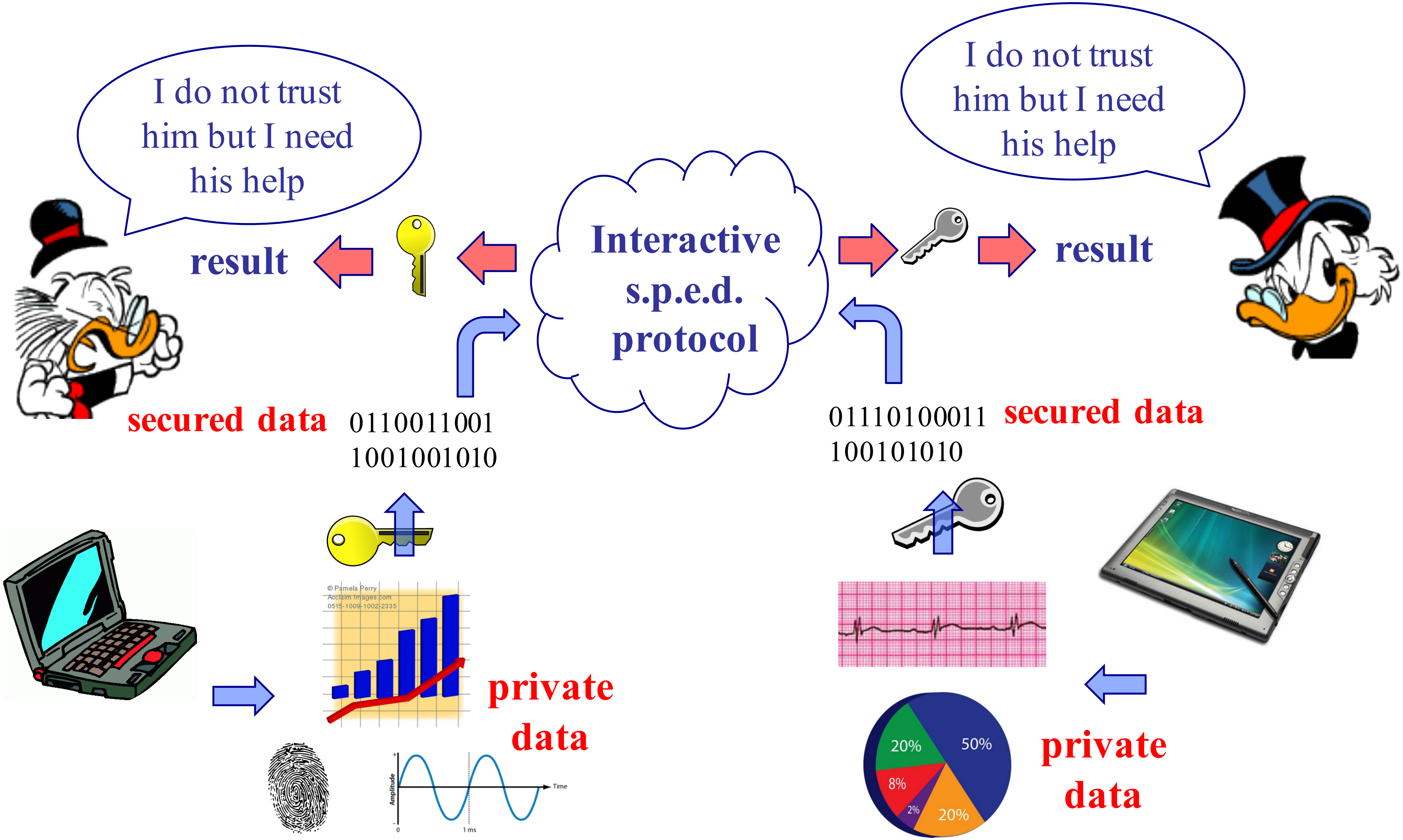
Cybersecurity

Computing with private data

Mauro Barni
University of Siena

What: the s.p.e.d. paradigm



I do not trust him but I need his help

result

Interactive s.p.e.d. protocol

result

I do not trust him but I need his help

secured data 0110011001 1001001010

01110100011 100101010 **secured data**

private data

private data

Why? Network and web security

- Privacy-Preserving Intrusion Detection
  - Analysis of private log files, traffic monitoring
- Abuse detection in social networks
  - Chat rooms or messaging services ensure user anonymity
  - Users should be traceable if they severely violate the terms of usage.
  - To limit traceability to severe instances, abuse detection could be carried out on encrypted data and anonymity revoked only in case of violation
- Oblivious Web Ratings
  - The popularity of web pages is assessed by a third party analyzing the encrypted log files of a web server

Why? Profiling / recommendation services

- Targeted Recommendations
  - Personalized recommendations have high business value but open a privacy-problem
  - Problems can be avoided by methods that analyze the relevant user habits in the encrypted domain (see position information)
- Data Mining for Marketing
  - Knowledge of preferences of class of users is invaluable information in marketing.
  - Performing classifications in the encrypted domain can prevent privacy concerns

Why? Access control and biometrics

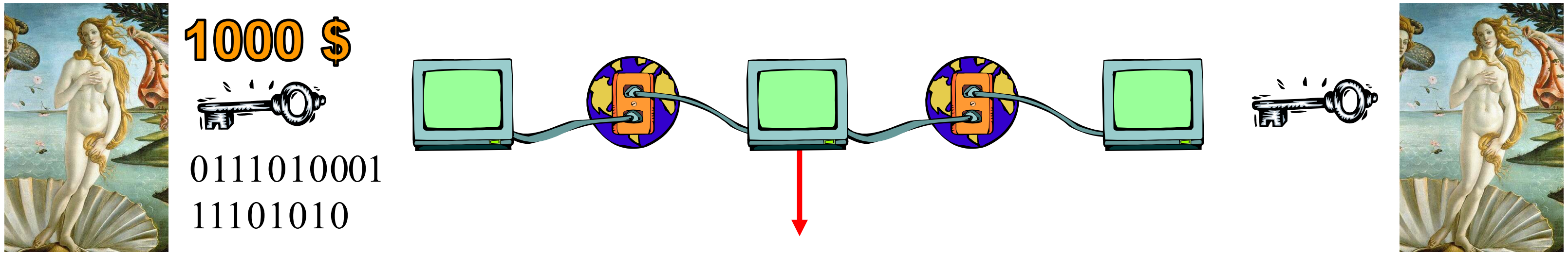
- Private Access control via encrypted queries
  - Access to a service is granted upon inspection of a biometric template (BT)
  - The BT is encrypted so to avoid revealing the biometry and the identity of the user accessing the service
- Biometric control in public places (airport …)
  - An encrypted BT is used to look for criminals or terrorists in public locations
  - Only if a match is found the identity is revealed thus avoiding tracing honest citizens

Why? Biomedical data processing

- Storing biomedical data on remote servers
  - Medical sensitive data/signals are stored under encryption
  - Additional services are provided by processing the encrypted data
  - Cloud services
- Privacy-preserving remote services
  - a remote diagnosis services analyses encrypted data and provides recommendations without violating the users' privacy
- Analysis of bio-signals
  - by processing encrypted bio-signals the analysis reveals only the information it is intended for

Why? Consumer electronics - entertainment

- Privacy preserving search for content
  - again a case of searching with encrypted queries
- DRM
  - the identity of the buyer is embedded in the purchased media without disclosing it to the seller
- Transcoding
  - transcoding of (encrypted) multimedia data at non-trusted nodes



Transcoding without decryption key

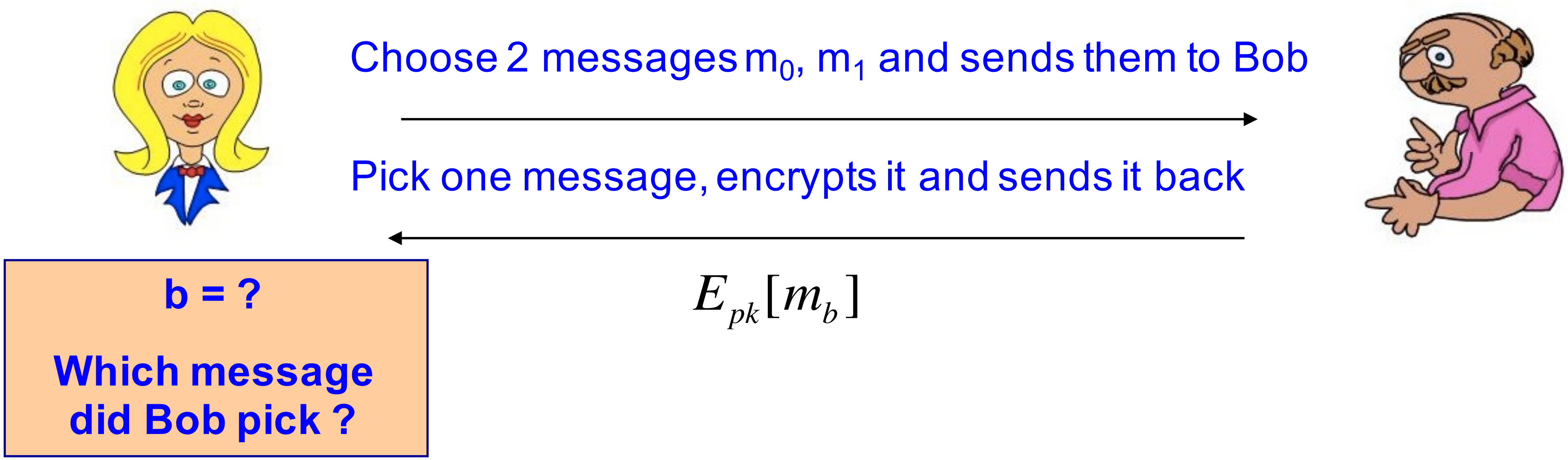
How ? The tools

- Homomorphic encryption
- Blinding / obfuscation
- Oblivious transfer
- Garbled circuits
- Hybrid approach

- Before describing them we need to consider more carefully what do we mean by security in a s.p.e.d. framework

Security model

- What does security mean in s.p.e.d. ? How do we prove security ?
- A huge zoo of security definitions exist
  - what do we want to impede to the attacker ?
  - what is the attacker allowed to know ?
  - what is the (computing) power of the attacker ?
- In s.p.e.d. applications where the message space is small, semantic security (IND-CPA) is needed



Choose 2 messages m_0, m_1 and sends them to Bob

Pick one message, encrypts it and sends it back

$$E_{pk}[m_b]$$

b = ?

Which message did Bob pick ?

Probabilistic encryption

- Randomness of the encryption is needed for semantic security (assume we want to componentwise encrypt a sequence of bits … some sort of randomness is needed)

- In a probabilistic encryption scheme the encrypted message depends on a secret key and a random parameter *r* …

$$c_1 = E_{pk}[x, r_1]$$

$$c_2 = E_{pk}[x, r_2]$$

- … however decryption does depend on *r*

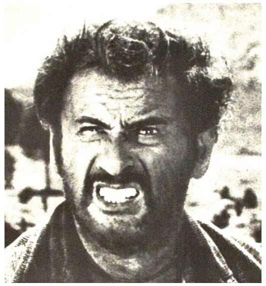
$$x = D_{sk}[c_1]$$

$$x = D_{sk}[c_2]$$

Security model

- In a s.p.e.d. setting further details must be specified: will the adversary follow the protocol or not ?

