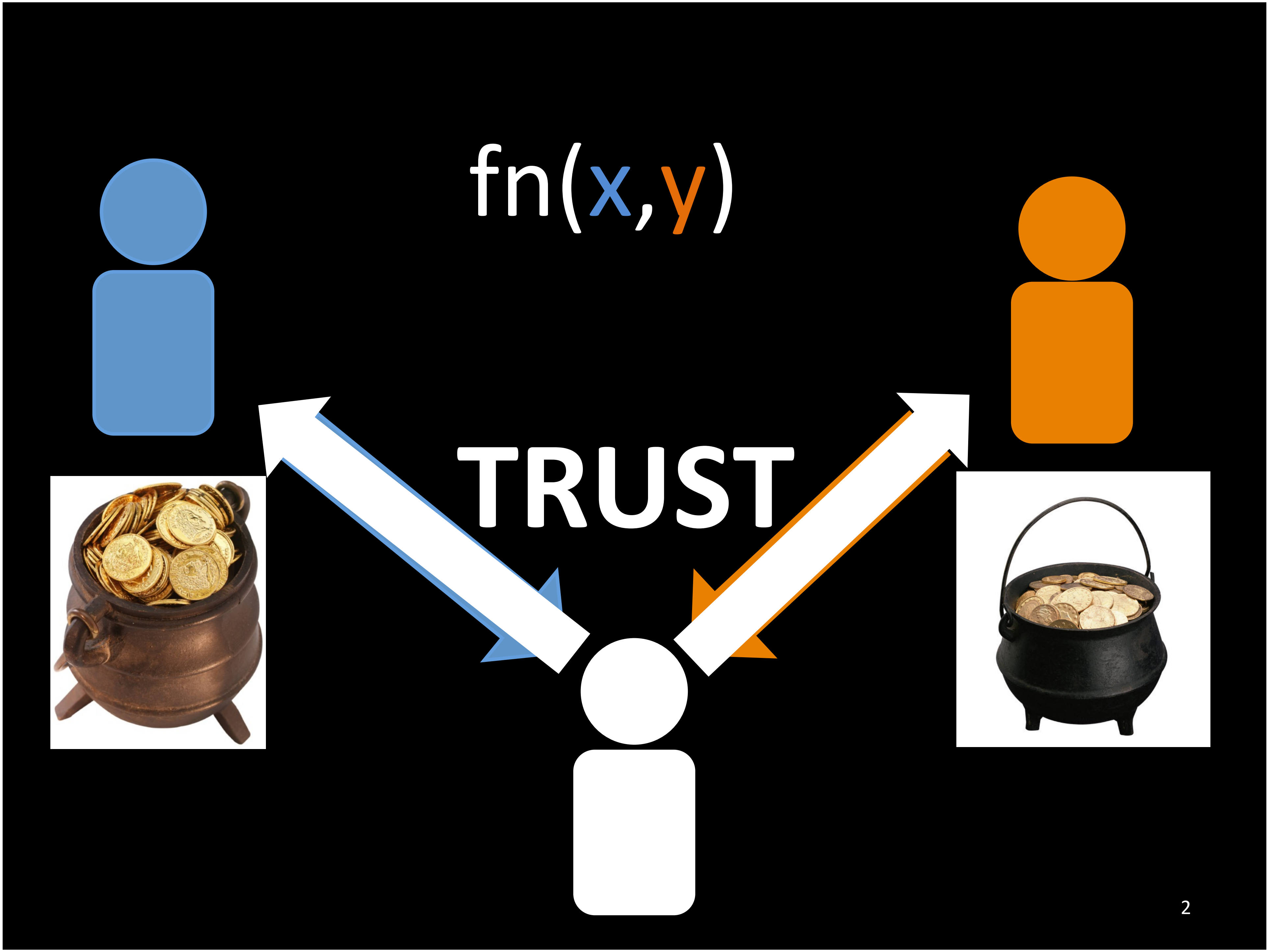
