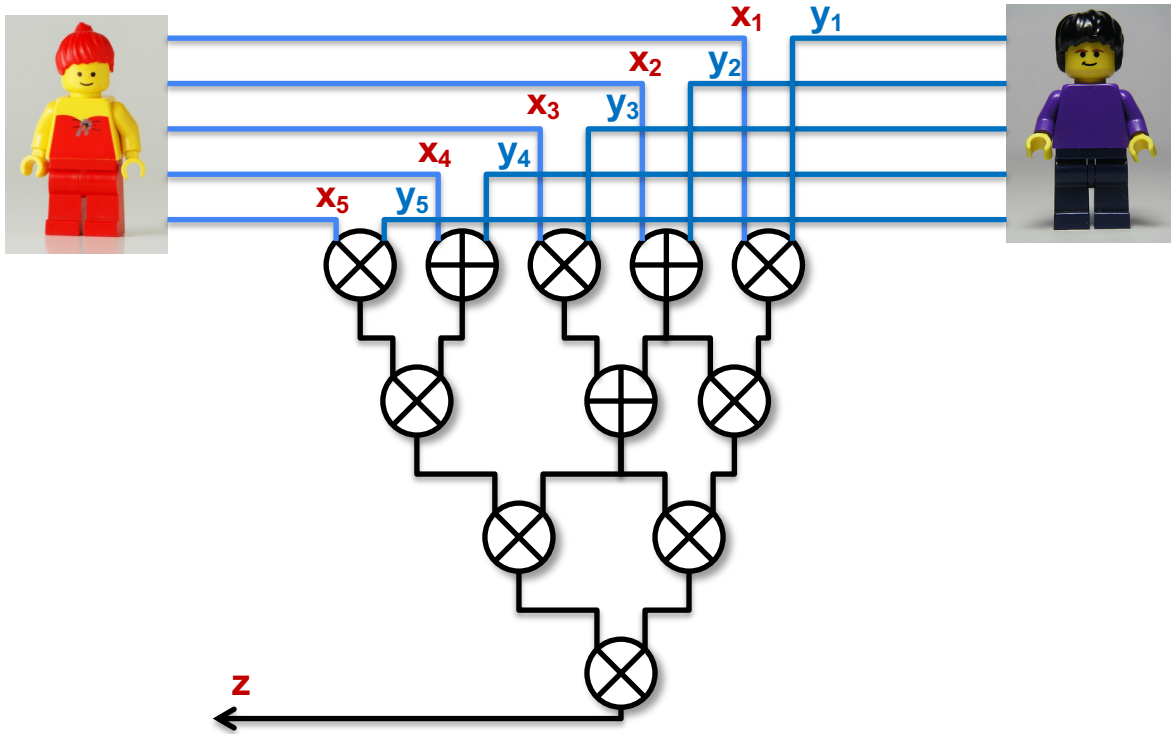
h(0)=xy (as desired)
But t has the wrong degree!