Semi-honest (honest but curious) adversary: he follows the protocol but tries to infer secret information



Malicious (active) adversary: any action is allowed even departing from the protocol

Covert adversary: he is willing to deviate from the protocol but does not want to be caught

s.p.e.d. tools

- **Homomorphic encryption**
- **Blinding / obfuscation**
- Oblivious transfer
- Garbled circuits
- Hybrid approach

The *homomorphic* route to s.p.e.d.

An algebraic operation on the plain messages is mapped into a (possibly different) algebraic operation on the encrypted messages

$$a \bullet b = D_{sk}[E_{pk}(a) \circ E_{pk}(b)]$$

$$if \begin{cases} \bullet = + \\ \circ = \times \end{cases} \Rightarrow a + b = D_{sk}[E_{pk}(a) \times E_{pk}(b)] \quad additive\ HE$$

⬇

$$Ka = D_{sk}[\underbrace{E_{pk}(a) \times E_{pk}(a) \ldots E_{pk}(a)}_{K\ times}] = D_{sk}[E_{pk}(a)^K]$$

$$if \begin{cases} \bullet = \times \\ \circ = \times \end{cases} \Rightarrow a \times b = D_{sk}[E_{pk}(a) \times E_{pk}(b)] \quad multiplicative\ HE$$

RSA homomorphism

In RSA we have

$c_1 = m_1^e \bmod n$

$c_2 = m_2^e \bmod n$

$c_{12} = c_1 c_2 = m_1^e m_2^e \bmod n = (m_1 m_2)^e \bmod n$

$D[c_{12}] = (m_1 m_2)^{ed} \bmod n = m_1 m_2 \bmod n$

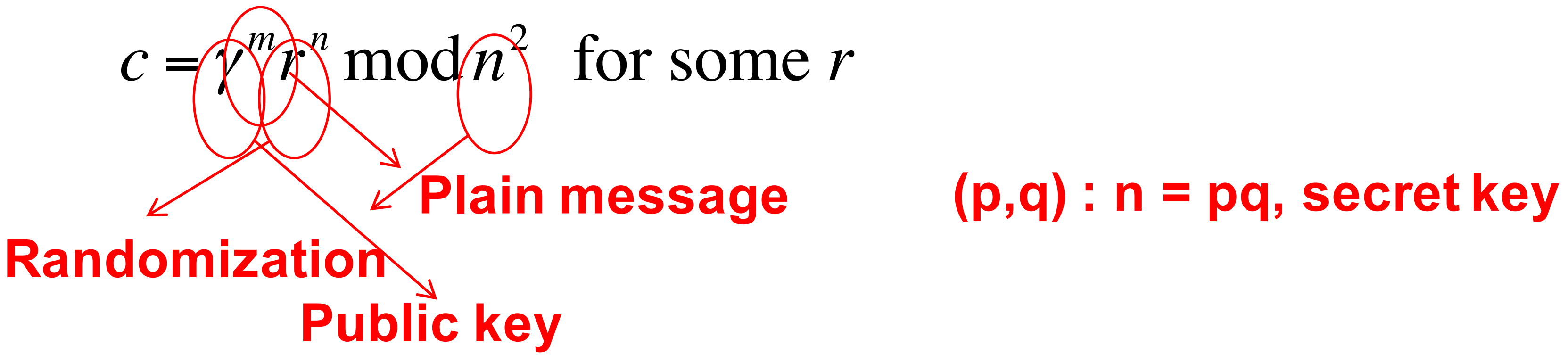
Multiplicative homomorphism

Possible problems with malleability

Pailler's cryptosystem

Composite residuosity problem

Given c, γ and n find m such that



$$c = \gamma^m r^n \bmod n^2 \quad \text{for some } r$$

Plain message

(p,q) : n = pq, secret key

Randomization

Public key

Additive Homomorphism follows from properties of exponentials

Security -> *c* at least 2048 bits

The *homomorphic* route to s.p.e.d.

With additive HE a number of interesting operators can be applied to signals:

$$\text{Component-wise encryption} \Rightarrow E[(a_1, a_2 \ldots\ a_n)] = (E[a_1], E[a_1] \ldots E[a_n])$$

$$\text{Scalar product (known vector } \mathbf{b}\text{): } \langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} a_i\, b_i \Rightarrow E[\langle \mathbf{a}, \mathbf{b} \rangle] = \prod_{i=1}^{n} E[a_i]^{b_i}$$

$$\text{FIR filtering}: \ a_n = \sum_{k=1}^{L} a_{n-k}\, h_k \Rightarrow E[a_n] = \prod_{k=1}^{L} E[a_{n-k}]^{h_k}$$

$$\text{Linear transforms}: \ X_k = \sum_{i=1}^{n} a_{k,i}\, x_i \Rightarrow E[X_k] = \prod_{i=1}^{L} E[x_i]^{a_{k,i}}$$

Non-linear functions and full HE

$$if\ \otimes and \oplus \exists : \begin{cases} a+b = D[E(a) \oplus E(b)] \\ a \times b = D[E(a) \otimes E(b)] \end{cases} \quad full\ HE$$

Kind of holy Graal in cryptography
recent breakthrough by Gentry

…

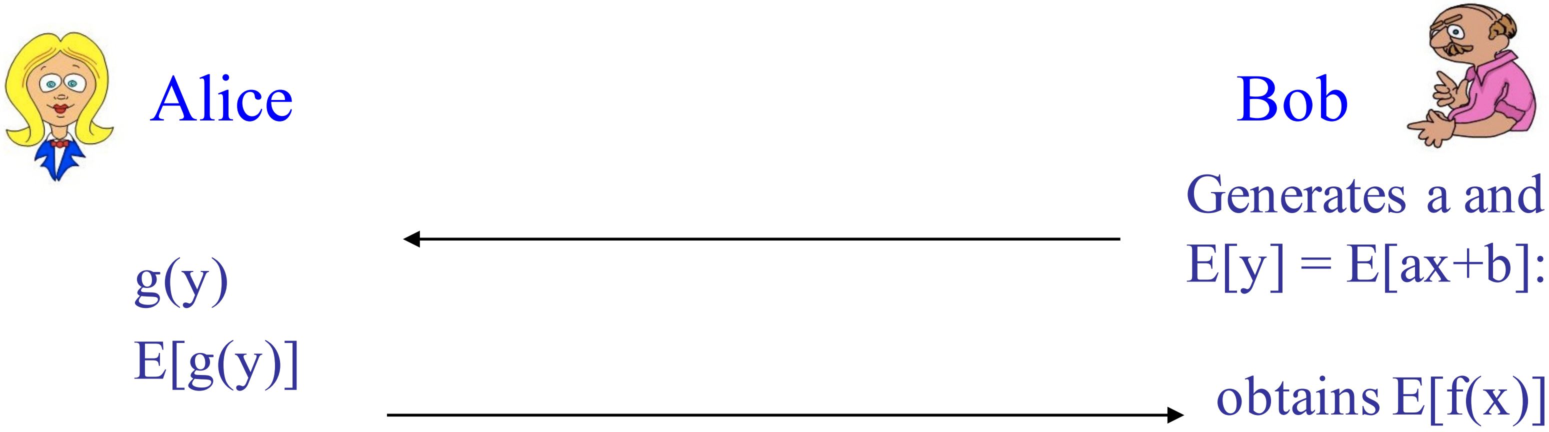
still impractical

…

For the moment s.p.e.d. designers can rely on additive HE only

Non-linear functions: HE + blinding

- Assume an additive cryptosystem is available
- Bob needs to apply a non-linear function *f* () to *x* available to him in encrypted format



Generates a and b randomly
E[y] = E[ax+b]: blinding

g(y)
E[g(y)]

obtains E[f(x)] from E[g(y)]

- Works if $f(x) = \alpha(a,b)g(y) + \beta(a,b)x + \gamma(a,b)$
- … and is difficult (impossible) to recover *x* from *y*

Example: squaring an encrypted number

Alice

Bob

$$E[x]$$

$$E[y] = E[x+b] = E[x]E[b]$$

$$\xleftarrow{\quad E[y] \quad}$$

$$y = D[E[y]]$$

$$g(y) = y^2 = x^2 + b^2 + 2xb$$

$$E[g(y)]$$

$$\xrightarrow{\quad E[g(y)] \quad}$$

$$E[x^2] = E[g(y) - b^2 - 2bx]$$

$$= E[g(y)]E[-b^2]E[x]^{-2b}$$

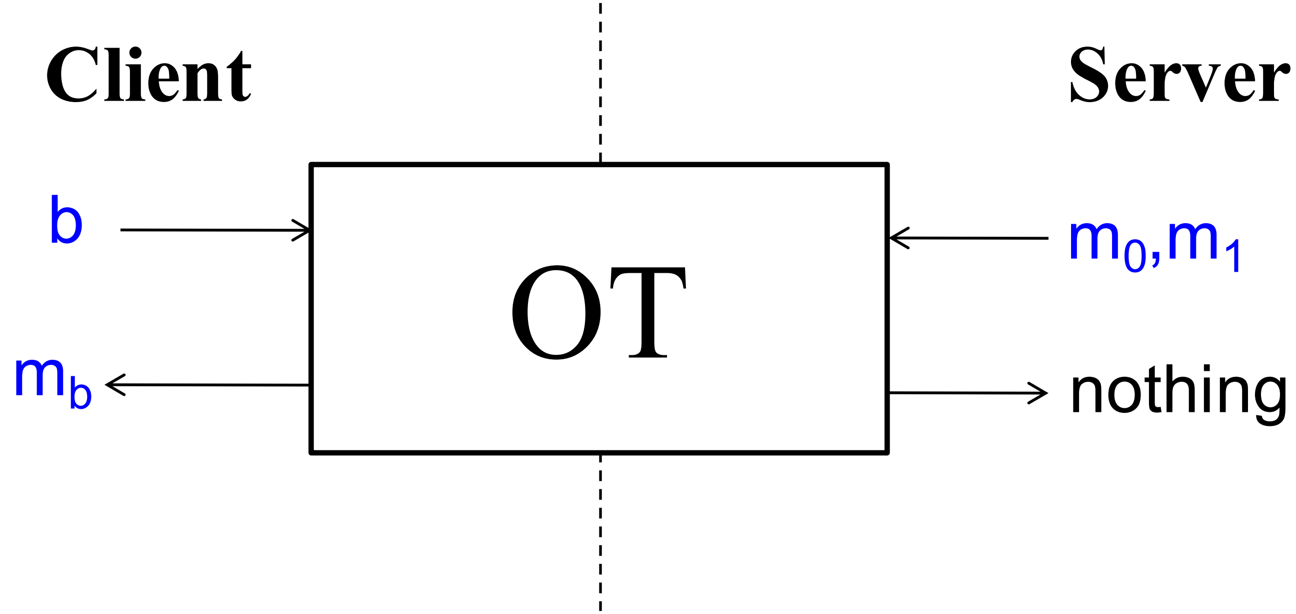
s.p.e.d. tools

- Homomorphic encryption
- Blinding / obfuscation
- **Oblivious transfer**
- **Garbled circuits**
- Hybrid approach