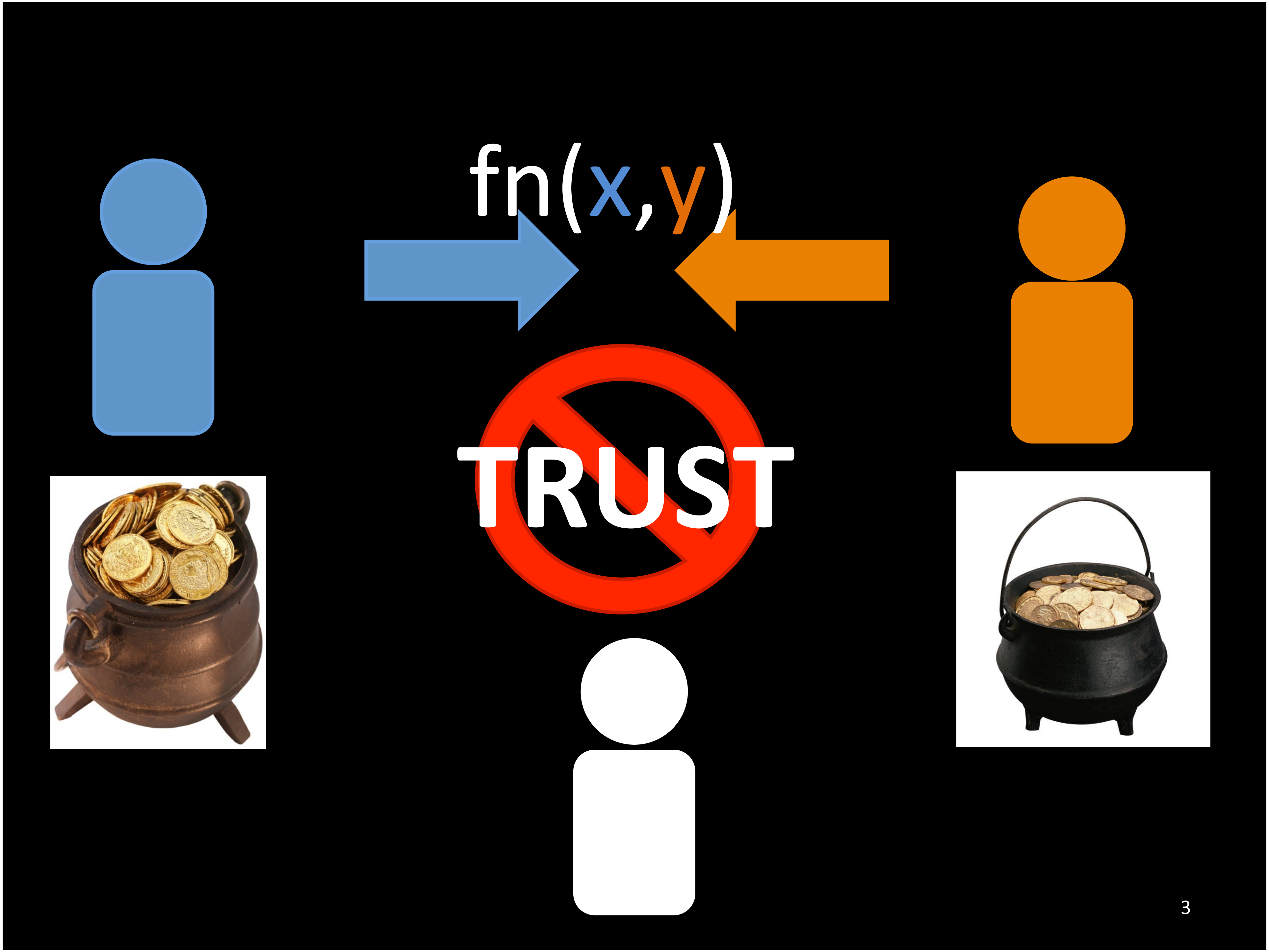
