Privacy preserving screening protocol

Arsème Vadèle Djeufack Nanfack[1] Claire Guichemerre[2]
Amaury Joly[3] Léo Louistiserrand[4] Kaïra Neily Sanon[5]

[1]Université de Picardie Jules Verne, MIS
[2]Université de Rennes, CNRS, IRISA
[3]Université Aix-Marseille, CNRS, LIS
[4]Université de Lorraine, CNRS, Inria, Loria
[5]Université de Caen, CNRS, GREYC

October 18, 2024

Introduction

ScreenAct

ScreenAct is an aggregator plateform of clinical trials

- help patients to find clinical trials
- help the health professional to watch and follow the local clinical trials
- Provide a tool to monitor the set of available patients

Introduction

ScreenAct

ScreenAct is an aggregator plateform of clinical trials

- help patients to find clinical trials
- help the health professional to watch and follow the local clinical trials
- Provide a tool to monitor the set of available patients

Key Features

- an AI based approach to labelised clinical trials
- smart matching algorithm
- a huge and up-to-date database of clinical states

Context

Their needs for the future of Screenact

- a fully (or the most we can) automatic application of matching between patients and promoter
  - with a fully privacy preservation of patient's private data
  - a realistic complexity to deploy it

Context

Their needs for the future of Screenact

- a fully (or the most we can) automatic application of matching between patients and promoter
  - with a fully privacy preservation of patient's private data
  - a realistic complexity to deploy it
- found and/or exclude approach to achieve these needs

Model Defintion

Actors

- A is the set of patients
- P is the set of promoters
- s is the solution
- C is the set of CROs

Model Definition

Model operations

- $DB = \{(id_1, d_1), (id_2, d_2), ..., (id_n, d_n)\}$

With id_x a unique id
and d_x the data associated to the unique id on the base

DB_s are health data and $DB_@$ are contact data of patients
DB term is use for $DB_s \cup DB_@$

Model Definition

Model operations

- $DB = \{(id_1, d_1), (id_2, d_2), ..., (id_n, d_n)\}$
- req = [requirements to satisfy]

req is a binary vector

Model Definition

Model operations

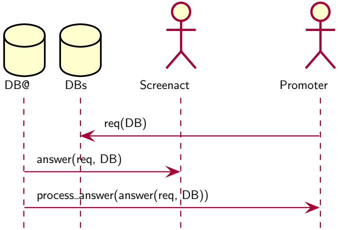
- $DB = \{(id_1, d_1), (id_2, d_2), ..., (id_n, d_n)\}$
- req = [requirements to satisfy]
- $answer(req, DB) = \{x \in DB | \text{x satisfies req}\}$

x satisfies req iff $req \cdot x \geq 1$

Model Definition

Model operations

- $DB = \{(id_1, d_1), (id_2, d_2), ..., (id_n, d_n)\}$
- $req =$ [requirements to satisfy]
- $answer(req, DB) = \{x \in DB | \text{x satisfies req}\}$
- $processed\_answer(answer(req, DB)) = ...$ (e.g. $|S|$ or $|\{x[1] | x \in S\}|$)



Theoritical Models

Synchronous

All messages are received at a same, known and constant time
All participant are processing the messages at the same, known and constant speed

Reliable

let E be the set of participants
$\forall a \in E, \forall m \text{ s.t. } send_a(m) \Rightarrow \forall b \in E - \{a\}, recv_b(m)$

Threat Model

Byzantin Promoter Tolerant

$$\forall p \in P, view(p) = \bigcup_{n=0}^{\infty} processed\_answer(answer(req_p^n, DB))$$

$$\approx$$

$$\overline{view}(p) = \bigcup_{n=0}^{\infty} processed\_answer(answer(req_p^n, \sigma(DB)))$$

"Our solution is Byzantin Promotor Tolerent iff the promotor is unable to distinguish the results computed from a legit database and from a permuted one."

