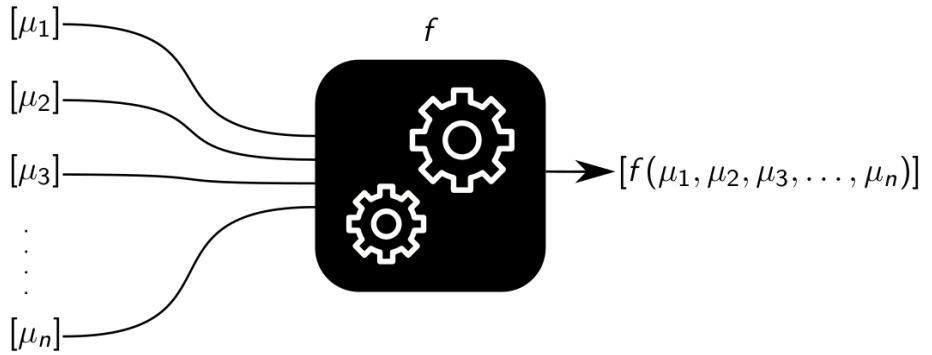
Securing Data Analytics with Homomorphic Encryption

encrypt

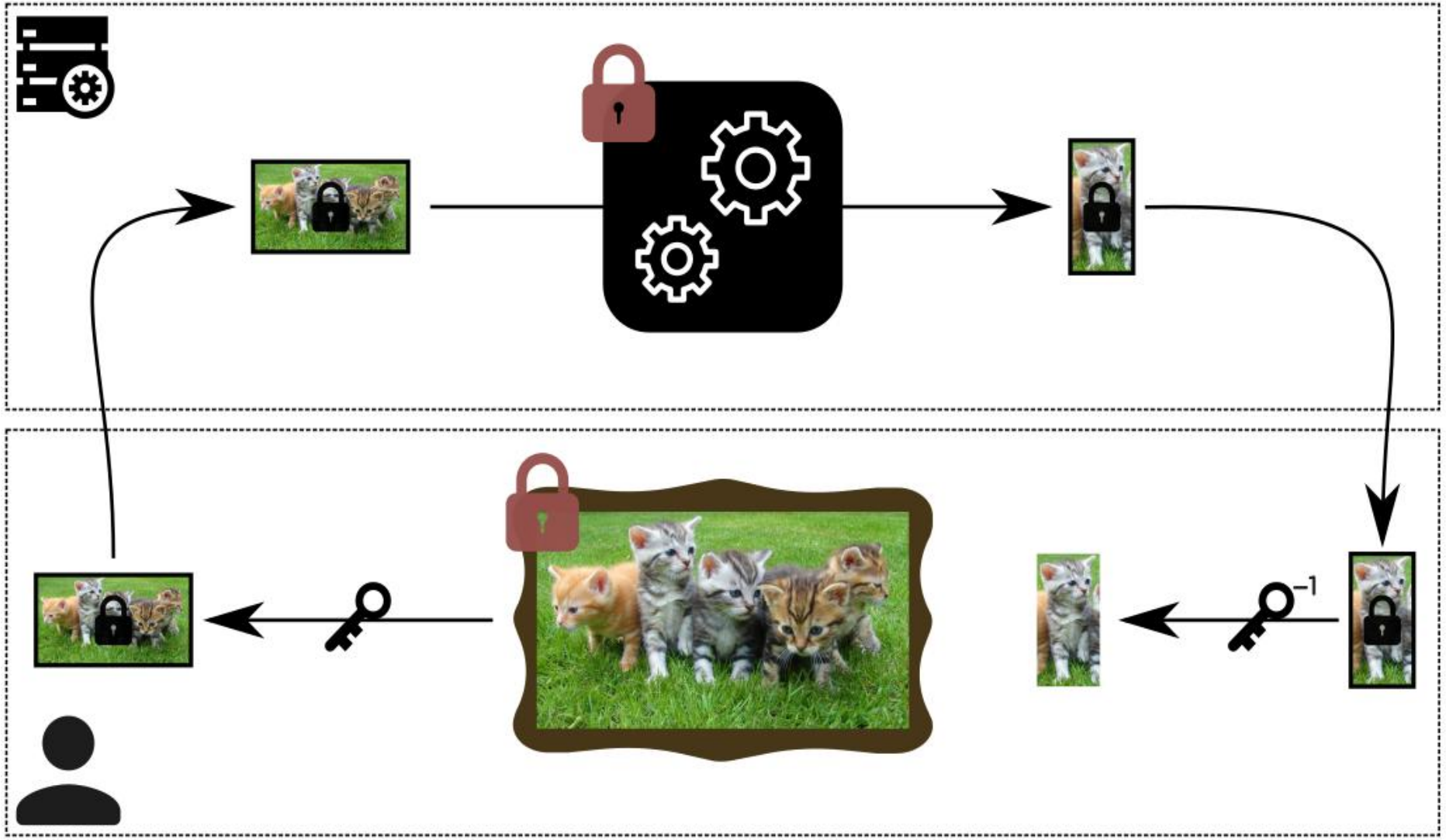
Homomorphic encryption

- **Homomorphic encryption (HE):** Recent cryptographic technique that allows a remote server to perform blind computations directly on encrypted data.



