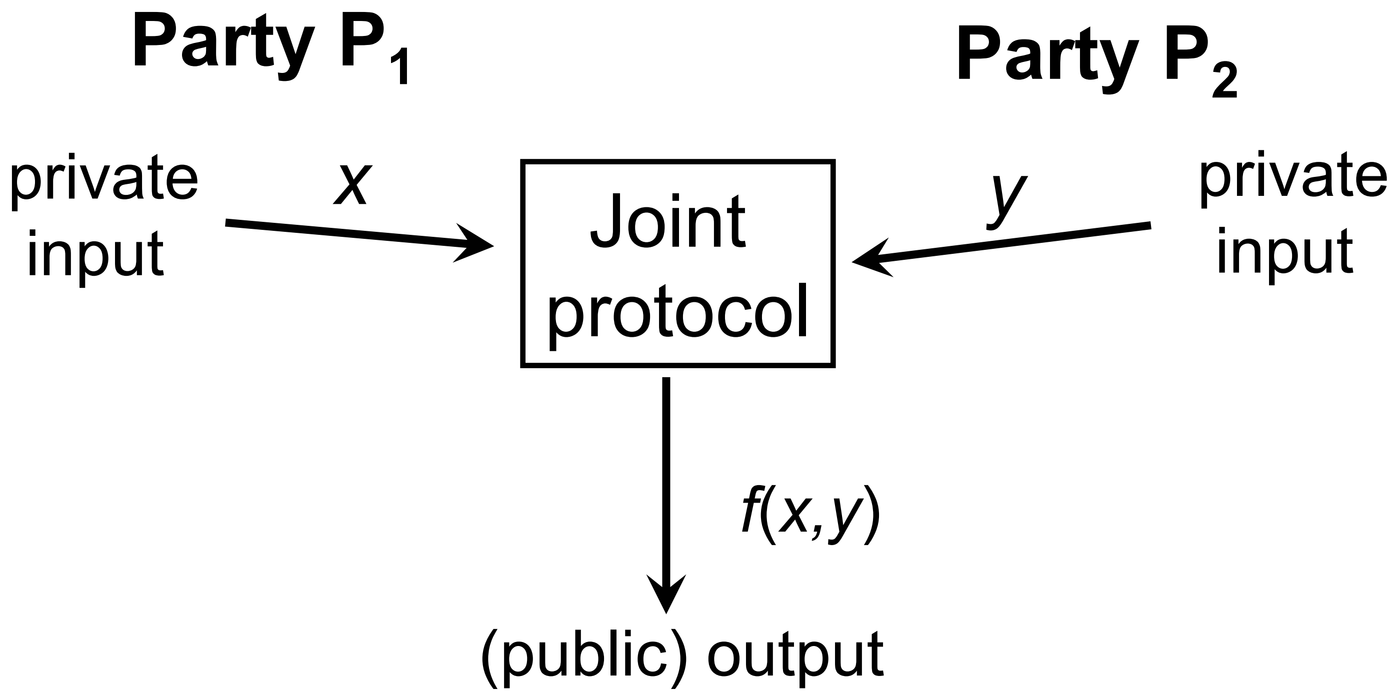
Asiacrypt 2004
December 5-9, 2004, Jeju Island, Korea

Practical Two-Party Computation Based on the Conditional Gate

Berry Schoenmakers TU Eindhoven

Pim Tuyls Philips Labs Eindhoven

Two-Party Computation: Secure Function Evaluation



Secure: $f(x,y)$ is computed *correctly*
Private: inputs x,y remain *secret* to P_2,P_1, resp.
Fair: P_1 and P_2 *both* obtain the output $f(x,y)$

Example: secure profile matching (e.g., musical prefs, or biometric profiles)

User 1

Client

Musical preferences

Pop: yes

Rock: no

Classic: yes

…

“Freshly” measured biometric profile

$x=(1,0,1,\dots)$

x → Joint protocol ← y

$f(x,y)$

User 2

Server

Musical preferences

Pop: yes

Rock: yes

Classic: no

…

Stored biometric profile (template)