An alternative approach: Garbled Circuits (GC)

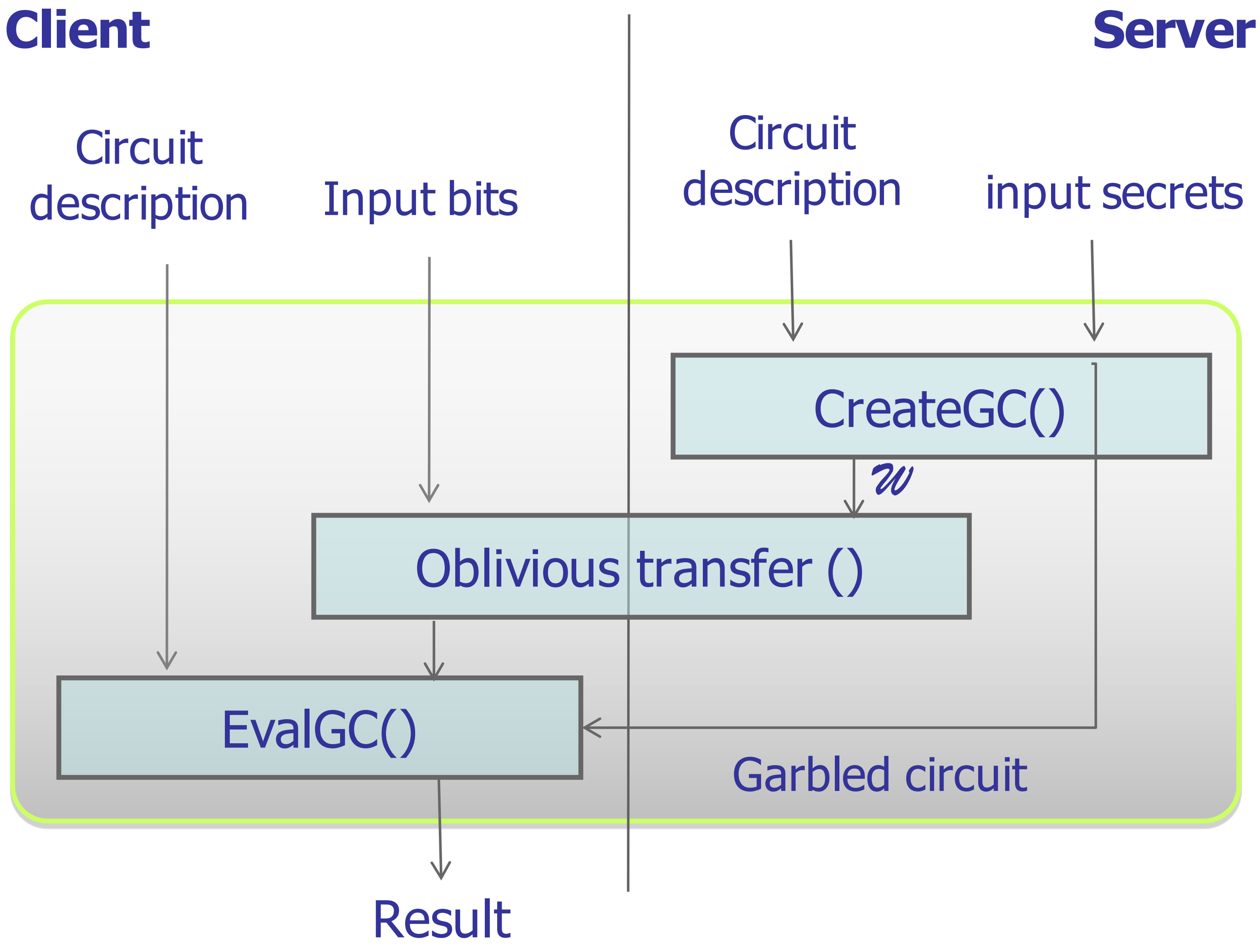
- Private computation of any function expressed as a Boolean (non recursive) circuit
- Symmetric cryptography
- Inputs at the bit level
- Thought to be impractical until 4-5 years ago
  - now: > 100.000 gates per second

Oblivious transfer (OT)



- 1-out-n, parallel version
- Base for a large number of s.p.e.d. protocols
- No details for sake of brevity

General structure of a GC protocol



Circuit description

- A file containing
  - the list of gates together with their truth tables and input ond output wires
  - the list of wires connecting the gates
  - the list of variables associated to the wires split into:
    - Server’s input bits
    - Client’s input bits
    - Internal variables
    - Server’s output bits
    - Client’s output bits

Creation of the GC

- Choose a random t-bit value R
- For each input wire *i* generate 2 t-bit secrets associated to bit 0 and 1 respectively

$$w_i^0 \; , w_i^1 = w_i^0 \oplus R$$

- For each input wire *i* generate a random permutation bit associated to bit 0 and 1

$$\pi_i^0 \; , \pi_i^1 = \pi_i^0 \oplus 1$$

- Note that the above secrets do not reveal any information about the actual input bits

Creation of the GC

- Given a gate and given the secrets associates to the input wires (say i, and j) the secrets associated to the output wire (say k) are created

$$w_k^0 \; , w_k^1 = w_k^0 \oplus R$$

$$\pi_k^0 , \pi_k^1 = \pi_k^0 \oplus 1$$

- For each gate a garble table is constructed as follows (exemplified for an AND gate)

- Table rows are rearranged according to the input permutation bits

Inputs			Garbled table
0,0	w_i^0 , w_j^0	π_i^0 , π_j^0	$(w_k^0 \parallel \pi_k^0) \oplus H(w_i^0 \parallel w_j^0)$
0,1	w_i^0 , w_j^1	π_i^0 , π_j^1	$(w_k^0 \parallel \pi_k^0) \oplus H(w_i^0 \parallel w_j^1)$
1,0	w_i^1 , w_j^0	π_i^1 , π_j^0	$(w_k^0 \parallel \pi_k^0) \oplus H(w_i^1 \parallel w_j^0)$
1,1	w_i^1 , w_j^1	π_i^1 , π_j^1	$(w_k^1 \parallel \pi_k^1) \oplus H(w_i^1 \parallel w_j^1)$

Creation of the GC

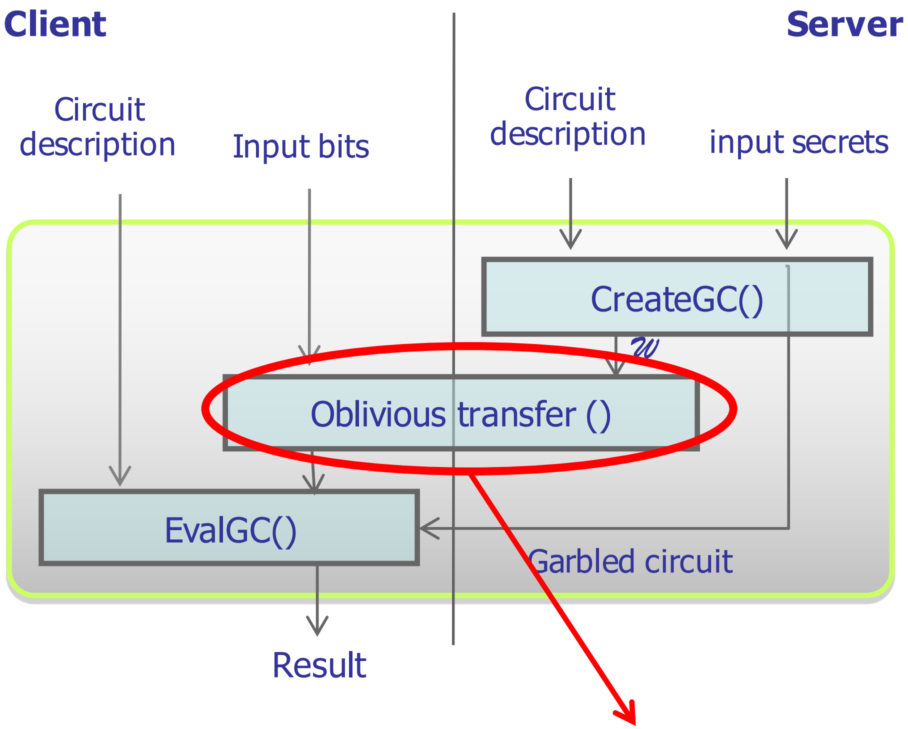
- Garbled tables are built gate by gate as soon as the corresponding secrets are generated
- For the output wires belonging to the client a simplified conversion table consisting of 2 rows only is built

$$0 \oplus H(w_k^0)$$

$$1 \oplus H(w_k^1)$$

- A simplified construction for XOR and NOT gates exists, we skip it for sake of brevity

Data exchange phase



Client

Server

Circuit description

Input bits

Circuit description

input secrets

CreateGC()

w

Oblivious transfer ()

EvalGC()

Garbled circuit

Result

- During the data exchange phase the serves passes to the clients the data necessary to evaluate the GC
- The passed data includes: the garbled tables, the secrets relative to the Server’s input, and the secrets relative to the Client’s inputs

- Secrets associated to client’s inputs are passed by means of OT: the clients inputs his bits and receives the corresponding secrets (and nothing else), the server obtains nothing
- OT is heavy -> better that the client has less inputs

Circuit evaluation

- Suppose that the client knows the input secrets of a certain gate (surely true for input gates)
- He can compute the output secrets as follows
- Select the row indexed by (π_i, π_j)
- Compute the output secret as (assume the input is (0,0):

$$(w_k^0 \| \pi_k^0) = [(w_k^0 \| \pi_k^0) \oplus H(w_i^0 \| w_j^0)] \oplus H(w_i^0 \| w_j^0)$$

Inputs			Garbled table
0,0	w_i^0, w_j^0	π_i^0, π_j^0	$(w_k^0 \| \pi_k^0) \oplus H(w_i^0 \| w_j^0)$
0,1	w_i^0, w_j^1	π_i^0, π_j^1	$(w_k^0 \| \pi_k^0) \oplus H(w_i^0 \| w_j^1)$
1,0	w_i^1, w_j^0	π_i^1, π_j^0	$(w_k^0 \| \pi_k^0) \oplus H(w_i^1 \| w_j^0)$
1,1	w_i^1, w_j^1	π_i^1, π_j^1	$(w_k^1 \| \pi_k^1) \oplus H(w_i^1 \| w_j^1)$

Known since inputs are known

- Iterating until the output gates, the client obtains the output secrets

Circuit evaluation

- If the output belongs to the server, the client sends the corresponding secret, the server retrieves the bits since he knows the secrets
- If the output belongs to the client, he retrieves it by using the output conversion table
- The correct row is selected thanks to the permutation bit, then

$$0 \oplus H(w_k^0) \longrightarrow [0 \oplus H(w_k^0)] \oplus H(w_k^0) = 0 \qquad \text{for a 0}$$

$$1 \oplus H(w_k^1) \longrightarrow [1 \oplus H(w_k^1)] \oplus H(w_k^1) = 1 \qquad \text{for a 1}$$