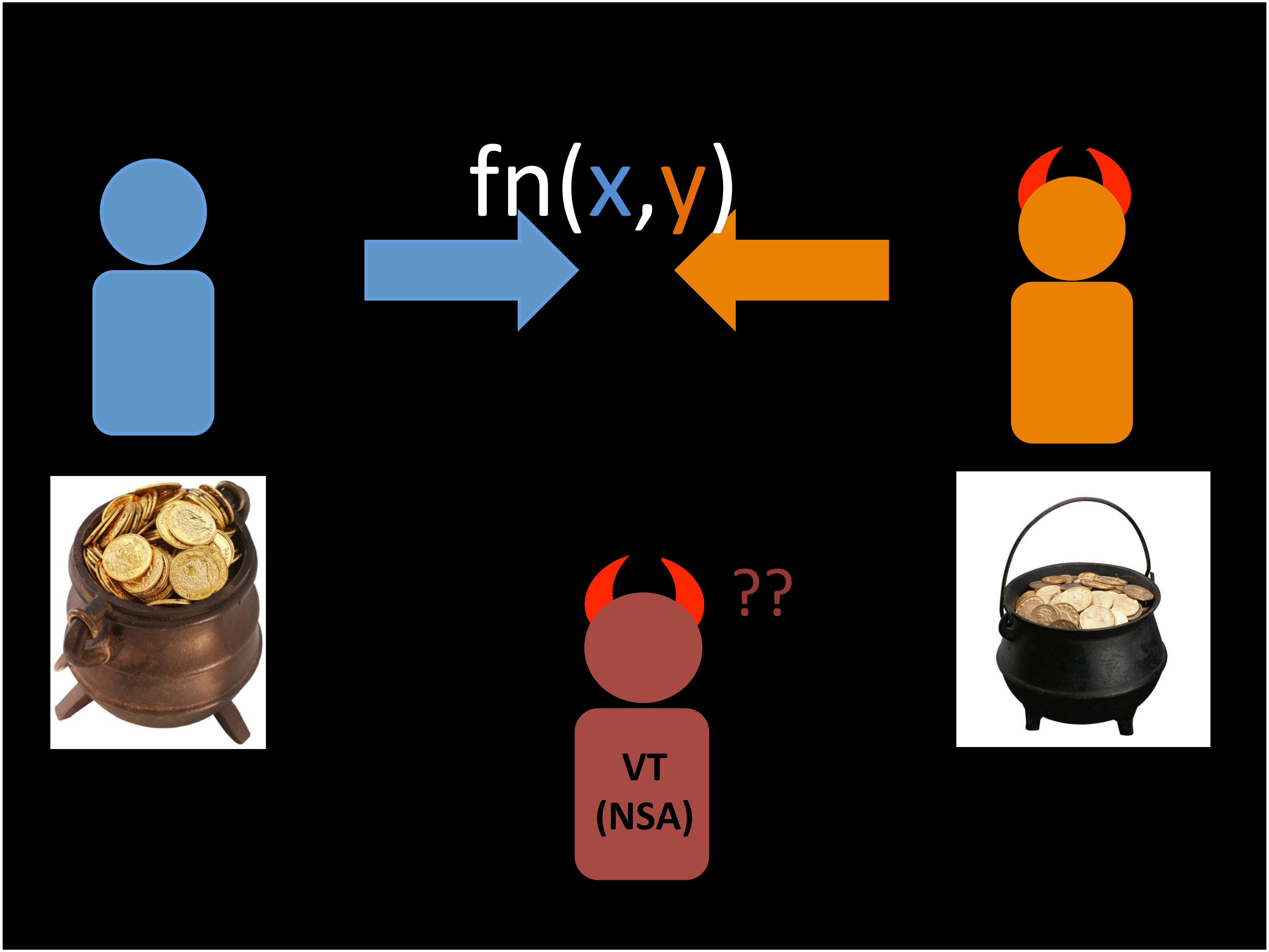
Garbled Circuits: An Intro