$y=(1,1,0,\dots)$

$f(x,y)$ = if distance(x,y)<T then 1 else 0

Outline

- Threshold Homomorphic Cryptosystems
  - main tool: threshold homomorphic ElGamal
- Simple & efficient secure computation
  - Conditional Gate: special multiplication gate
  - Example: Yao’s Millionaires problem
  - Extensions: private outputs, fairness (also under DDH assumption)

Framework: THCs

- **Threshold Homomorphic Cryptosystem (THC)**:
  - Distributed Key Generation (DKG): to share private key
  - Homomorphic Encryption: under single public key
  - Threshold Decryption: joint decryption protocol
- THCs form basic tool for secure multiparty computation, following [FH93,JJ00,CDN01,DN03]
  - we focus on 2-party case (but results extends to multiparty case, incl. case of dishonest majority)
- **Advantage**: low broadcast complexity of $O(|C|\ n\ k)$ bits for circuit of size $|C|$, n parties, security parameter k
- **Issue:** DKG can be relatively **expensive**

Many user scenario

- Large population of users (say 1 million)

- Ad-hoc pairs of users U_i and U_j execute these **two** stages:
  1) They run a **DKG** (Distributed Key Generation) protocol for a (2,2)-threshold homomorphic cryptosystem.
  2) They run a 2-party protocol using the (2,2)-THC.

- Performance: **total time to completion (incl. DKG)**
  - depends on a variety of factors, where the relative influence of each factor depends on the specific platform (computing scenario)
    - computational complexity
    - communication complexity
    - round complexity (latency)

Popular choice of THCs

- **Homomorphic ElGamal**
  - DDH assumption
  - $E_{g,h}(m,r) = (g^r, h^r g^m)$
  - **Pros**:
    - *efficient DKG* to share private key $\alpha = \log_g h$ [Ped91,…,AF04]
    - allows for *elliptic curves (exponential security)*
  - **Cons**:
    - *limited decryption* (only full decryption of $\boldsymbol{g^m}$, from which m needs to be recovered still).
- **Paillier**
  - RSA-like assumption
  - $E_n(m,r) = (1+n)^m r^n \bmod n^2$
  - **Pros**:
    - *full decryption* of message m
  - **Cons**:
    - *expensive DKG* for generating a shared RSA modulus [Gil99,ACS02]. Cost of DKG may **dominate** total cost.
    - only subexponential security

