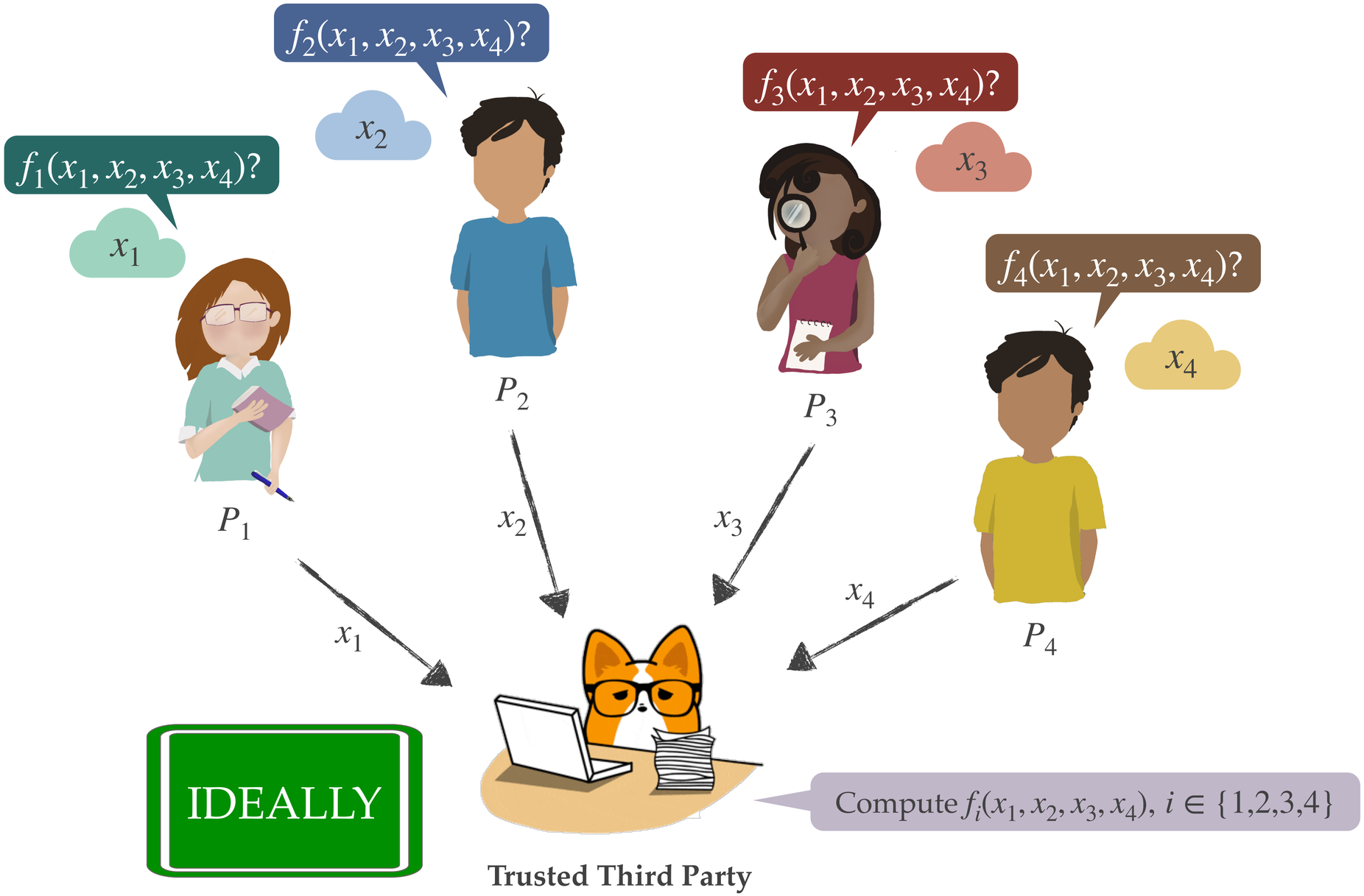
07 December 2021 : Vivek Arte