- The server has no information on the clear data, not even the result of its own computations.
- This ensures the protection of the data not only during sharing and storage, but also during computation.
- Specific encryption/decryption algorithms (not AES or RSA etc.)

Minimal workflow for HE



encrypt

Technical challenges

- Concretely, the building blocks are additions and multiplications on discrete rings (e.g. integers, booleans).
- **Fully** HE : allows homomorphic evaluation of any polynomial on input encrypted data

  so theorically… allows anything ?
- Limited multiplicative depth
- Performances of the schemes/libraries recently improved so that it is possible now to scale up for real-life use.

  - Batching : Simple instruction multiple data for parallelism.

  - TFHE : logic gates with possibly high multiplicative depth.

Homomorphic encryption in action

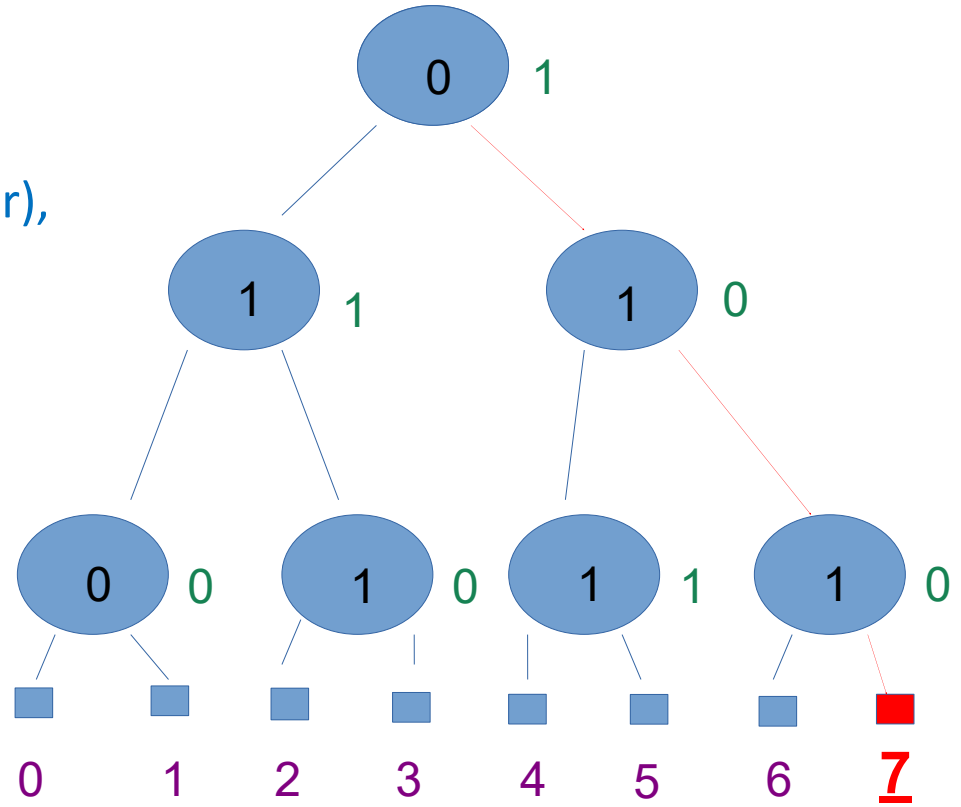
- Efficient evaluation techniques for specific algorithms.
- Finding such specific per-algorithm efficient evaluation is part of the work of the community.

- HE applied to ENCRYPT use-cases:

1. *Blind evaluation of binary decision trees (financial domain).*
2. *Blind computation on an encrypted table (medical domain).*
3. *Blind search on a blacklist of IP addresses (CTI domain).*

Binary decision trees (financial domain)

- HE evaluation of binary decision trees from an already trained random forest model.
- **Input:**
  - an encrypted parameter per node (from a model owner),
  - an encrypted piece of data (from a user).
- **Output:**
  - the encrypted tag of some leaf.