Threat Model

Honest but Curious Solution Tolerent

$$view(s) = \bigcup_{n=0}^{\infty} \bigcup_{p=0}^{|P|} answer(req_p^n, DB)$$

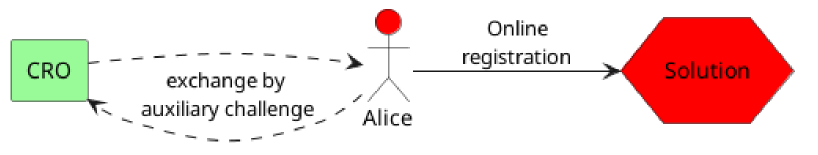
$$\approx$$

$$\overline{view}(s) = \bigcup_{n=0}^{\infty} \bigcup_{p=0}^{|P|} answer(req_p^n, \sigma(DB))$$

"Our solution is Honest but Curious Solution Tolerent iff the Solution is unable to dinstinguish the results produce by requests submited by promoter on a legit database and a permuted one"

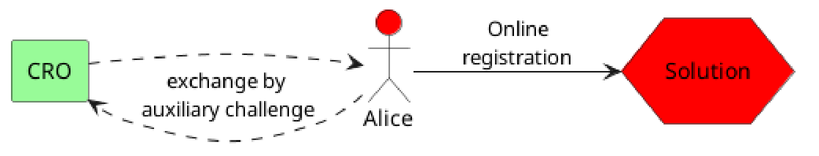
Problem statement

Patient enrollment



Problem statement

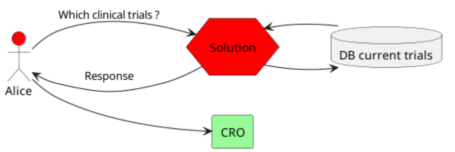
Patient enrollment



Our Goal

Pull version:

- A patient: Searches for clinical trials

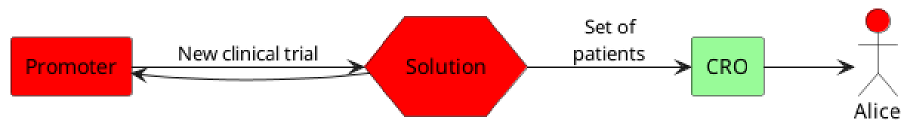


Problem statement

Our Goal

Push version:

- The promoter: New clinical trial adds to the database
- The promoter: Learns statistics on patients matching the clinical trial requirement
- The selected patients: Notified



Model properties (1/2)

Databases unlinkability

Without the knowledge of the link database, there is no way to map a line of $DB_{@}$ with a line of DB_s

Model properties (1/2)

Databases unlinkability

Without the knowledge of the link database, there is no way to map a line of $DB_{@}$ with a line of DB_s

Patient unicity

$\nexists x, y \in DB$ s.t. $x \neq y$ and $x[id] = y[id]$ where *id* is the unique identification of a patient in *DB*

Model properties (2/2)

Patient revocability

$\exists x \in DB_{@}$ s.t. $x[@A] = Alice$, $\exists y \in DB_s$ s.t. $x[id] = y[id]$
$Revoke(Alice, DB) = DB'$
$\nexists z \in DB'$ s.t. $z[id] = y[id]$

Model properties (2/2)

Patient revocability

$\exists x \in DB_{@}$ s.t. $x[@A] = Alice$, $\exists y \in DB_s$ s.t. $x[id] = y[id]$
$Revoke(Alice, DB) = DB'$
$\nexists z \in DB'$ s.t. $z[id] = y[id]$

Patient confidentiality

Each patient $a \in A$ is the only one who knows his line $x \in DB_s$ s.t. $x[@A] = a$

Model properties (2/2)

Patient revocability

$\exists x \in DB_{@}$ s.t. $x[@A] = Alice$, $\exists y \in DB_s$ s.t. $x[id] = y[id]$
$Revoke(Alice, DB) = DB'$
$\nexists z \in DB'$ s.t. $z[id] = y[id]$

Patient confidentiality

Each patient $a \in A$ is the only one who knows his line $x \in DB_s$ s.t. $x[@A] = a$

Confidential patient delivery

Each patient $a \in A$ is the only one who knows the result of a clinical trial.

Functional encryption

Alice

Bob

x

Carol

$f(x)$?

- Bob has a secret x
- Carol wants to compute $f(x)$
- Alice has a surprise tool that will help us later

Functional Encryption

Alice

Bob

x

Carol

$f(x)$?