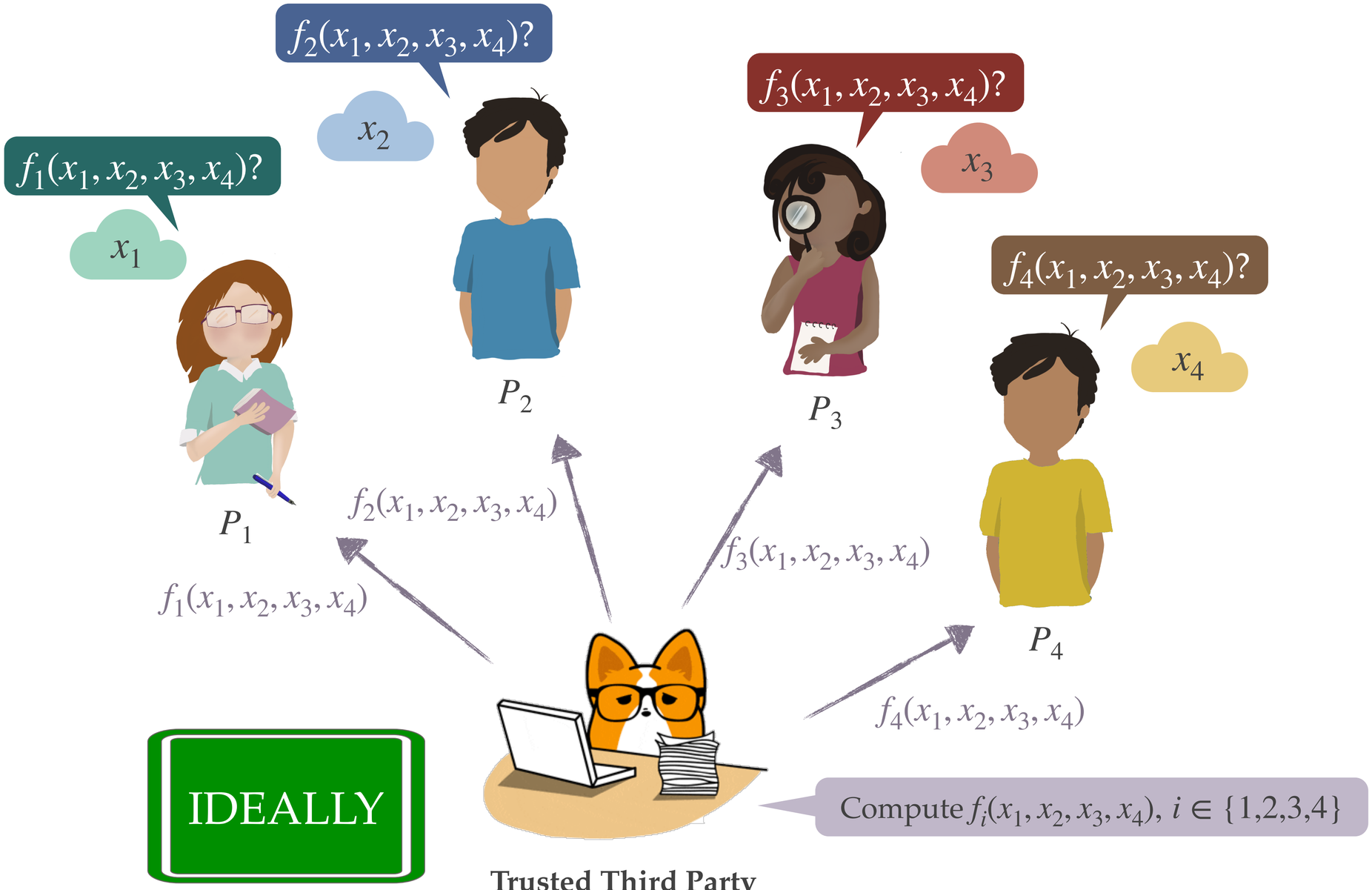
What are we PSIgning up for?