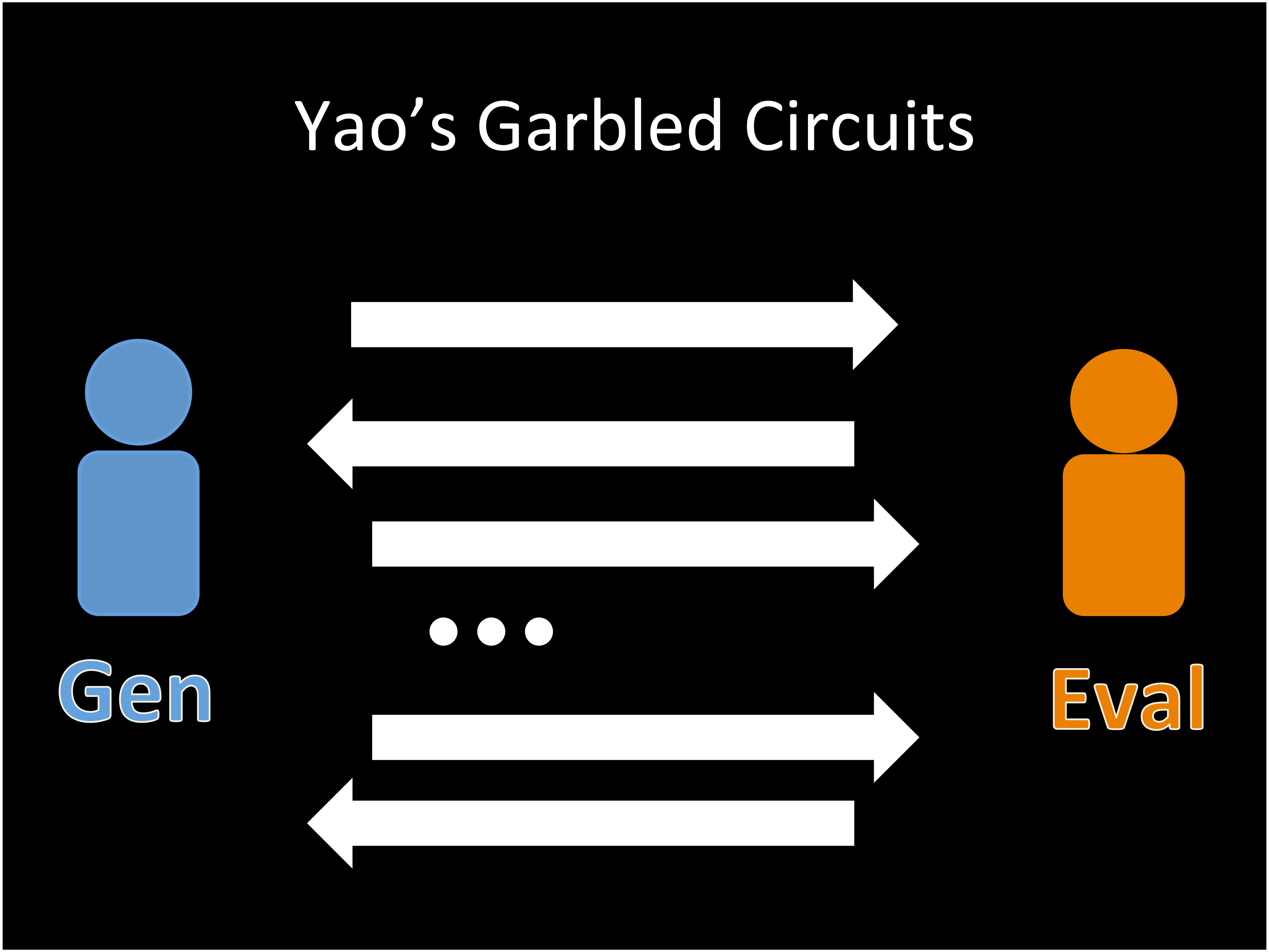
Ben Terner

CS6501 - Cryptography


fn(x,y)