Functional Encryption

Alice

Bob

Carol

x

$f(x)$?

$\boxed{(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}}$

Functional Encryption

Alice	Bob	Carol
		
	x	$f(x)$?

$(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

$\xrightarrow{\mathsf{pk}}$

Functional Encryption

Alice

Bob

Carol

x

$f(x)$?

$(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

$\xrightarrow{\mathsf{pk}}$

$c \leftarrow \mathsf{Enc}(x, \mathsf{pk})$

Functional Encryption

Alice

Bob

Carol

x

$f(x)$?

$(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

$\xrightarrow{\mathsf{pk}}$

$c \leftarrow \mathsf{Enc}(x, \mathsf{pk})$

$\xrightarrow{c}$

Functional Encryption

Alice

Bob

Carol

Bob: x

Carol: $f(x)$?

Alice: $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

Alice → Bob: pk

Bob: $c \leftarrow \mathsf{Enc}(x, \mathsf{pk})$

Bob → Carol: c

Carol → Alice: f

Functional Encryption

Alice

Bob

Carol

Bob: x

Carol: $f(x)$?

$(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

Alice → Bob: pk

$c \leftarrow \mathsf{Enc}(x, \mathsf{pk})$

Bob → Carol: c

Carol → Alice: f

$\mathsf{sk}_f \leftarrow \mathsf{Keygen}(f, \mathsf{msk})$

Functional Encryption

Alice

Bob

Carol

Bob: x

Carol: $f(x)$?

Alice: $(\mathrm{pk}, \mathrm{msk}) \leftarrow \mathrm{Setup}$

Alice → Bob: pk

Bob: $c \leftarrow \mathrm{Enc}(x, \mathrm{pk})$

Bob → Carol: c

Carol → Alice: f

Alice: $\mathrm{sk}_f \leftarrow \mathrm{Keygen}(f, \mathrm{msk})$

Alice → Carol: sk_f

Functional Encryption

Alice

Bob

Carol

Bob: x

Carol: $f(x)$?

Alice: $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}$

Alice → Bob: pk

Bob: $c \leftarrow \mathsf{Enc}(x, \mathsf{pk})$

Bob → Carol: c

Carol → Alice: f

Alice: $\mathsf{sk}_f \leftarrow \mathsf{Keygen}(f, \mathsf{msk})$

Alice → Carol: sk_f

Carol: $y \leftarrow \mathsf{Dec}(c, \mathsf{sk}_f)$

Reminder: homomorphic encryption

El-Gamal encryption

- Parameters: $(\mathbb{G}, p, g)$
- Keys: secret key : x, public key : $h = g^x$
- Encryption: $\text{Enc}(m) = (g^r, g^m h^r)$
- Decryption: $\text{Dec}(c_1, c_2) = c_2 c_1^{-x} = g^m$

Limitation

Decryption requires to compute a discrete log: small message space only

Theorem

$\forall m_1, m_2, Enc(m_1)Enc(m_2) = Enc(m_1 + m_2)$

DDH-Inner Product

f is the inner product with a vector $\mathbf{y}$.

DDH-IP

- Masterkey: $\mathsf{msk} = \mathbf{x} = (x_1, \ldots, x_l) \in \mathbb{Z}_p^l$
- Public key: $\mathbf{h} = (g^{x_1}, \ldots, g^{x_l})$
- Encryption: $\mathbf{m} = (m_1, \ldots, m_l) \rightarrow \mathbf{c} = (g^r, h_1^r g^{m_1}, \ldots, h_l^r g^{m_l})$
- Key derivation: $\mathsf{sk}_y = \mathbf{x} \cdot \mathbf{y}$
- Decryption: $c_0^{-\mathsf{sk}_y} \prod c_i^{y_i} = g^{\mathbf{m} \cdot \mathbf{y}}$

Application of DDH-IP

User manual

- Inputs are vectors
- Computations are inner products
- Message space has to be small