Analyzing Applications of Two-Party Private Set Intersection

Interactions between multiple parties

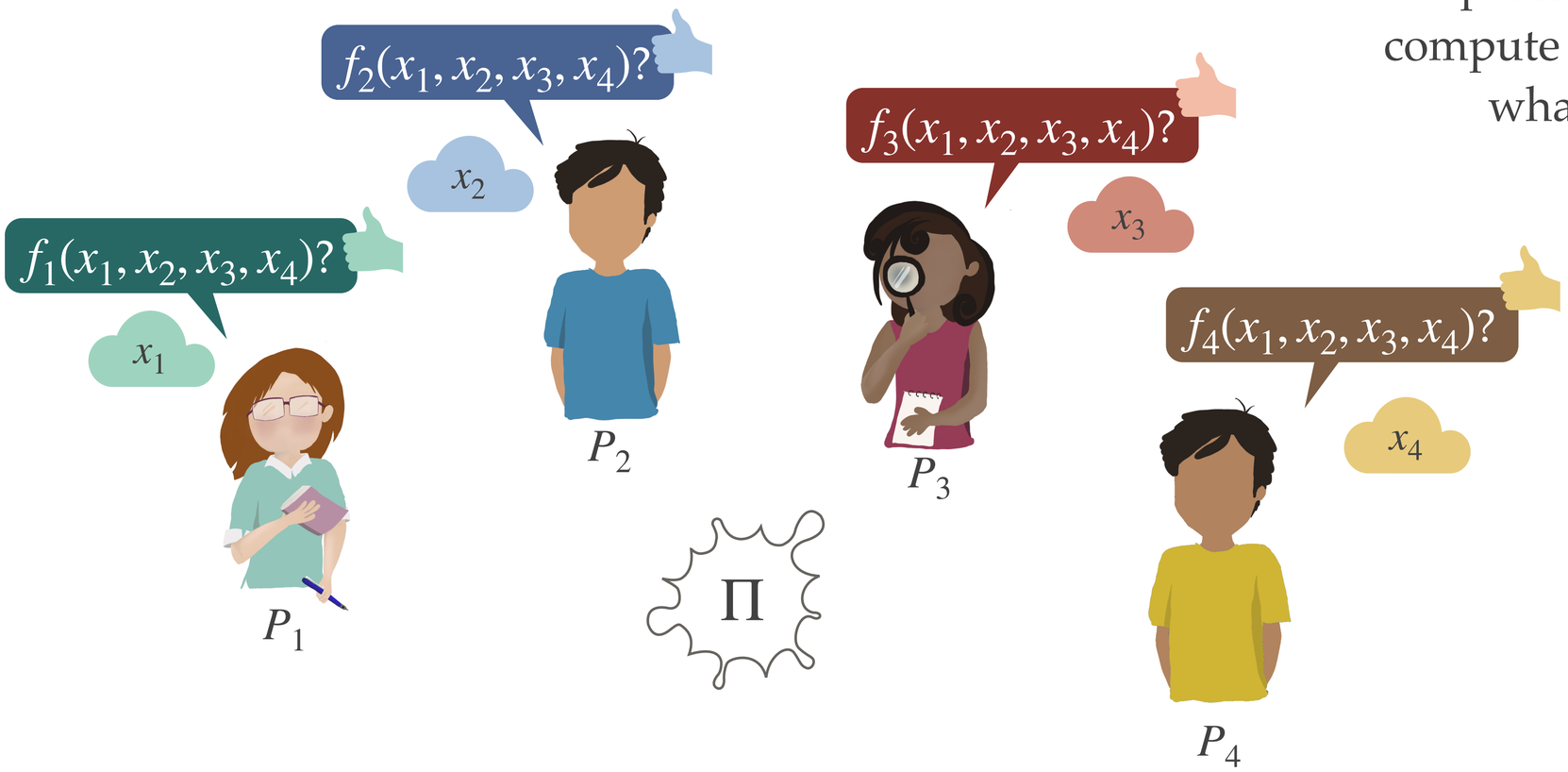


Interactions between multiple parties



Multiparty Computation (MPC)

[Yao86, GMW87]



A protocol Π that describes how the parties compute and communicate in order to achieve what they want from the interaction