1. **Loops are not allowed since the GC must be evaluated sequentially**
2. **The GC can be evaluated only once, for a second evaluation a new GC must be built**
3. **Precomputation may help reducing the on-line computation time**

HE vs GC

- HE:
  - pros: no interaction for linear operations, no need of bit-wise representation
  - cons: difficulty with non-linear operations, asymmetric encryption, expansion factor
- GC:
  - pros: universal computing, symmetric crypto
  - cons: bit-wise representation, size of logic circuit may grow more than linearly

Security: most protocols secure against semi-honest adversaries

HE without interaction: secure against any adversary

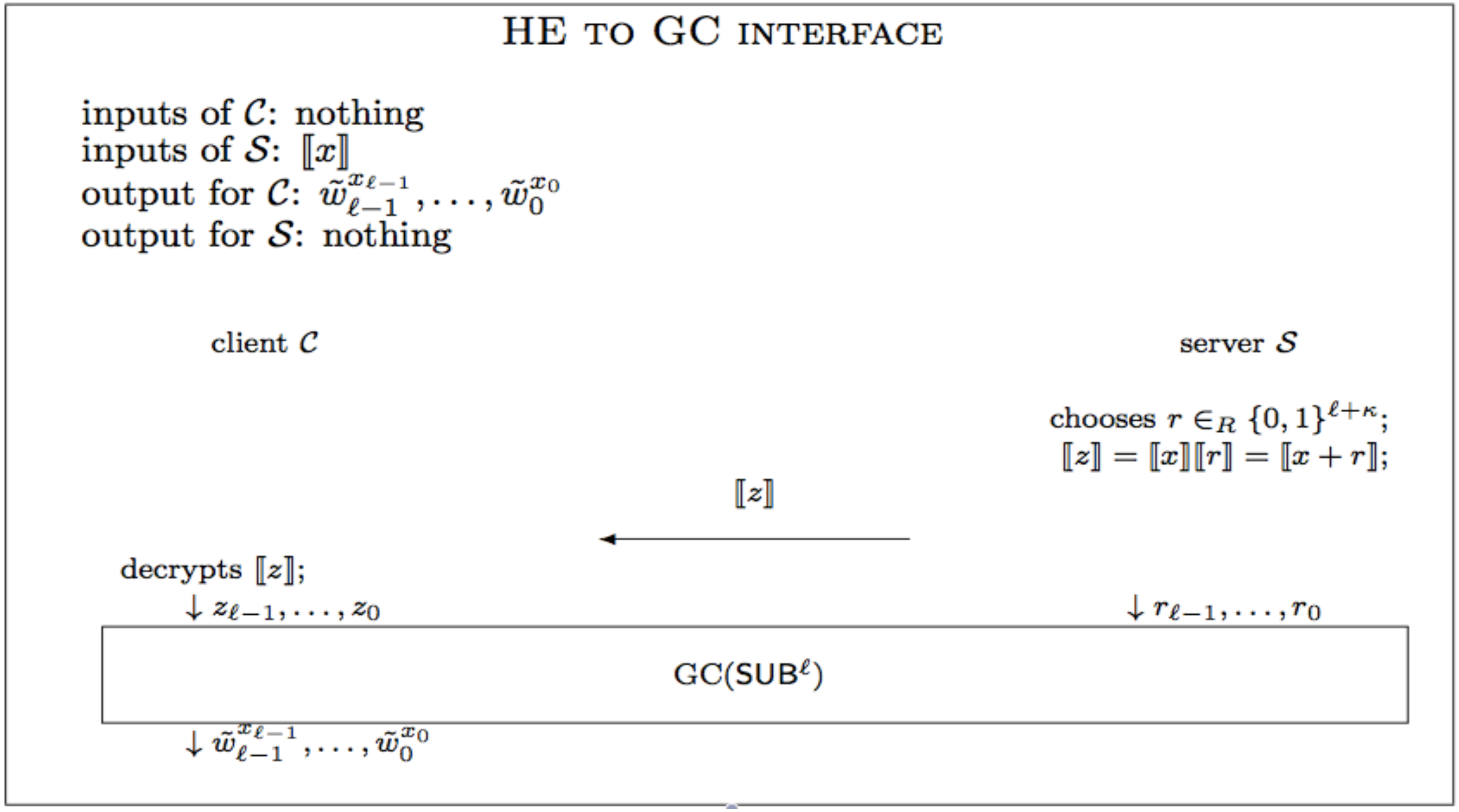
GC secure against malicious client

Hybrid solution

- Recent trend:
  - combine GC and HE to take the best of the two worlds
  - transcoding overhead
  - two protocols are needed
    - to pass from GC to HE
    - to pass from HE to GC

HE -> GC

We assume that a value *x* is available under encryption and we want to generate the bit-wise secrets to go on with a GC computation



HE TO GC INTERFACE

inputs of $\mathcal{C}$: nothing
inputs of $\mathcal{S}$: $[\![x]\!]$
output for $\mathcal{C}$: $\tilde{w}_{\ell-1}^{x_{\ell-1}}, \ldots, \tilde{w}_0^{x_0}$
output for $\mathcal{S}$: nothing

client $\mathcal{C}$

server $\mathcal{S}$

chooses $r \in_R \{0,1\}^{\ell+\kappa}$;
$[\![z]\!] = [\![x]\!][\![r]\!] = [\![x+r]\!]$;

$\xleftarrow{[\![z]\!]}$

decrypts $[\![z]\!]$;

$\downarrow z_{\ell-1}, \ldots, z_0$

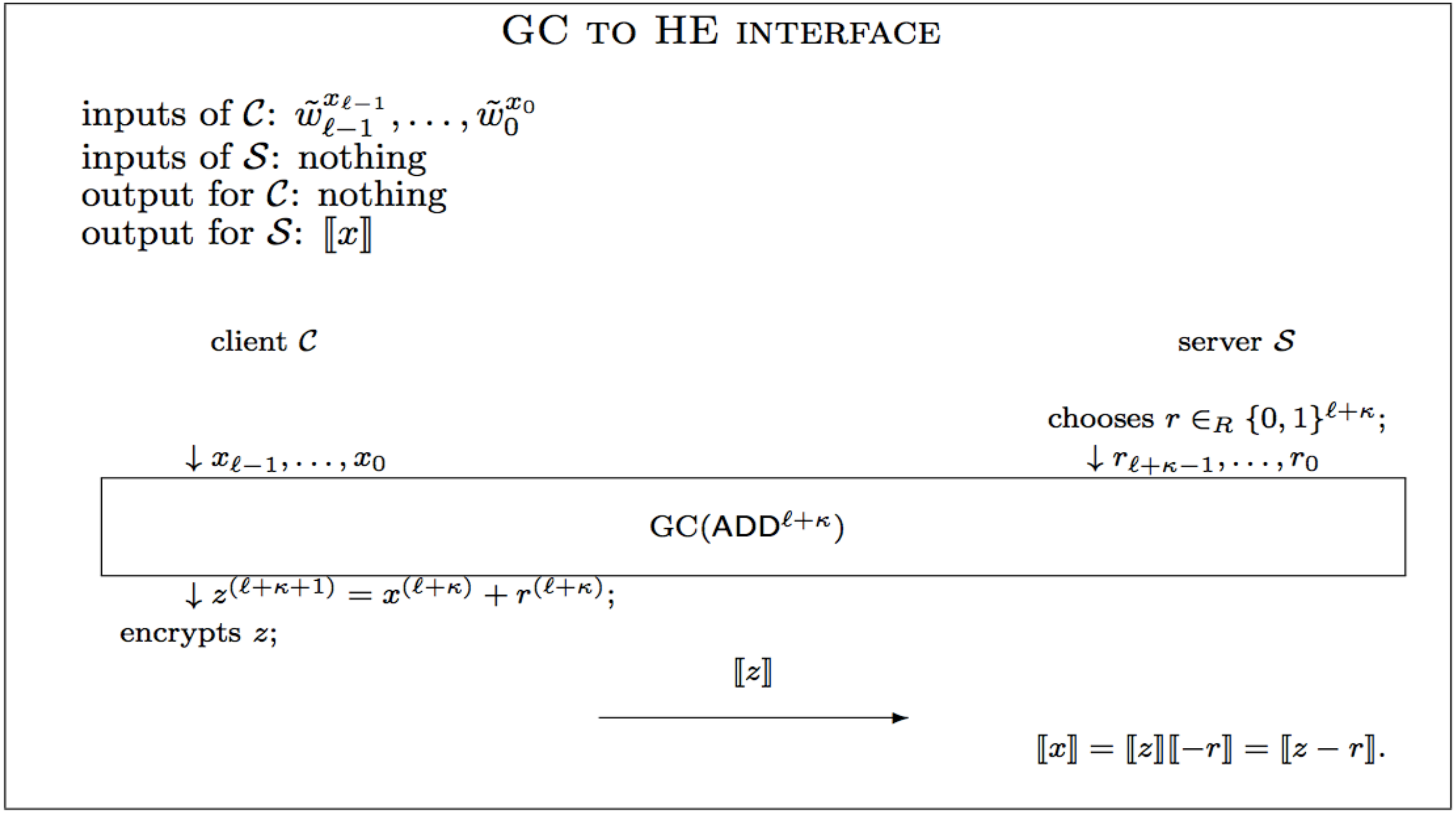
$\downarrow r_{\ell-1}, \ldots, r_0$

GC(SUB$^\ell$)

$\downarrow \tilde{w}_{\ell-1}^{x_{\ell-1}}, \ldots, \tilde{w}_0^{x_0}$

GC -> HE

We assume that the secrets relative to the bits of *x* are available. We want to obtain an encrypted version of *x*



GC TO HE INTERFACE

inputs of $\mathcal{C}$: $\tilde{w}_{\ell-1}^{x_{\ell-1}}, \ldots, \tilde{w}_0^{x_0}$
inputs of $\mathcal{S}$: nothing
output for $\mathcal{C}$: nothing
output for $\mathcal{S}$: $[\![x]\!]$

client $\mathcal{C}$

server $\mathcal{S}$

chooses $r \in_R \{0,1\}^{\ell+\kappa}$;

$\downarrow x_{\ell-1}, \ldots, x_0$

$\downarrow r_{\ell+\kappa-1}, \ldots, r_0$

GC($\mathsf{ADD}^{\ell+\kappa}$)

$\downarrow z^{(\ell+\kappa+1)} = x^{(\ell+\kappa)} + r^{(\ell+\kappa)}$;

encrypts z;

$$[\![z]\!] \longrightarrow$$

$$[\![x]\!] = [\![z]\!][\![-r]\!] = [\![z-r]\!].$$

What is the role SP designers ?

- **Optimize algorithms in terms of**
  - – bit length and number of variables
    - All cryptographic primitives work only on integer values -> data quantization necessary
    - Integer representation allowed but no truncation
    - Representation complexity may grow during the computation
    - Possible surprises: DFT more efficient than FFT
  - – Representation accuracy has a strong impact on
    - Accuracy of results
    - Complexity of the protocol
    - Trade-off needed

What is the role SP designers ?