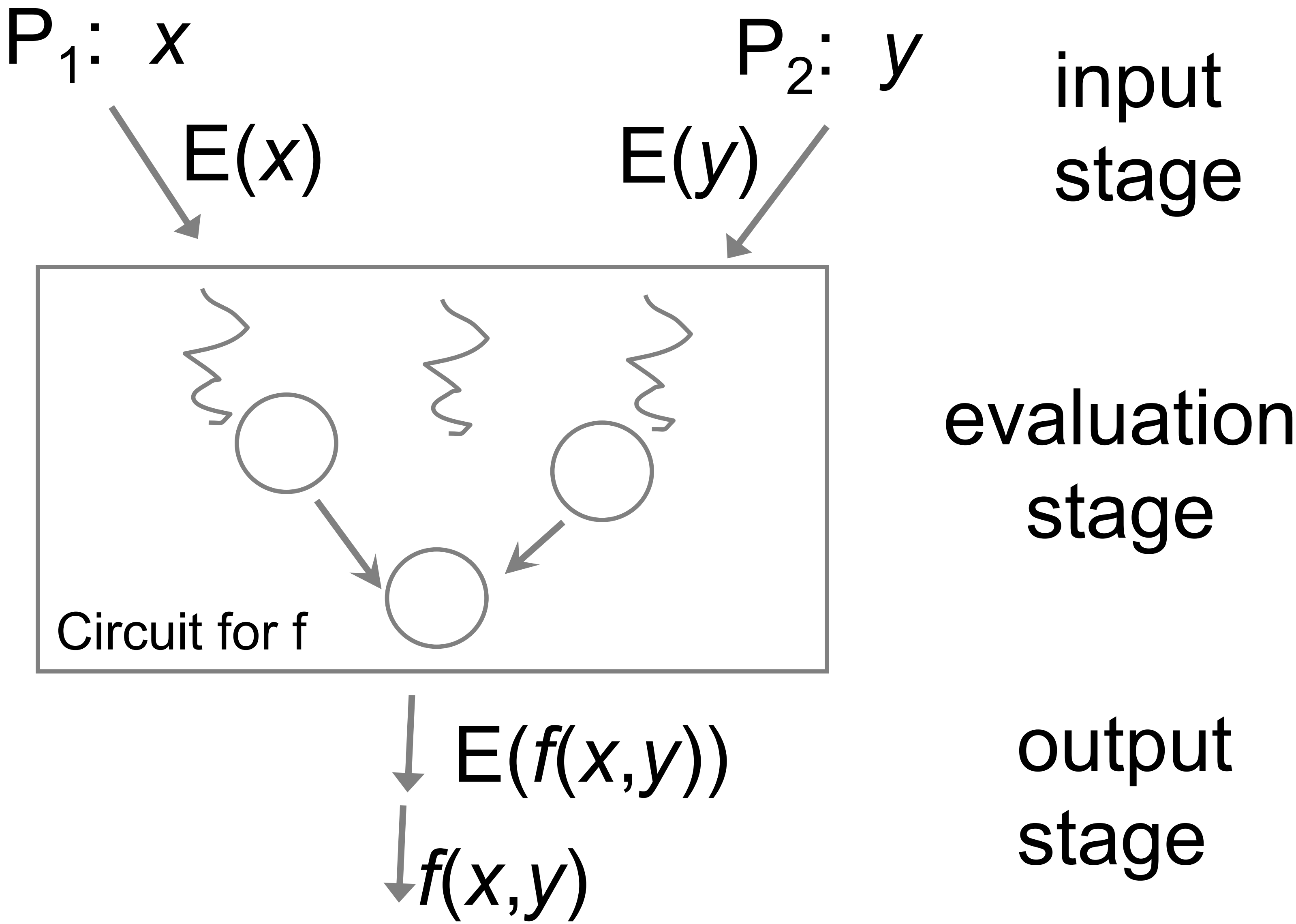
Popular choice of THCs (cont.)

- **ELGamal-Paillier amalgam (CraSho'02, DamJur'03)**
  - DDH and RSA-like assumption
  - $E_{g,h,n}(m,r) = (g^s \bmod n, (1+n)^m (h^s \bmod n)^n \bmod n^2)$
  - **Pros**:
    - *full decryption* of message *m*
    - *expensive DKG* now **only at system setup**
      (single, system-wide RSA modulus *n* for all users)
  - **Cons**:
    - large overhead due to large ciphertexts, e.g. compared to ElGamal combined with elliptic curves
      - even if secure computation is mostly bitwise (Boolean circuits)
    - two assumptions:
      - factorization of RSA modulus *n* is actually a trapdoor (and could get compromised)

Abstract view of (2,2)-THC

- E(m) denotes a *probabilistic* encryption of m for a key pair (pk,sk), where sk is shared in a (2,2)-threshold fashion
- Homomorphic properties:
  - $E(m_1)\,E(m_2) = E(m_1+m_2)$ "additive"
  - $E(m)^c = E(c\,m)$ "scalar multiplication"
  - $E(m)\,E(0) = E(m)$ "re-randomization (blinding)"
- Decryption done by a protocol between two parties
  - for homomorphic ElGamal: m must be from a small range such that m can be recovered from g^m