TRUST

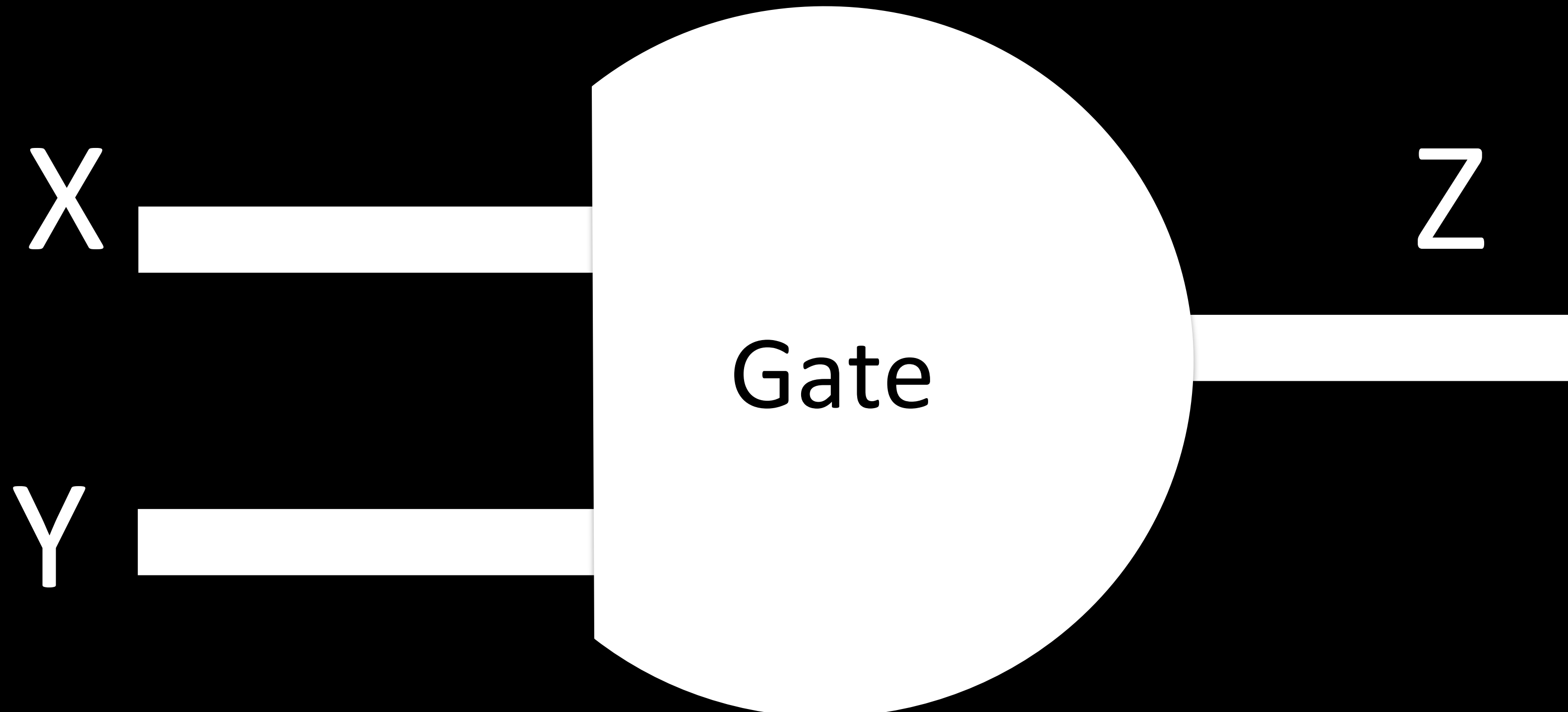

fn(x,y)
TRUST
3


fn(x,y)
??
VT
(NSA)