- **Optimize algorithms in terms of**
  - adopted tools in view of available s.p.e.d.primitives
  - Simple operations in the plain domain may be very complex when applied on encrypted signals
    - Comparisons, if-then-else, sorting: very complex operations with HE
    - Multiplications and divisions: very complex with GC

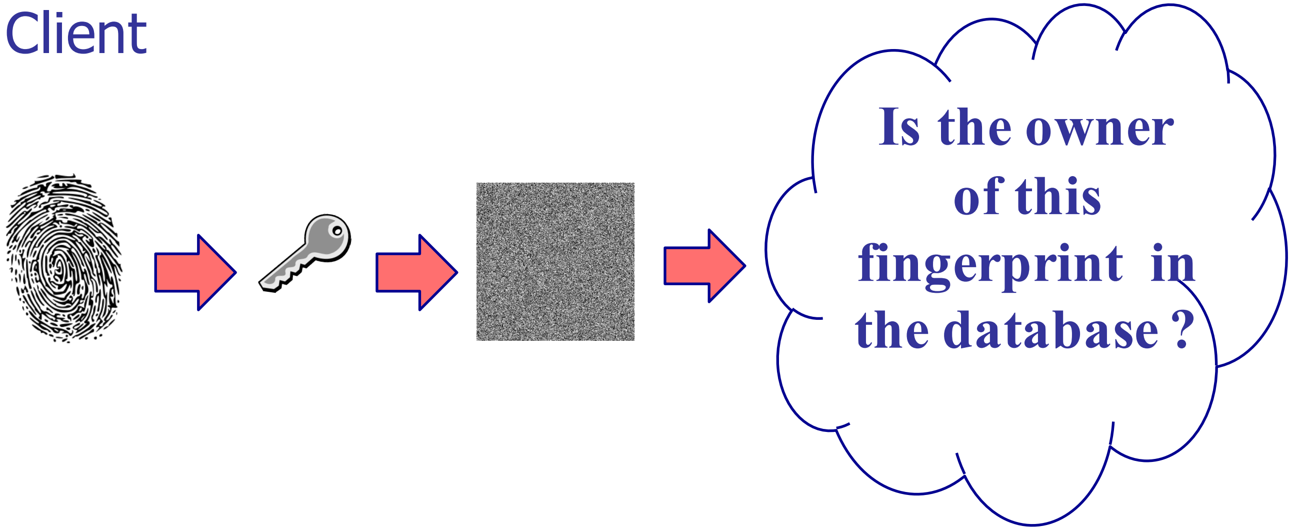
THE – RELATIVE - PRICE TO PAY TO PASS FROM PLAIN DOMAIN COMPUTATION TO SPEED IS ALWAYS QUITE LARGE

s.p.e.d. at work

[1] M. Barni, et al. “A Privacy-compliant Fingerprint Recognition System Based on Homomorphic Encryption and Fingercode Templates”, *Proceedings of BTAS 2010, IEEE Fourth International Conference On Biometrics: Theory, Applications And Systems*, Washington DC, USA, September 27-29, 2010.

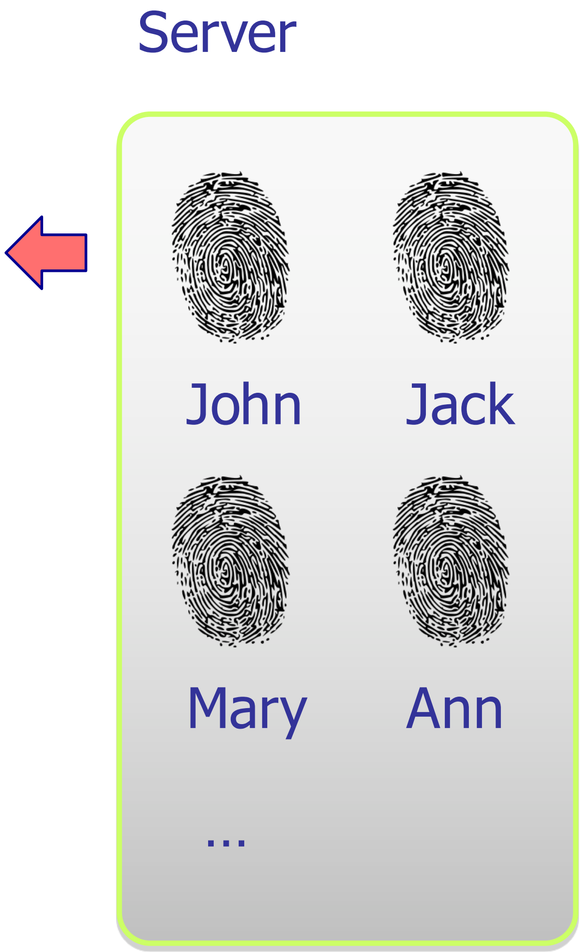
[2] M. Barni, P. Failla, R. Lazzeretti, A-R. Sadeghi, T. Schneider, “Privacy-Preserving ECG Classification with Branching Programs and Neural Networks”, IEEE Trans. on Information Forensics and Security, vol. 6, no. 2, June 2011, pp. 452-468.

Biometric-based authentication



Client

Is the owner of this fingerprint in the database ?



Server

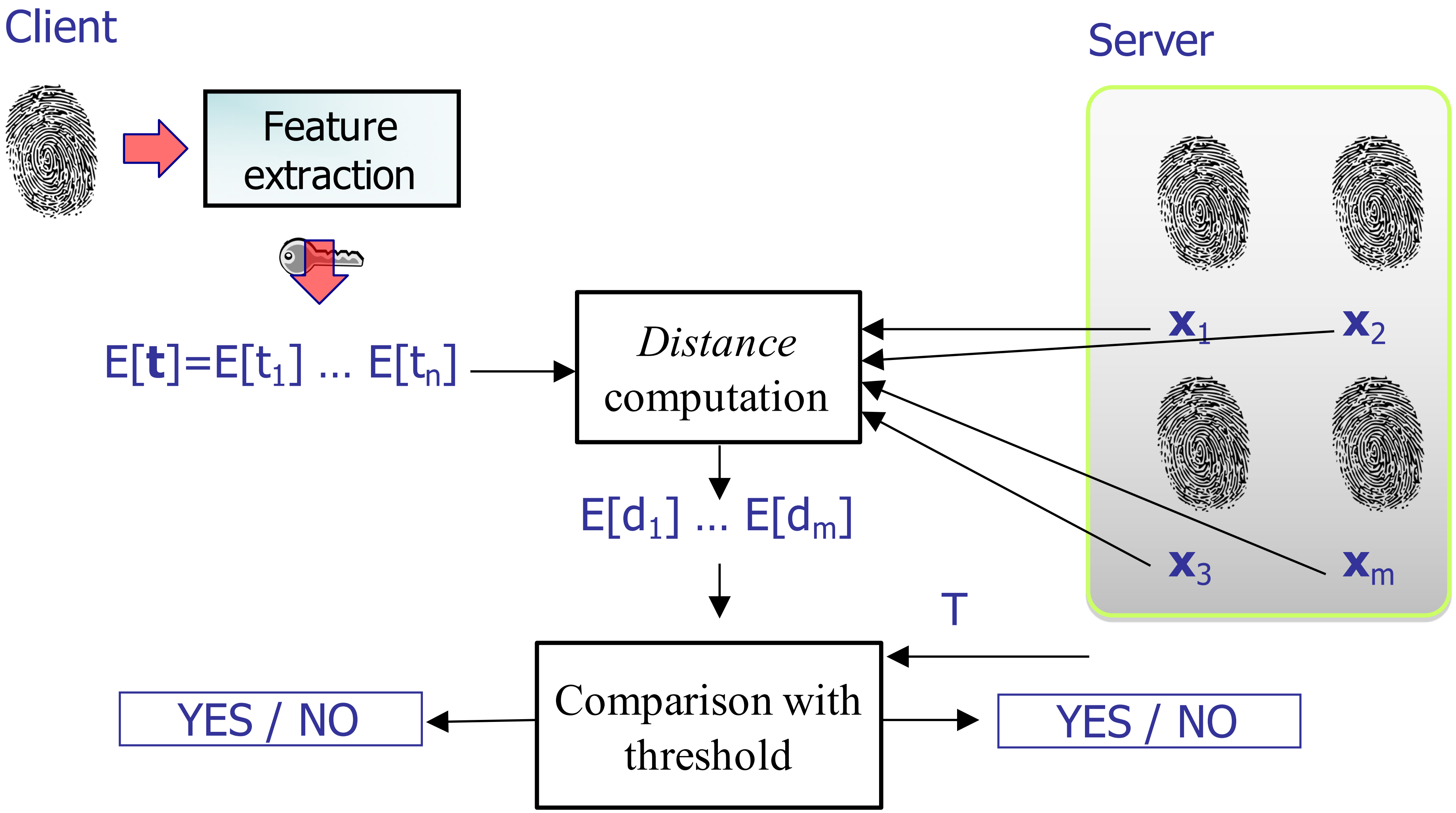
John Jack

Mary Ann

…

- Criminal tracking with privacy protection for citizens: if you are not a criminal the system will not track you
- Privacy preserving access control: I know you can access a service but don't know who you are