Secure Function Evaluation



Secure Function Evaluation from THCs

- Franklin, Haber (1993)
  - applies to **Boolean** circuits
  - uses GM-ElGamal variant (factoring-based), **expensive DKG**
  - secure against **passive** adversaries
- Jakobsson, Juels (2000) "Mix and Match"
  - applies to **Boolean** circuits
  - uses ElGamal, **easy DKG**
  - secure against **active, static** adversaries
- Cramer, Damgård, Nielsen (2001)/Damgård, Nielsen (2003)
  - applies to **arithmetic** circuits
  - uses factoring-based cryptosystems (e.g., Paillier), **hard DKG**
  - secure against **active, static/adaptive** adversaries

- Our result
  - applies to "**enhanced Boolean**" circuits or "**restricted arithmetic**" circuits
    - more powerful and more efficient than Mix and Match
  - uses ElGamal, **easy DKG**
  - secure against **active, static** adversaries

Addition Gate

- Input: $E(x)$, $E(y)$
- Output: $E(x+y)$

- For free, because of homomorphic property:

$$E(x)\ E(y) = E(x+y)$$

Also, for given c,

$$E(\mathrm{x})^{c} = E(c\,x)$$

Multiplication Gate

- Input: $E(x)$, $E(y)$
- Output: $E(x\,y)$
- Hard!
- General solution using just homomorphic ElGamal encryption would solve the Diffie-Hellman problem (computing g^{xy} from g^x and g^y), even knowing the private key for E().
- Thus, use **restricted** multiplication gates

(Auxiliary) Private-Multiplier Gate

- Input: $E(x)$, $E(y)$
- Output: $E(x\,y)$

- Suppose **multiplier *x* is private** to a single party P_i, say.
- Multiplicand y is not restricted.

- Easy: P_i computes the x-th power (+Σ proof)

$$E(y)^x = E(x\,y),$$

also including re-randomization.

Conditional Gate

- Input: E (x) , E (y)
- Output: E $(x\,y)$

- Suppose **multiplier *x*** is from a **2-valued** domain, say {-1,1}
  - Enables the use of blinding/deblinding using limited decryption.
- **Multiplicand *y*** can be **any value in Z_q** for large prime q, say $|q|$=160 bits.

Conditional Gate - Protocol

Let $\boldsymbol{x} \in \{\boldsymbol{-1,1}\}$, $y \in Z_q$.

P$_1$

random $s_1 \in \{-1,1\}$

$E(x_1) \leftarrow E(x)^{s_1}$

$E(y_1) \leftarrow E(y)^{s_1}$

(+ Σ proof)

$E(x_1), E(y_1)$ →

P$_2$

random $s_2 \in \{-1,1\}$

$E(x_2) \leftarrow E(x_1)^{s_2}$

$E(y_2) \leftarrow E(y_1)^{s_2}$

(+ Σ proof)

← $E(x_2), E(y_2)$

- threshold-decrypt $E(x_2)$ and check $x_2 \in \{-1,1\}$
- output $E(y_2)^{x_2}$.

Note: $E(y_2)^{x_2} = E(s_1 s_2 x\ s_1 s_2 y) = E(xy)$

since $s_1^2 = s_2^2 = 1 \pmod{q}$

Simple Application

- Conditional gate corresponds to an "if-then-else" control structure.

- **Verifiable MIX of two ciphertexts**:

Let $x \in \{0,1\}$ and $y_1, y_2 \in Z_q$.

$$f(x, y_1, y_2) = \textbf{if } x{=}0 \textbf{ then } (y_1, y_2) \textbf{ else } (y_2, y_1)$$
$$= (y_1 + x\,(y_2 - y_1)\,,\; y_2 - x\,(y_2 - y_1))$$

Requires a single conditional gate only.