- $h(\alpha) = xy + h_1*\alpha + h_2*\alpha^2$

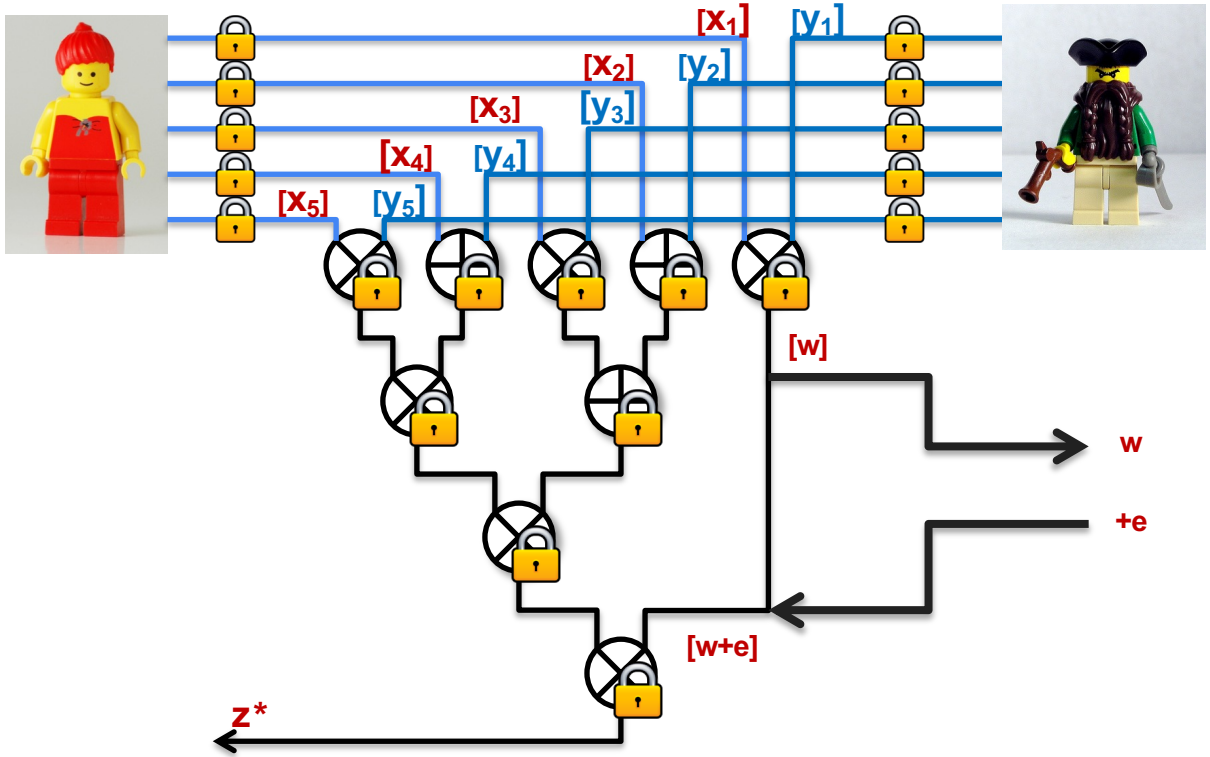
Solution: parties ”reshare” their results with the right degree and perform (linear) reconstruction in the shared domain.



Circuit based computation



Secure Computation of Any Function with Linear Secret Sharing




Take Aways

- Secret-Sharing MPC can be used to securely evaluate any function.
  - Linear operations are free!
  - Multiplications require more work.
- Threshold Cryptography can be seen as a special example of MPC, providing meaningful tradeoffs between availability and confidentiality of the secret key.
  - Many natural schemes are compatible with linear secret sharing techniques.


Do you want more?

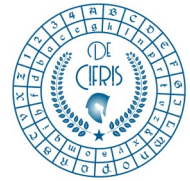
- Basic topics not covered here:
  - Secure Two-party computation (or in general MPC vs. n-1 corruptions)
  - Garbled Circuits, Homomorphic Encryption, Oblivious Transfer, …
  - Active security: Zero-Knolwedge, homomorphic MACs, …
  - Special purpose protocols (e.g., PSI, …)
  - See e.g., my lecture notes on “Cryptographic Computing”

- Active topics of research
  - MPC with sublinear communication complexity (PCG, FSS, HSS, …)
  - Threshold Cryptography for Post-Quantum Scheme
  - ….

De Componendis Cifris

https:/www.decifris.it

Claudio Orlandi **Aarhus University, Denmark**

Shamir Reconstruction

- Given p(1), p(2) one can reconstruct p(x) as

  $p(\alpha)=\delta_1(\alpha)p(1)+\delta_2(\alpha)p(2)$

- $\delta_i(\alpha)$ is a poly s.t.

  $\delta_i(i)=1$

  $\delta_i(j)=0$ for all j in the reconstruction set (except i)

In our case
$\delta_1(\alpha)= (\alpha -2)(1-2)^{-1}$
$\delta_2(\alpha)= (\alpha -1)(2-1)^{-1}$

To reconstruct secret enough to compute

$$p(0)=\delta_1(0)p(1)+\delta_2(0)p(2)$$

This is a linear computation and therefore we can also perform it easily in the secret-shared domain

$$[p(0)]=\delta_1(0)[p(1)]+\delta_2(0)[p(2)]$$

(Generalizes to any other degree)

Multiplication with Shamir Secret Sharing

[z]=Mul([x],[y]) (part 2):

- $[z_1] \leftarrow$ Input(P_1,t(1))
- Symmetric for P_2, P_3
- Then reconstruct i.e.

$[t(0)]=\delta_1[t(1)]+\delta_2[t(2)]+\delta_2[t(3)]$

  - But t(0)=z, so we're done!

- Exercise: find the the values $\delta_1,\delta_2,\delta_3$
  (Hint, the degree is different this time!)