Biometric-based authentication



Client

Server

Feature extraction

$E[\mathbf{t}]=E[t_1]\ \dots\ E[t_n]$

Distance computation

$\mathbf{x}_1$

$\mathbf{x}_2$

$\mathbf{x}_3$

$\mathbf{x}_m$

$E[d_1]\ \dots\ E[d_m]$

T

Comparison with threshold

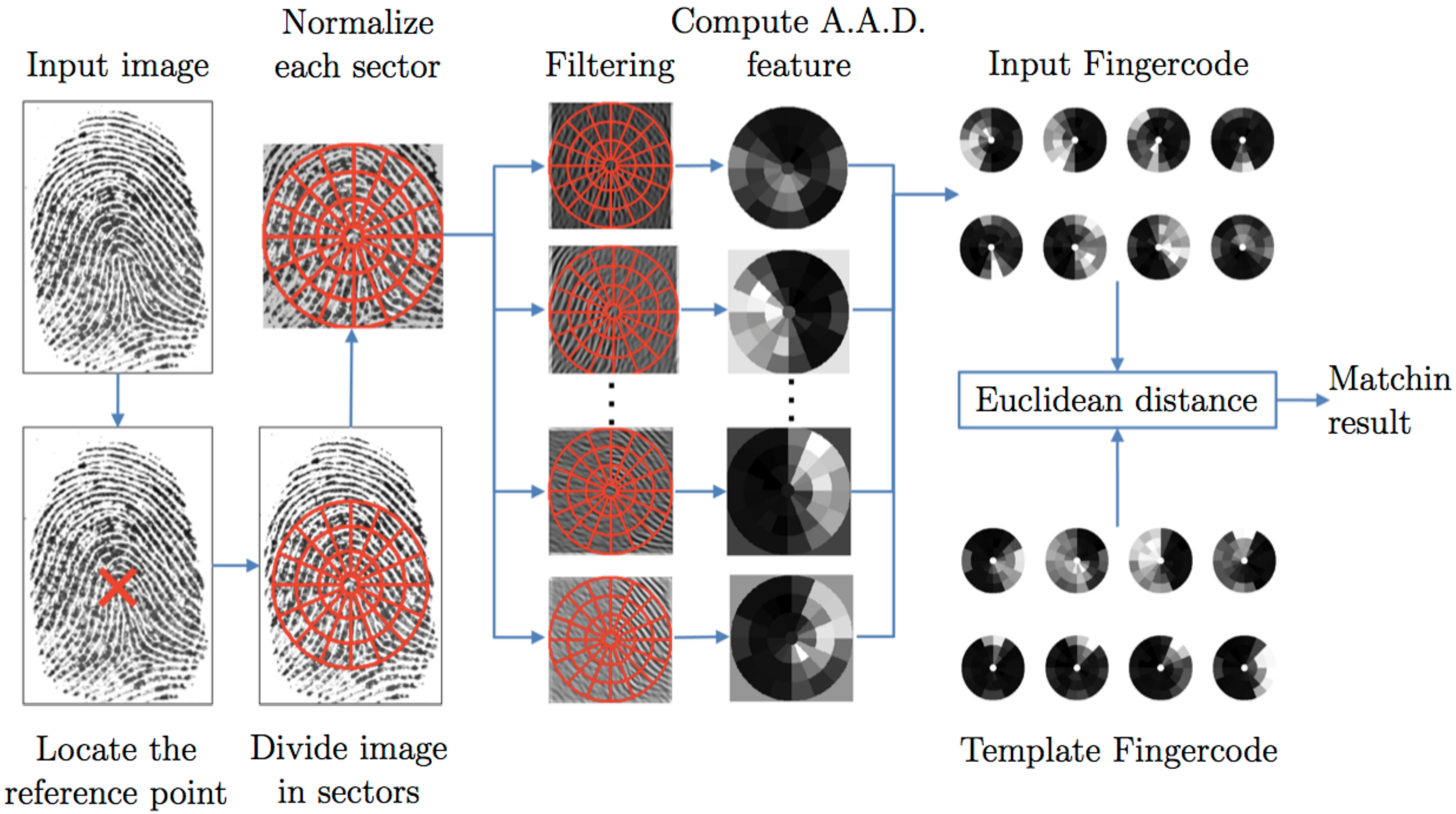
YES / NO

YES / NO

SP choices

- Choice of feature set and distance function that ease a s.p.e.d. implementation
- Classical approaches based on minutiae are difficult to implement
- Our choice:
  - **Fingercode**
    - Energy contained in different areas of the fingerprint image in different frequency bands
    - **Minimize number of features**
    - **Optimize representation accuracy**
  - **Euclidean distance**

Fingercode representaion of fingerprints



Optimization of fingercode representation

Size of feature vector

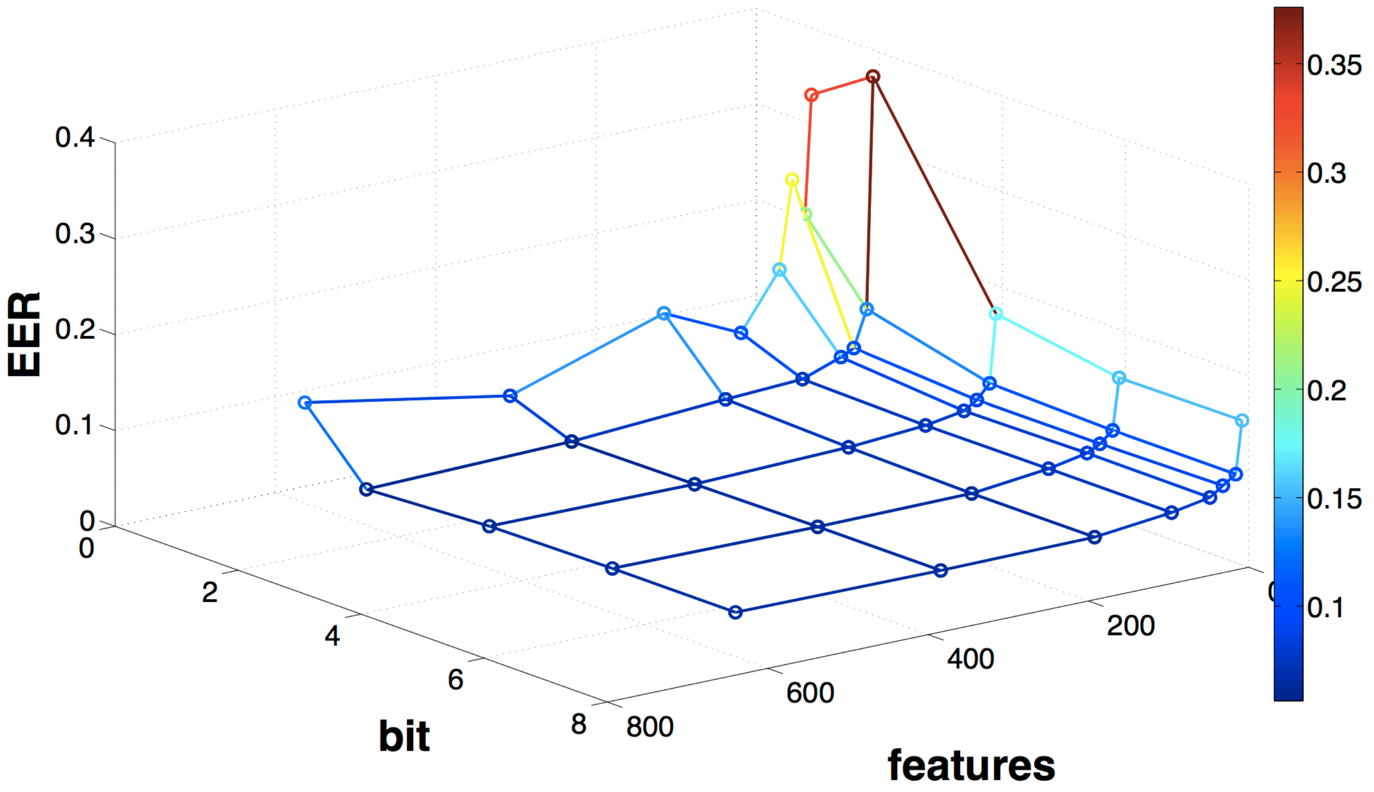
- N_R = number of rings
- N_A = number of arcs
- $N_S = N_R \times N_A$ = number of sectors
- N_F = number of filters
- $N_V = N_F \times N_S$ = size of feature vector
- N_θ = number of rotated templates for enrolled user (9)

Representation accuracy

- N_b= number of bits for each feature (from 1 to 8)

Optimization of fingercode representation

We evaluated the impact on matching accuracy (EER) by relying on a database with 408 fingerprints acquired by a CrossMatch verifier 300 sensor (500 dpi, 512x480 pixels).



Selected configuration

Size of feature vector

- $N_R = 3$
- $N_A = 8$
- $N_S = 24$
- $N_F = 8$ (configuration C) or 4 (configuration D)
- $N_V = 192$ (C) or 96 (D)
- N_θ = number of rotate templates for enrolled user (9)

Representation accuracy

- N_b= 1bit, 2 bits

Distance computation: classical approach

- The Squared Euclidean distance between an encrypted and a known vector is easy to compute by relying on HE

$$d(t,x)^2 = \sum_{1=1}^{n} (t_i - x_i)^2 = \underbrace{\sum_{i=1}^{n} t_1^2}_{\text{computed by the client}} + \underbrace{\sum_{i=1}^{n} x_1^2}_{\text{computed by the server}} - 2\sum_{i=1}^{n} \underbrace{t_i\, x_i}_{\text{computed by the server via HE}}$$

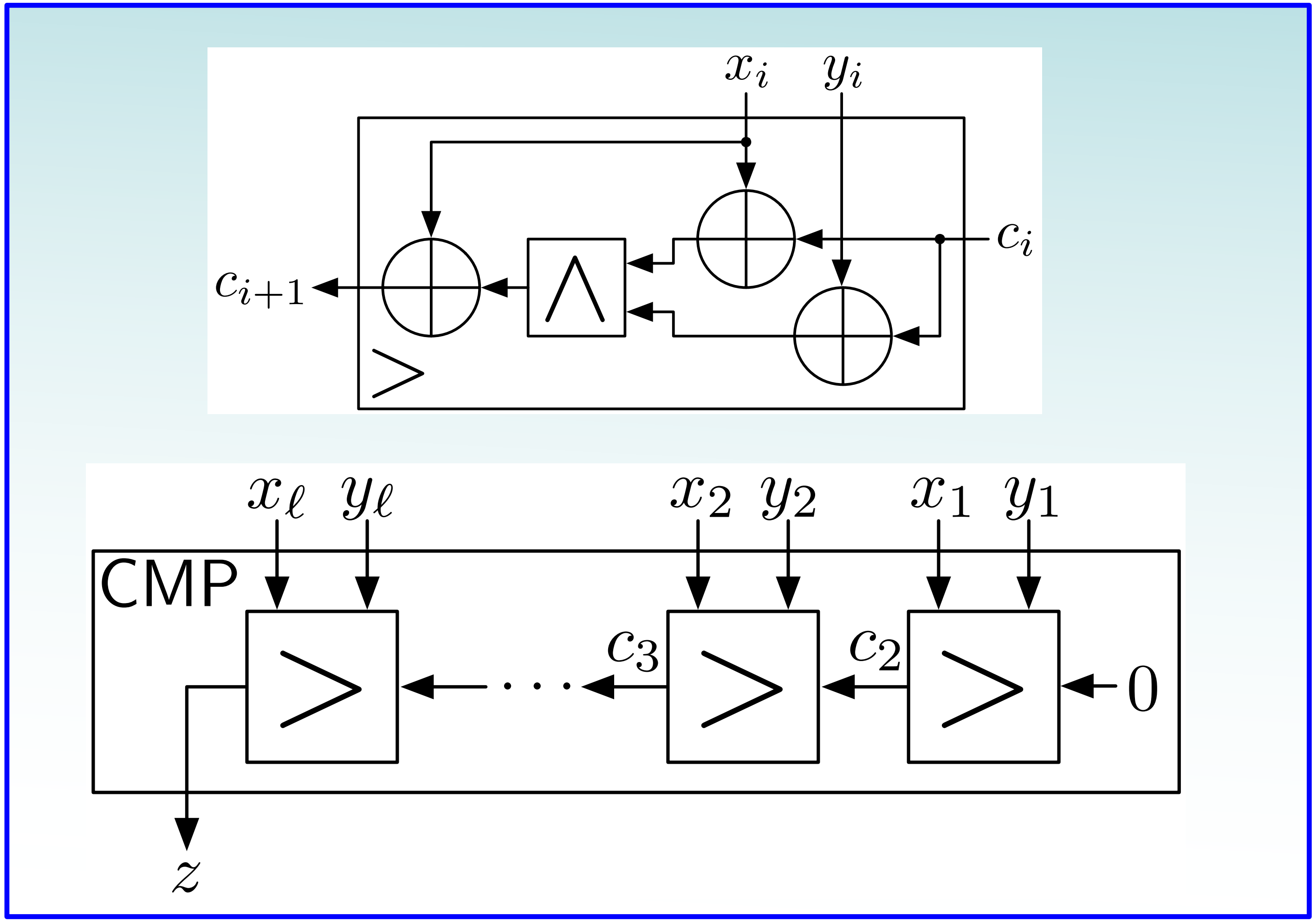
$$E[d^2] = E\left[\sum_{i=1}^{n} t_1^2\right] E\left[\sum_{i=1}^{n} x_1^2\right] \prod_{i=1}^{n} E[t_i]^{-2x_i}$$