Structure of the data

Pathology	∅	0
	Melanoma	0
	Leiomyosarcoma	1
	Liver cancer	0
Medical history	∅	0
	Cardiac arrhythmia	0
	Myocardial infarction	1
	Hearth failure	0
Smoke ?		0
Med. history status	Qualified	1
	Incomplete	0
	Prefer not to say	0

$$\text{data} = (0, 0, \underbrace{1}_{\text{Leiomuosacoma}}, 0, 0, 0, \underbrace{1}_{\text{Myocardial infraction}}, 0, 0, \underbrace{1}_{\textit{Qualified}}, 0, 0)$$

Simple query

Query

Find all patients with Melanoma and Hearth failure

$$\mathbf{y} = (0, \dots 0, \underbrace{1}_{\text{Melanoma}}, 0, \dots, 0, \underbrace{1}_{\text{Hearth failure}}, 0, \dots, 0)$$

The patient is compatible if and only if $\mathbf{y} \cdot \mathbf{data} = 2$.

More subtle query

Query

Find all patients with a qualified Cardiac arrhythmia

		query
...	...	0
Medical history	Cardiac arrhytmia	1
Med. history status	Qualified	1
	Incomplete	2^{10}
	Prefer not to say	2^{10}

Possible outputs are 0, 1, 2^{10} and $2^{10}+1$

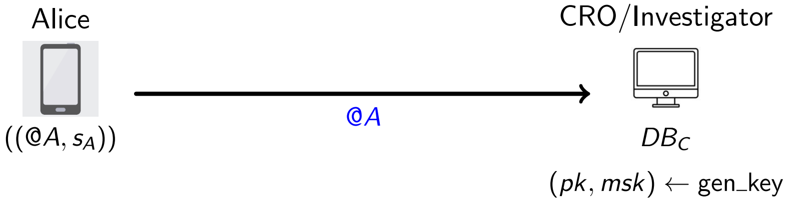
In the system used by the person authorised to handle health data, a key is generated.

CRO/Investigator

DB_C

$(pk, msk) \leftarrow$ gen_key

A Patient goes to be enrolled by the CRO/Investigator.



SOLUTIONS

1st Protocol: Use a functional encryption (3/11)

The CRO/Investigator saves his personal data and gives it a pseudonym and also the public key.

Alice

$((@A, s_A))$

$(pseudo_A =$ [1]$H(@A)$, pf$)$ ←

→ $@A$

CRO/Investigator

DB_C

$(pk, msk) \leftarrow$ gen_key

[1] H: a function (hash) that generates a unique, fixed-length fingerprint from input data, ensuring integrity, collision resistance, and resistance to inversion.

New data is added for each side.

Alice

($pseudo_A$, @ A, s_A, pk)

CRO/Investigator

$DB_c \cup \{(pseudo_A, @A)\}$
(pk,msk)

SOLUTIONS

1st Protocol: Use a functional encryption (5/11)

Use case of a enrolment in ScreenAct solution.

Alice

($pseudo_A$, @ A, s_A, pk)

Solution

DB_{sol}

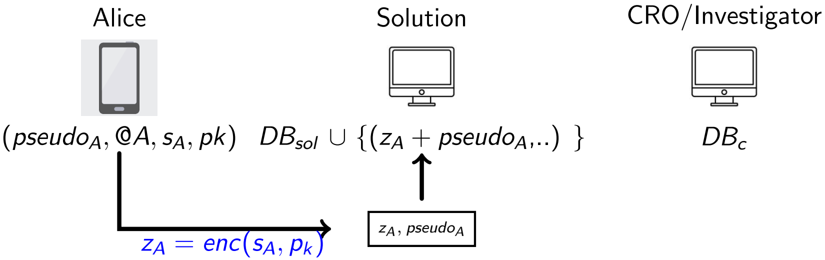
CRO/Investigator

$DB_c \cup \{(pseudo_A, @A)\}$
(pk,msk)

SOLUTIONS

1st Protocol: (Use a functional encryption 6/11)