Partisia

- Surveys
- Auctions
- Contract exchange

Google

- Password checkup
- Data aggregation

LIBRARIES

Fair Play Secure Function Evaluation · JustGarble · CrypTen · MOTION · swanky

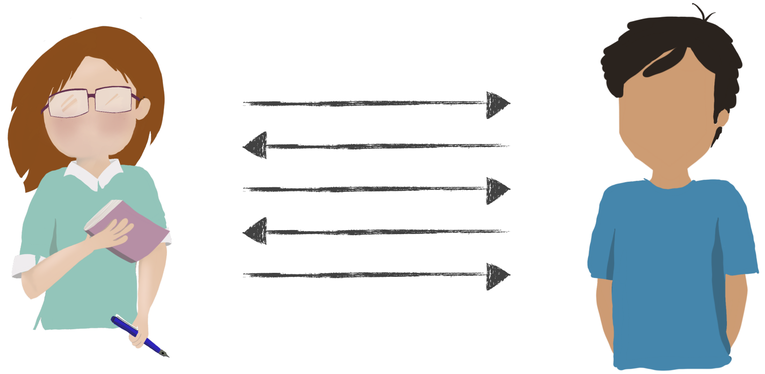
sharemind · Obliv-C · Rosetta · FRESCO EFFICIENT SECURE COMPUTATION FRAMEWORK · JIFF · MP-SPDZ · SCAPI

Two-party Private Set Intersection (PSI)

[FNP04]

A special case of multi-party computation

$\{x_1, x_2, \ldots x_n\} =$ X



Y $= \{y_1, y_2, \ldots y_n\}$

Alice

Bob

Output

$\perp$

$I = X \cap Y$

There are different choices here: Alice receives the intersection too, Bob receives only some function of the intersection, etc.

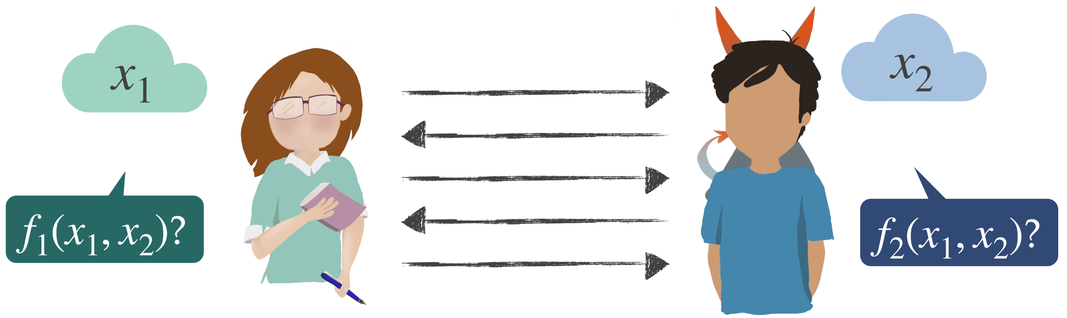
Variants

Symmetric version:	$f(X \cap Y)$	$f(X \cap Y)$
PSI-CA	$\perp$	$\lvert X \cap Y \rvert$
PSI-Sum	$\perp$	$\text{sum}(X \cap Y)$
f-PSI	$\perp$	$f(X \cap Y)$

Authorized - PSI

When is a protocol secure?

[HL10]



Correctness Party P_i should receive the value of $f_i(x_1, x_2)$

Privacy Party P_i should not learn anything more than its input and $f_i(x_1, x_2)$

We assume the adversary corrupts exactly one party

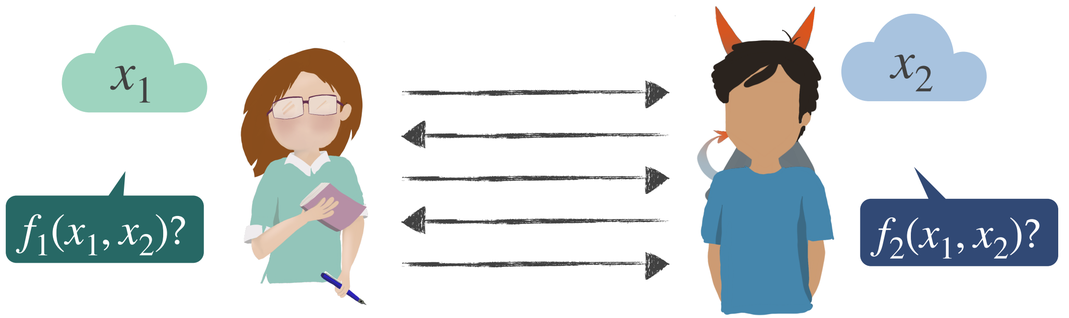
Independence of inputs Corrupted party's inputs should be independent of the honest party's inputs