Threshold comparison

- Comparison is by far easier through GC’s

- Hybrid solution
    - distances computed via HE are converted into (secret) bits
    - Pass from HE to GC representation
    - Run the GC

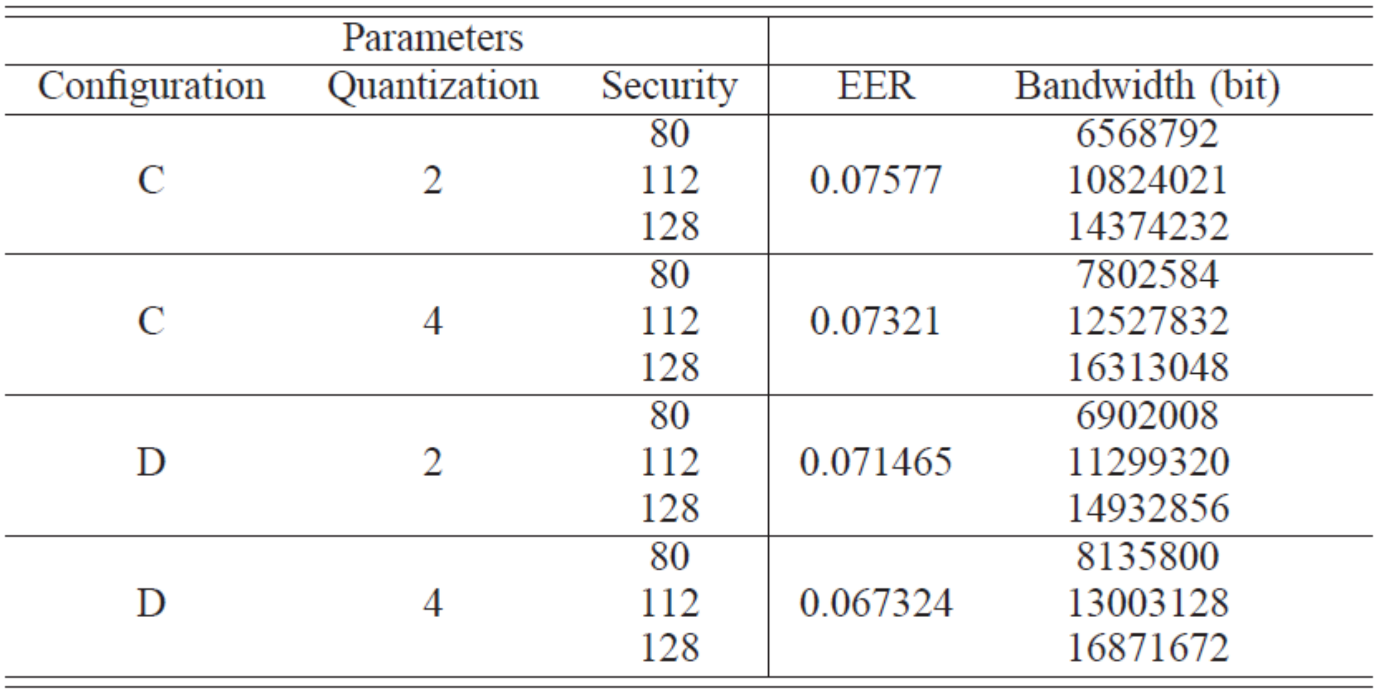
Comparison circuit



Performance (bandwidth)

TABLE III

PERFORMANCE OF THE PROPOSED METHOD WITH A DATABASE OF 408 ENTRIES (3672 FEATURE VECTORS).

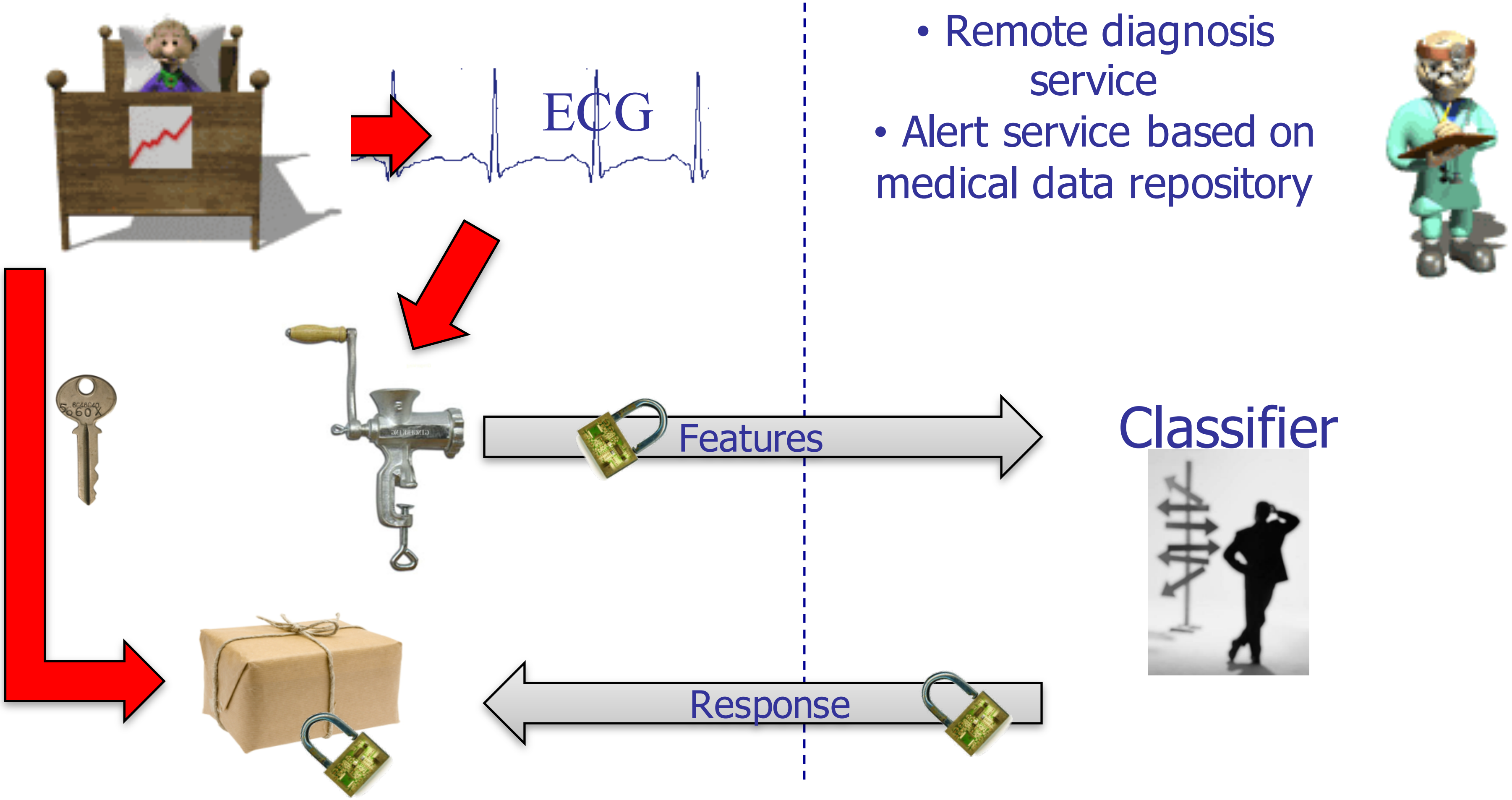


Parameters				
Configuration	Quantization	Security	EER	Bandwidth (bit)
C	2	80	0.07577	6568792
		112		10824021
		128		14374232
C	4	80	0.07321	7802584
		112		12527832
		128		16313048
D	2	80	0.071465	6902008
		112		11299320
		128		14932856
D	4	80	0.067324	8135800
		112		13003128
		128		16871672

Performance (execution time)

- Set-up
  - Java-based implementation
  - PC-platform (clock 2GHz, RAM 2GByte)
  - Pailler (key = 1024 bits) + GC (t = 80 bits)
  - 96 features, 2 bits per feature
- **Computing time:**
  - **time: < 0.1 sec for template**
- Similar performance with
  - face recognition, iris recognition

Remote classification of ECG signals



Client

Server

- Remote diagnosis service
- Alert service based on medical data repository

ECG

Features

Classifier

Response

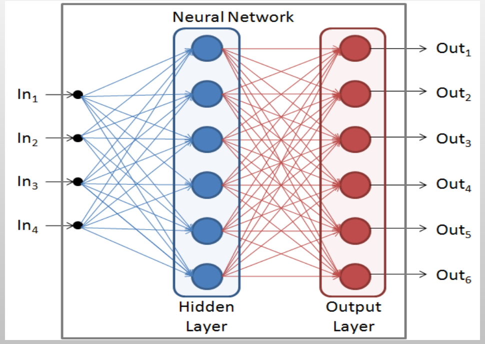
The SSP protocol

Client

ECG
↓
Preprocessing
↓
Feature extraction(AR model)