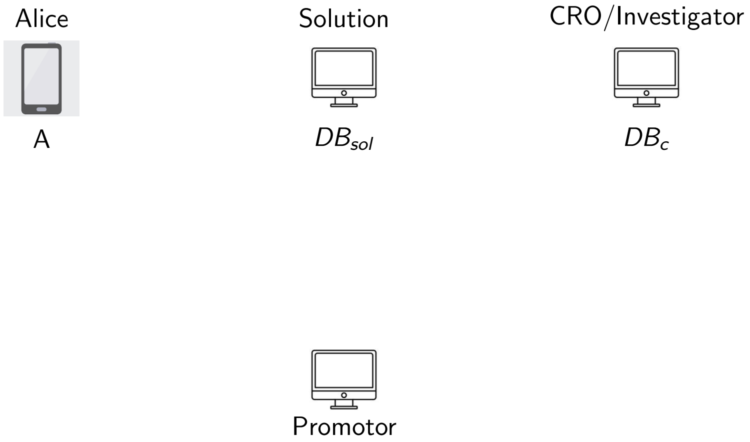
Alice registers by encrypting her health data.This result and her contact details are added to the ScreenAct database.



SOLUTIONS

1st Protocol: Use a functional encryption (7/11)

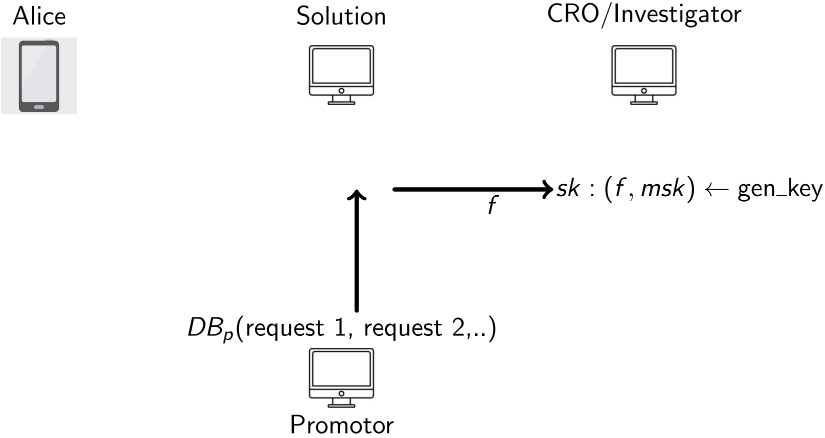
Use case: A promotor has a request for a clinical trial



SOLUTIONS

1st Protocol: Use a functional encryption (8/11)

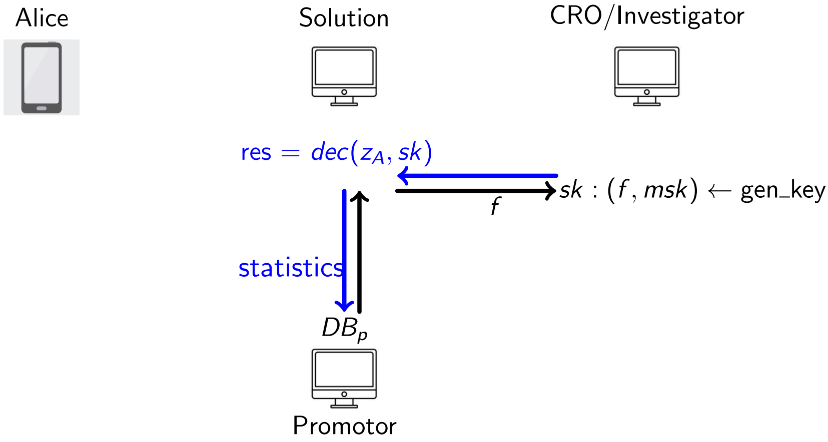
The request is sent to Screenact whom sends it to CRO/Investigator. By the way, CRO/Investigator generates a secret key to permit ScreenAct to access some statics.