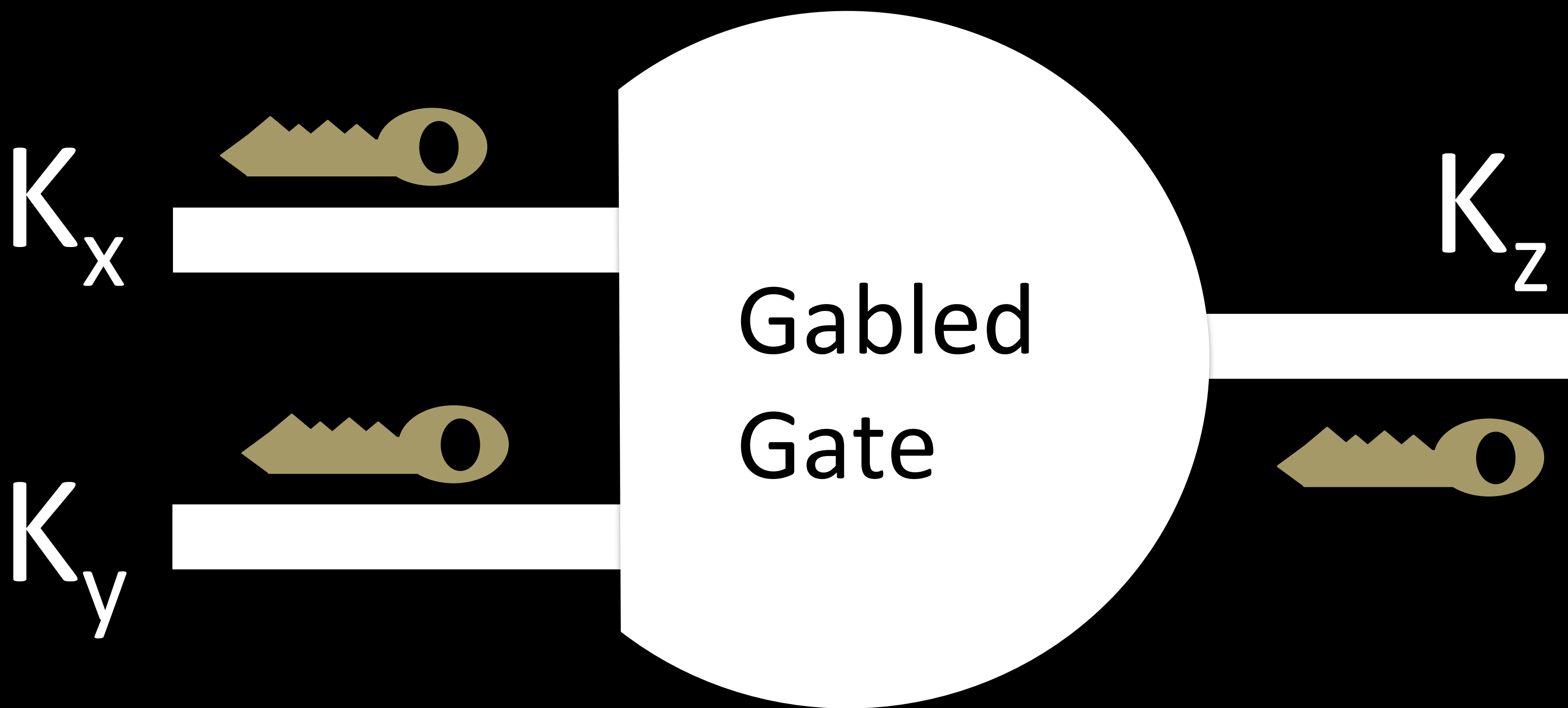


Yao's Garbled Circuits
Gen
Eval

Briefly, Definitions



Briefly, Definitions



K <- G(1^n)

To Eval: $K_z = 0 \approx_c K_z = 1$

Garbled Gates

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

x	y	z
k_x^0	k_y^0	k_z^0
k_x^0	k_y^1	k_z^0
k_x^1	k_y^0	k_z^0
k_x^1	k_y^1	k_z^1

$$g(x, y) : \{0,1\} \times \{0,1\} \rightarrow \{0,1\}$$

$$garbledgate(k_x^\alpha, k_y^\beta) : k_z^\alpha \times k_y^\beta \rightarrow k_z^{g(\alpha,\beta)}$$