- The client protects
  - ECG data (features)
- Server protects
  - NN weights

neural net classification

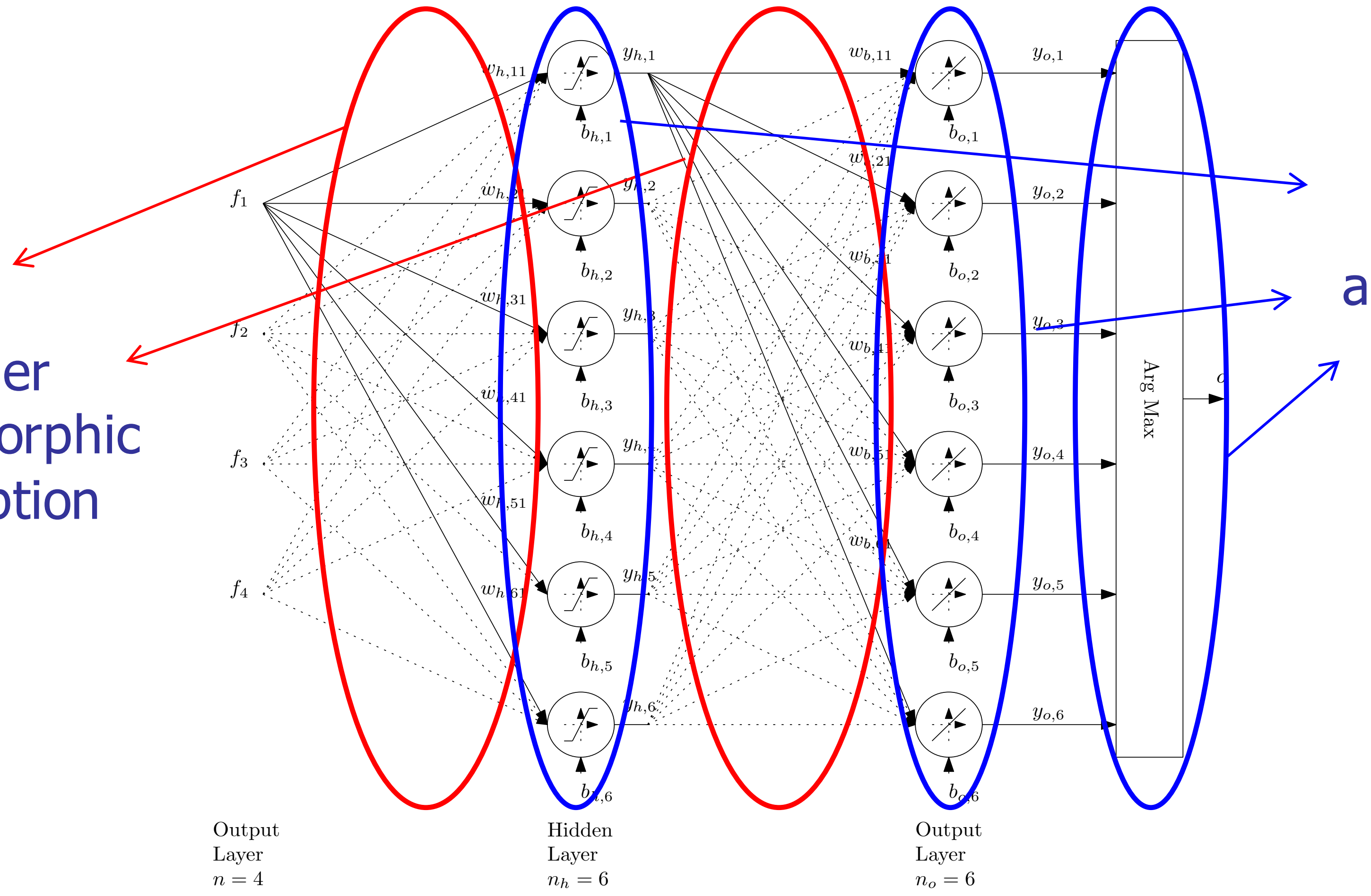


NN parameters

Server

Result

Secure computing with NN

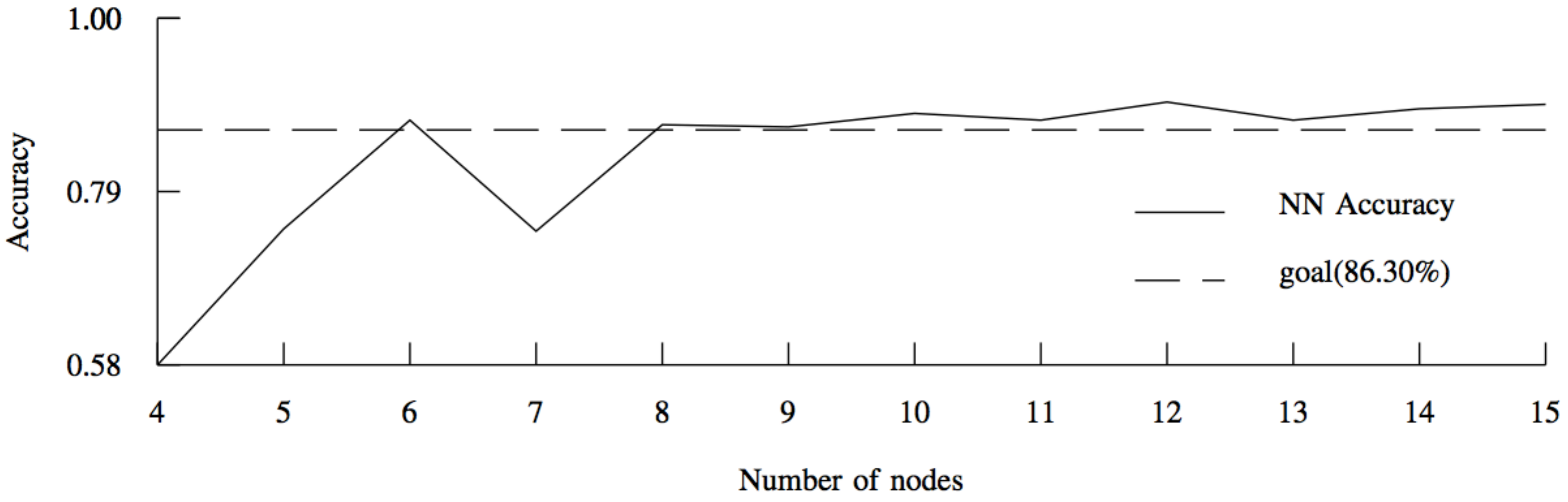


Pailler Homomorphic encryption

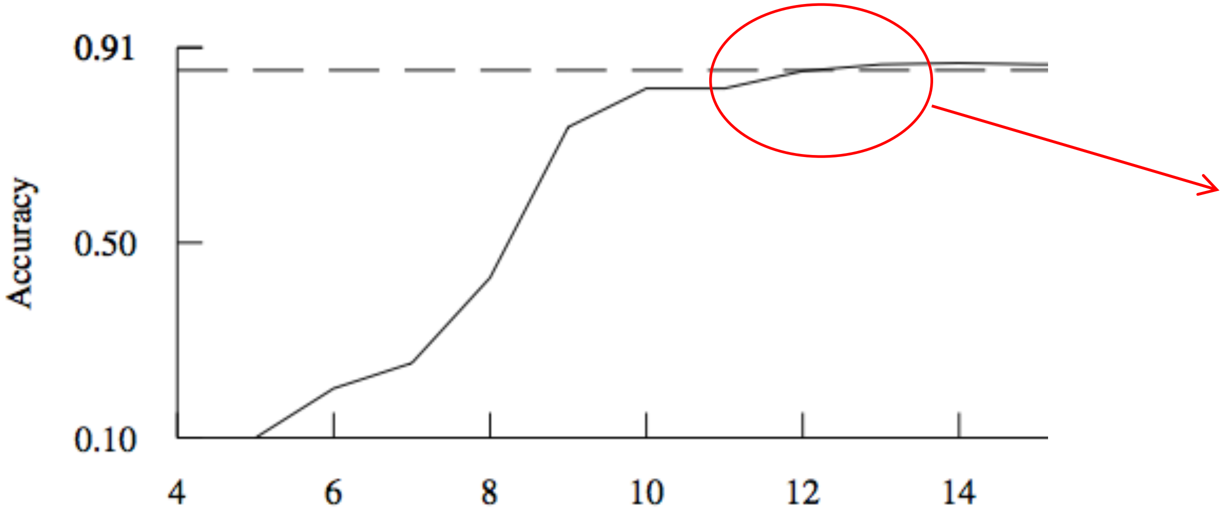
GC approach

NN topology

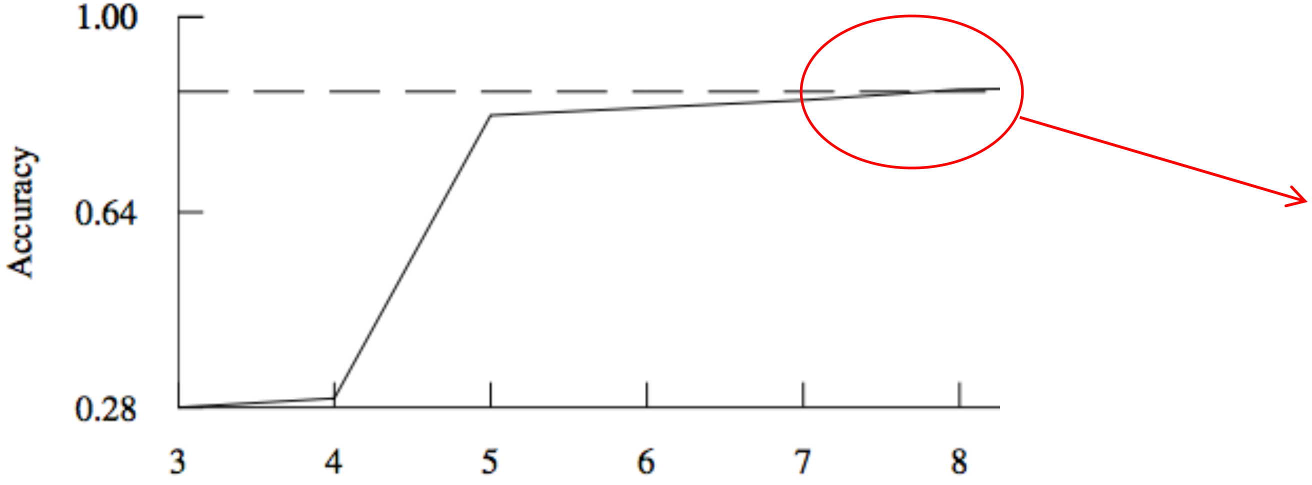
- The number of nodes in the input and output layers are set by the problem
- The number of internal nodes is determined experimentally



Representation accuracy



Representing the input and intermediate values with 12 bits is enough to achieve the best classification accuracy



7-8 bits are enough to represent the output of the NN (before taking the max)

Performance

- Set-up
  - Java-based implementation
  - PC-platform (clock 2GHz, RAM 2GByte)
  - Pailler + GC
- Communication complexity (per heart beat)
  - 80 Kbit (for short term security)
  - 120 Kbit (for long-term security)
- Running time
  - 3-4 seconds per heart beat
  - Almost real-time

Other applications

- s.p.e.d. technology has been applied to several other fields (to mention a few):
  - Privacy-preserving K-means clustering for social grouping
    - Z. Erkin. T.Veugen. T. Toft, R. L. Lagendijk, «Privacy preserving user clustering in a social network», Proc. of IEEE WIFS 2009, pp. 96-100
  - Privacy-preserving collaborative filtering for content recommendation
    - Z. Erkin. T.Veugen. T. Toft, R. L. Lagendijk,, «Generating private recommendations efficiently using homomorphic encryption and data packing», IEEE Trans. on Information Forensics and Security, vol. 7, no. 3, June 2012.
  - Smart grids
    - A. Rial, G. Danezis, “Privacy-preserving smart metering”, in Proceedings of the 10th annual ACM workshop on Privacy in the Electronic Society, 2011, pp. 49-60.

On-going research

- Efficiency, efficiency, efficiency
  - Crypto-level
    - more efficient primitives: fully homomorphic encryption
  - SP level
    - s.p.e.d. oriented algorithm design
    - Ad-hoc security measures
- Security against malicious adversaries
  - recent breakthrough: GC construction against malicious adversary at 7000 gates/s
  - FHE is becoming feasible
- System-level solutions, new applications

References

- R. L. Lagendijk, Z. Erkin, M. Barni, “Encrypted signal processing for privacy protection: conveying the utility of homomorphic encryption and multiparty computation”, *IEEE Signal Processing Magazine*, vol. 30, no. 1, pp. 82-105, January 2013
- M. Barni, G. Droandi, R. Lazzeretti, “Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing”, IEEE Signal Processing Magazine, vol. 32 no. 5, pp.66-76, September 2015