- Purpose: know if a client/institution is eligible for a bank loan.
- Each bit comparison corresponds to a question **e.g.**
  - *"has the client a source of income greater than X euros per month?"*
  - *"has the client any outstanding debt?"*

encrypt

Binary decision trees (financial domain)

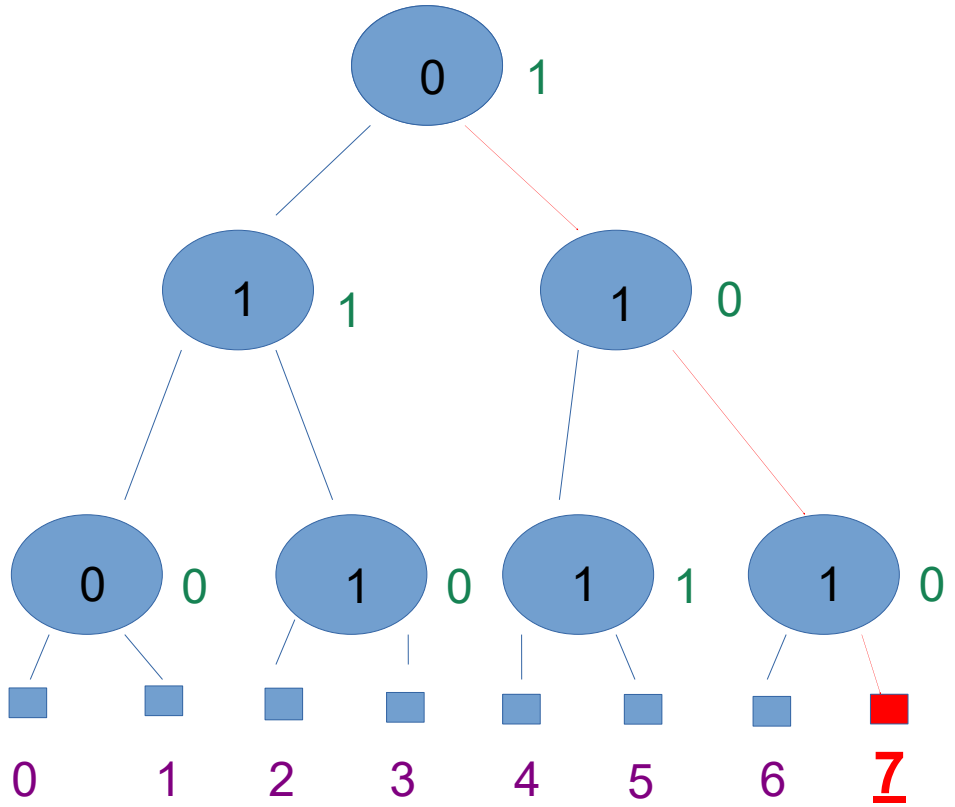
- FHE evaluation:
  only + and x allowed.

- For all path (root to leave), compute the list of the

  [(x_i + t_i) + dir_i]

  (« + » is a XOR ; dir_i is 0 for right, 1 for left).

- **[111]** for the right path.
  Product : 1x1x1=**1** (0 for other paths)



Binary decision trees (financial domain)

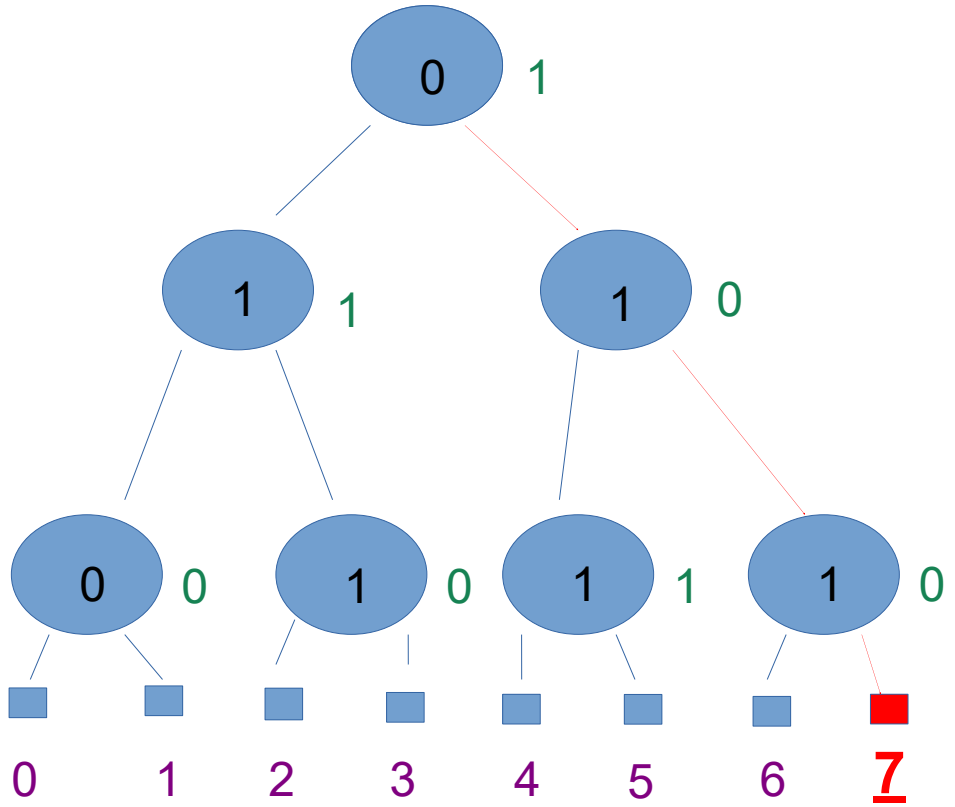
- FHE evaluation:
  only + and x allowed.