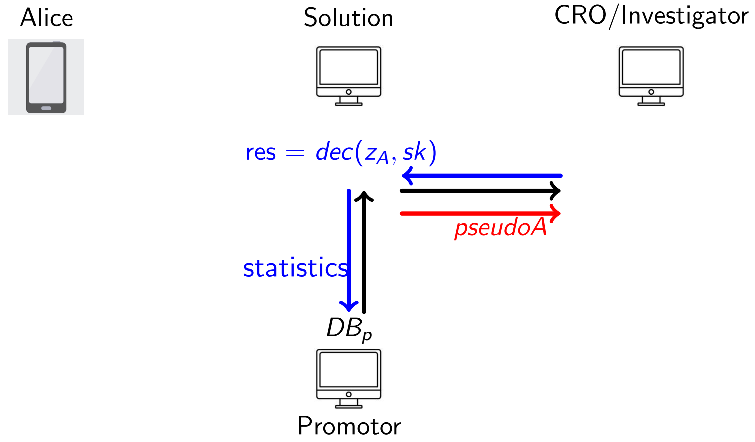
SOLUTIONS

1st Protocol: Use a functional encryption (9/11)

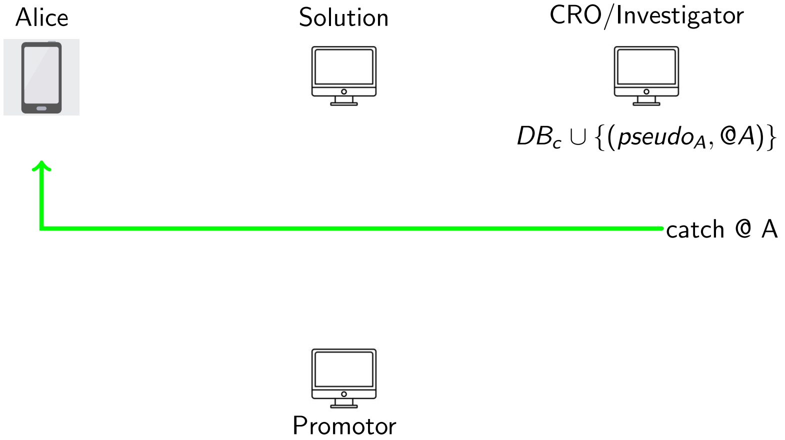
ScreenAct decrypts this secret and shares the informations needed by the promotor



ScreenAct decrypts this secret and share the informations needed by the promotor



CRO/Investigator use the filter to access the personal data of patients concerned (like Alice) and send their a notification.



Goal achieved

Properties

- CRO can only access the contact data
- Screenact can only access the encrypted health data
- Promoter can perform a wide range of operations to select patients
- Patients can suppress their data at any time

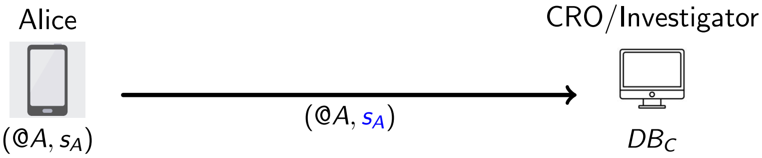
Limitations

- Requires a lot of cryptography
- High complexity

SOLUTIONS

2nd Protocol: Trust CRO/Investigator (1/6)

A Patient goes to be enrolled by the CRO/Investigator.



SOLUTIONS

2nd Protocol: Trust CRO/Investigator (2/6)

Deux databases were created.

Alice

(@ A, s_A)

CRO/Investigator

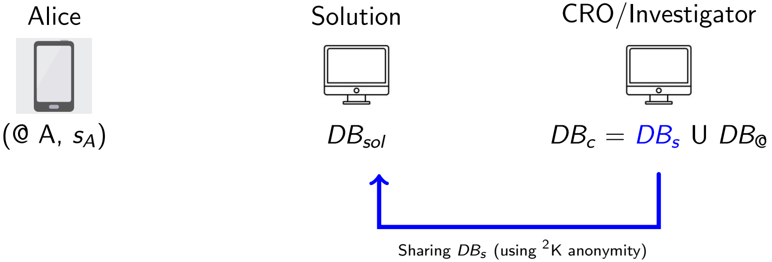
$DB_c = DB_s \cup DB_@$

Where $s_A \in DB_s$, $@A \in DB_@$

SOLUTIONS

2nd Protocol: Trust CRO/Investigator (3/6)

The database relative to the status of patients' health is anonymised.