Garbled Gates

$$garbledgate(k_x^{\alpha}, k_y^{\beta}) : k_z^{\alpha} \times k_y^{\beta} \rightarrow k_z^{g(\alpha,\beta)}$$

$$c_{x,y} = E_{k_x^{\alpha}}(E_{k_y^{\beta}}(k_z^{g(\alpha,\beta)}))$$

Garbled Gates

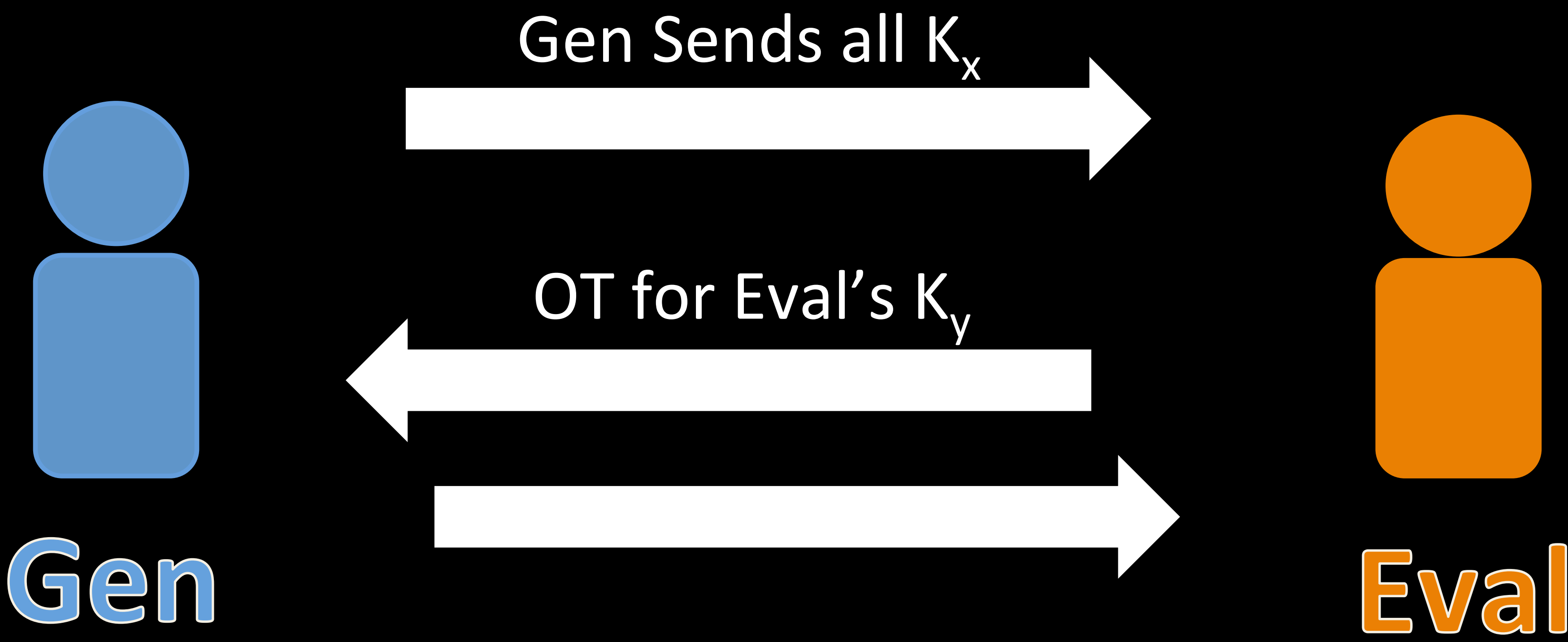
$$c_{0,0} = E_{k_x^0}(E_{k_y^0}(k_z^{g(0,0)}))$$