Integer comparison *x>y*

- Input: $E(x_{m-1}), \ldots, E(x_0)$
  $E(y_{m-1}), \ldots, E(y_0)$
- Output: if $x > y$ then E(1) else E(0)

- Circuit, or oblivious program (lsb to msb):

$$t_0 = 0$$

$$t_{i+1} = (1-(x_i-y_i)^2)t_i + x_i(1-y_i), \quad i = 0,\ldots,m-1$$

Output: t_m

- Circuit requires 2m conditional gates

Yao's Millionaires Problem

- Same as x>y, but with simplification that ***x* and *y* are private inputs for parties P_1 and P_2**, resp.
- Set t_0 and for $i = 0, \ldots, m$-1:
  - P_2 sets $h_i = y_i t_i$
  - P_1 sets $t_{i+1} = t_i - h_i - x_i(t_i - 2h_i + y_i - 1)$
- Only private-multipliers are used!

- Computational complexity:
  - only about 12*m* modular exponentiations (incl. proofs)
- Round complexity: O(*m*)
  - can be reduced to O(log *m*)

Some Infeasible Problems

ElGamal encryption: $E(x) = (g^r, h^r g^x)$

- Given $E(x)$, $E(y)$, compute $E(xy)$
- Given $E(x)$, compute $E(x^2)$ (or, $E(1/x)$)
- Given $E(x)$, compute $E(x \bmod 2)$

- For $0 \le x < 2^m$, given $E(x)$, compute $E(x \bmod 2)$
- For $0 \le x < 2^m$, given $E(x)$, compute $E(x < 2^{m-1})$

- A way-out for $0 \le x < 2^m$:
  - work bit-wise using $E(x_{m-1})$,…, $E(x_0)$

Extensions

- Private outputs
  - for two party case:
    - $f(x,y) = (f_1(x,y), f_2(x,y))$

      where $f_1(x,y)$ is private output for P_1

      $f_2(x,y)$ is private output for P_2

- Fairness: make threshold decryption of outputs of the circuit evaluation fair.

Private outputs

- Given encryption E(m), m should be output to a single party P_j, say.
- Common approach:
  - blind E(m) to E($m+r$) where r is chosen by P_j, and decrypt $m+r$. Only P_j gets m.
- Requires:
  - full decryption of E($m+r$)
  - interaction with P_j

Non-interactive private output

- Input: ElGamal ciphertext (a,b) for public key $h = g^{\alpha}$
- Output: private output for party P_j is a^{α}

- Let a^{α_i} denote party P_i's decryption share, where α_i is P_i's share of the private key.
- Idea: modify threshold decryption by having **each party P_i encrypt a^{α_i} under P_j's public key h_j.**
  - Encryption for P_j: $(c_i,d_i) = (g^r, h_j^r\, a^{\alpha_i})$ + proof.
- Party P_j interpolates

$$\Pi_i\, (c_i,d_i)^{\lambda i} = (g^{\Sigma r_i \lambda i}, h_j^{\Sigma r_i \lambda i}\, a^{\alpha})$$

and decrypts to get a^{α}.

Fairness

- 2-party protocol is not robust. If either party stops, the protocol is aborted:
  - during input or evaluation stage: no problem.
  - during threshold-decryption in the output stage: not fair, other party does not learn output
- "*Weak* fairness": achieved by **gradual release of decryption shares**; can be added **modularly** onto the non-fair protocol.
- But under standard DDH assumption.