- For all path (root to leave), compute the list of the

  [(x_i + t_i) + dir_i]

  (« + » is a XOR ; dir_i is 0 for right, 1 for left).

- **[111]** for the right path.
  Product : 1x1x1=**1** (0 for other paths)



0x1 + 0x2 + … + 0x6 + 1x7 = **7**

Computation on an encrypted table (medical domain)

- **Main idea :** Blind checking in a table if some metadata from a patient take specific values
  (if at least one match is found, the patient must be reported and highlighted).

Patient id	Sex	Age range	Surgeries	…	…
Patient 1	1	3	0	…	…
Patient 2	0	2	2	…	…
…	…	…	…	…	…
…	…	…	…	…	…

Computation on an encrypted table (medical domain)

- **Workflow:**
- The table is filled by different medical actors from different departments.

- The table is encrypted with FHE and sent to the oncology department.

- The oncologist works on this encrypted table by means of homomorphic analysis.

- If some columns show specific values

  (i.e. IF column R=1 or column X=1),

  then the patient ID must be reported and highlighted to the oncologist.

Patient id	Sex	Age range	Surgeries	…	…
Patient 1	1	3	0	…	…
Patient 2	0	2	2	…	…
…	…	…	…	…	…
…	…	…	…	…	…

- The oncologist decrypts the file and decides whether the patient needs a more accurate follow-up.

Computation on an encrypted table (medical domain)

Some technical details:

- **Pre-processing (before encryption):** 1 for the specific target values, 0 otherwise.

- **HE evaluation:**
  - sum of each involved cell

    e.g. **[*P1,A1*] + [*P1,A3*] + [*P1,A4*]**

    (*Px, Ax* means line of patient *x*, column of attribute *x*)

  - homomorphic product with a nonzero random mask

    ([*P1,A1*] + [*P1,A3*] + [*P1,A4*])x[mask]

    = [(*P1,A1 + P1,A3 + P1,A4*) x mask]

0	1	1	0	...	...
0	0	1	1	...	...
...	...	...	...	...	...
...	...	...	...	...	...

- **Decryption:**
  The result **(*P1,A1 + P1,A3 + P1,A4*) x mask** is decrypted. Thanks to the mask it gives no information except the required one. If this result is not zero, then the patient ID is reported and highlighted to the oncologist.

Private information retrieval (CTI domain)

- Blind search of an IP address from CTI extraction in an encrypted blacklist of IP addresses.
- **Input:**

  - an encrypted IP address;
  - an encrypted list of IP addresses (the blacklist).

- **Output:**
  A ciphertext that allows (after decryption) to know if a match has been found.

- PIR algorithm based on a variant of the Paillier homomorphic encryption scheme:
  - allows one level of multiplication,
  - secure in the CCA model,
  - security based on modular arithmetic.

Private information retrieval (CTI domain)

- Before encryption, the blacklist is put in the form of a table **T** of lists of IP addresses such that the IP address x is in the cell H(x).

  H : cryptographic hash function.

  *x belongs to **T[**H(x)**]**.*

- **Search request** for an IP address x:

  H(x) is computed, then a table **S** is created, filled with 1 in the cell H(x) and 0 elsewhere

  - ***S[**H(x)**]** = 1,*
  - ***S[**y**]** = 0 for all y distinct to H(x),*

  encrypted and sent to the server.

- **HE evaluation:** sum_y ([**T[**y**]**][**S[**y**]**]) = [sum_y **T[**y**]S[**y**]**]
  = [0+…+0+**T[**H(x)**]S[**H(x)**]**+0+…] = [**T[**H(x)**]**]

- **Decryption**: x is retrieved in **T[**H(x)**]** if a match has been found.

Thank you for your attention

Find out more:

https://encrypt-project.eu/

encrypt-project

@encrypt_project

encrypt