$$c_{0,1} = E_{k_x^0}(E_{k_y^1}(k_z^{g(0,1)}))$$

$$c_{1,0} = E_{k_x^1}(E_{k_y^0}(k_z^{g(1,0)}))$$

$$c_{1,1} = E_{k_x^1}(E_{k_y^1}(k_z^{g(1,1)}))$$

Inputs



Point and Permute

$$c_{0,0} = E_{k_x^0}(E_{k_y^0}(k_z^{g(0,0)}))$$

$$c_{0,1} = E_{k_x^0}(E_{k_y^1}(k_z^{g(0,1)}))$$

$$c_{1,0} = E_{k_x^1}(E_{k_y^0}(k_z^{g(1,0)}))$$

$$c_{1,1} = E_{k_x^1}(E_{k_y^1}(k_z^{g(1,1)}))$$

Point and Permute

$$c \quad = E_{k_x}(E_{k_y}(k_z^{g(\,?,\,?)}))$$

$$c \quad = E_{k_x}(E_{k_y}(k_z^{g(\,?,\,?)}))$$

$$c \quad = E_{k_x}(E_{k_y}(k_z^{g(\,?,\,?)}))$$

$$c \quad = E_{k_x}(E_{k_y}(k_z^{g(\,?,\,?)}))$$

Point and Permute

p is a *random* permutation bit

b_i is the semantic value of wire *i*

$p_i = \pi_i$ XOR b_i

$w_i = K_z \,||\, p_i$

$C = E_{k1}(E_{k2}(w_i))$

(p1,p2) from input wires identify which table entry Eval should decrypt *and* how Gen should permute

Free XOR

For all *i* in {x,y,z}:

$K_i^1 = K_i^0$ XOR *R*

x	y	z
k_x^0	k_y^0	$k_z^0 = k_0^0 \oplus k_y^0$
k_x^0	$k_y^0 \oplus R$	$k_z^1 = k_x^0 \oplus k_y^0 \oplus R$
$k_x^0 \oplus R$	k_y^0	$k_z^1 = k_x^0 \oplus k_y^0 \oplus R$
$k_x^0 \oplus R$	$k_y^0 \oplus R$	$k_z^0 = k_x^0 \oplus k_y^0$

Garbled Row Reduction 3

Let s be a unique identifier:

$s = Gid \,||\, p_0 \,||\, p_1$

Let ciphertexts be

$C = H(k_x \,||\, s) \text{ XOR } H(k_y \,||\, s) \text{ XOR } K_z$

For the first gate ($p_0 = p_1 = 0$),
define $C = K_z =$

$H(k_x \,||\, s) \text{ XOR } H(k_y \,||\, s)$

Garbled Row Reduction 2

- Consider each key to be a point in GF(2^n)
- Idea:
  - Gen constructs polynomials using input, output keys
    - Gen sets output keys to be points on a curve
  - Gen sends info (2 ciphertexts) about the polynomial to Eval
  - Eval uses her input keys and the ciphertexts to interpolate the polynomial

Garbled Row Reduction 2

Even Gates

Gen computes c1,c2,c3,c4 cipher texts as before

$c = H(k_x \,||\, s) \text{ XOR } H(k_y \,||\, s) \text{ XOR } k_z$

$c_1 = P(1)$

$c_2 = Q(2)$

$c_4 = P(4)$

$c_3 = Q(3)$

$k_z^0 = P(0)$

$k_z^1 = Q(0)$

Gen sends P(5), Q(5) to Eval

Eval interpolates P(0) or Q(0) using one point sent by Gen and one point she derives from input keys

Garbled Row Reduction 2

Odd Gates

Gen computes c1,c2,c3,c4 cipher texts as before

$c = H(k_x \,||\, s) \text{ XOR } H(k_y \,||\, s) \text{ XOR } k_z$

$c_1 = Q(1)$ $c_2 = P(2)$

$c_4 = P(4)$ $c_4 = P(4)$

Gen also calculates $c_5 = Q(5) = P(5)$, $c_6 = Q(6) = P(6)$

$k_z^0 = Q(0)$

$k_z^1 = P(0)$

Gen sends c_5, c_6 to Eval

Eval interpolates P(0) or Q(0) using two points sent by Gen and one point she derives from input keys