Conclusion

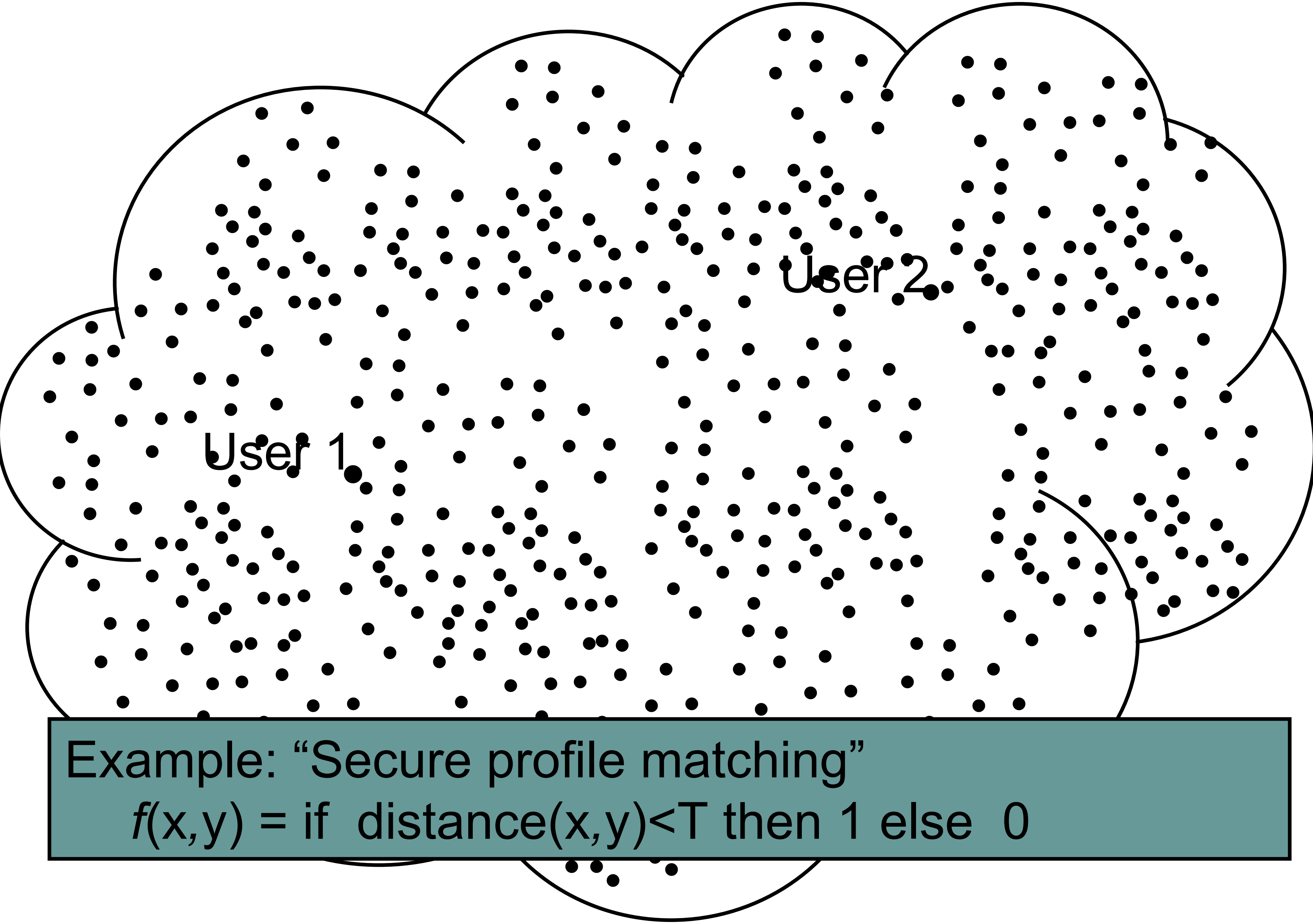
- Simple & Efficient two/multi-party computation using just threshold homomorphic ElGamal.
- Competition between approaches?
  - e.g., Yao's garbled circuits (used by Fairplay):
    - Garbled circuits good at large circuits (or rather, **with relatively many gates**)
      - good if average number of gates per input is large
    - Gate-by-gate THC approach good at small circuits, or rather circuits **with relatively many inputs**.
      - good if average number of gates per input is small
- Precise comparison is open!

Author’s address

Berry Schoenmakers

Coding and Crypto group
Dept. of Math. and CS
Eindhoven University of Technology
P.O. Box 513
5600 MB Eindhoven
Netherlands

berry@win.tue.nl
http://www.win.tue.nl/~berry/



Example: “Secure profile matching”

$f(x,y)$ = if distance$(x,y)<T$ then 1 else 0