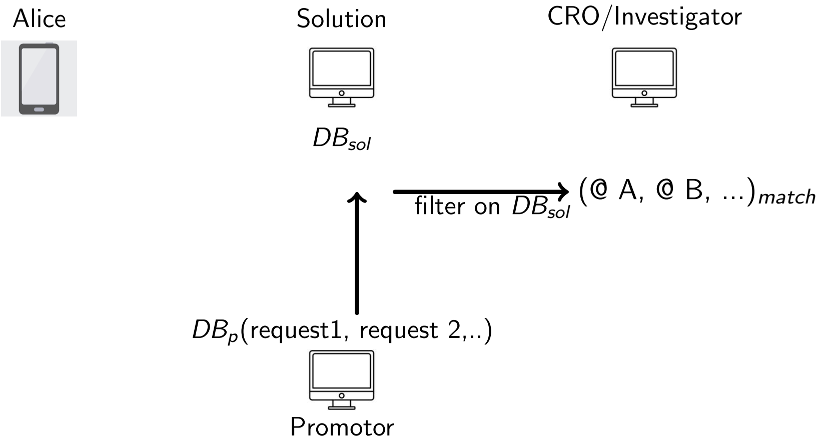


[2] k-anonymity is a data protection Protocol that ensures an individual cannot be distinguished from at least k other individuals within a dataset, thereby reducing the risk of re-identification.

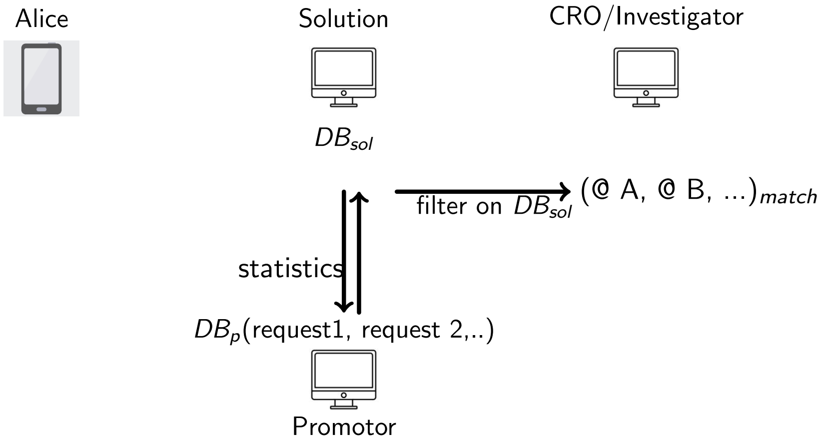
SOLUTIONS

2nd Protocol: Trust CRO/Investigator 4/6)

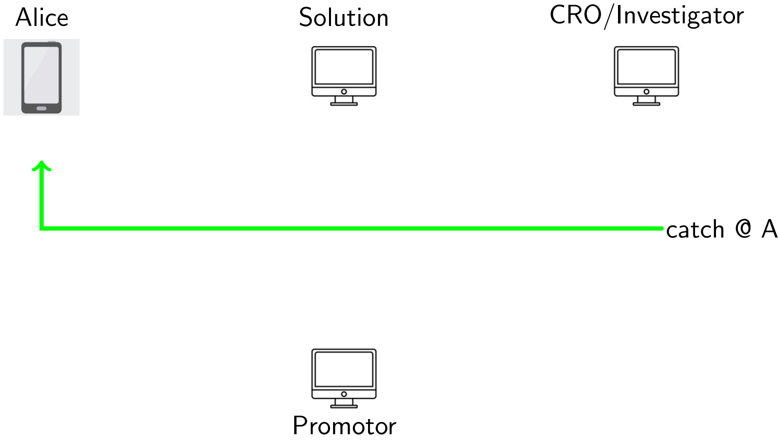
A promotor makes a request filtered by ScreenAct.



ScreenAct can give some statistics to the promotor



CRO/Investigator uses the filter to access the personal data of concerned patients (like Alice) and sends them a notification.



Demonstration

Conclusion

Two solutions proposed:

- A low complexity protocol without cryptography
- A more ambitious protocol based on a fancy primitive

Conclusion

Two solutions proposed:

- A low complexity protocol without cryptography
- A more ambitious protocol based on a fancy primitive

What's next ?

Conclusion

Two solutions proposed:

- A low complexity protocol without cryptography
- A more ambitious protocol based on a fancy primitive

What's next ?