Guaranteed output delivery Corrupted party should be unable to stop the honest party from receiving its result

Fairness Corrupted party should get their result iff the honest party receives their result

When is a protocol secure?

[HL10]



Correctness Party P_i should receive the value of $f_i(x_1, x_2)$

Privacy Party P_i should not learn anything more than its input and $f_i(x_1, x_2)$

We assume the adversary corrupts exactly one party

Independence of inputs Corrupted party's inputs should be independent of the honest party's inputs

arant output delivery Corrupted party should be unable to stop the honest party from receiving its result

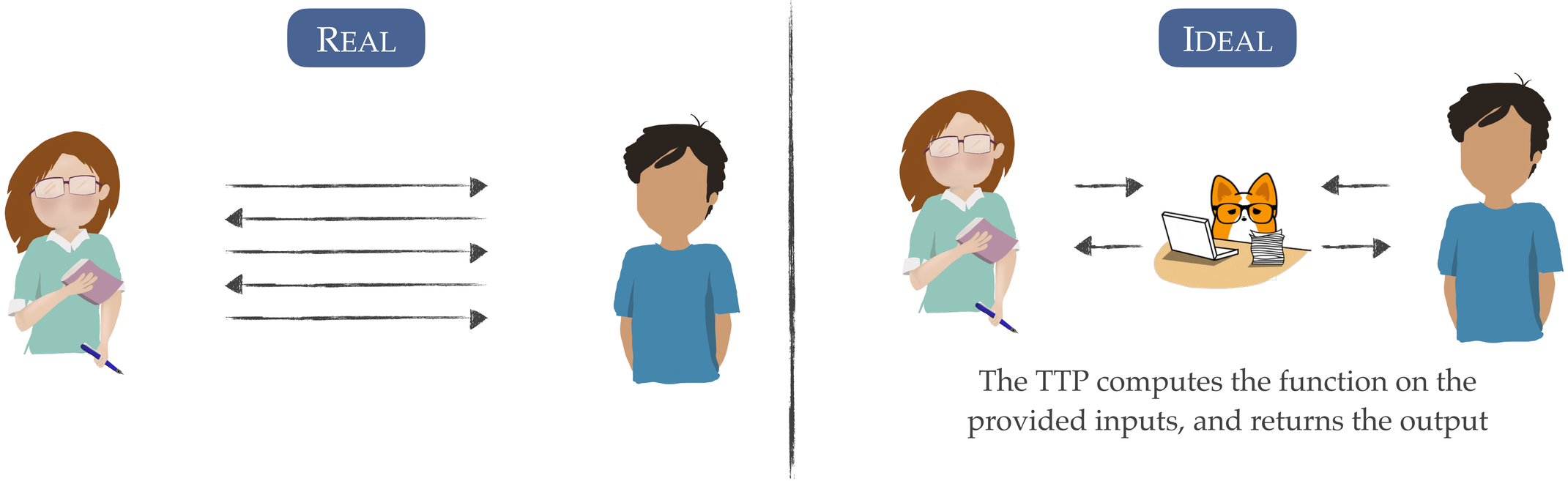
irne Corrupted party should get their result iff the honest party receives their result

In general, impossible for the two-party setting

… and so on

The Real/Ideal Paradigm

[Gol04, HL10]



Informally, security is achieved when any attack on the real world can also be carried out in the ideal world

There are different variants of security, based on the power of the adversary

Semi-honest | Malicious | Covert

Two-party protocols $\Longrightarrow$ we need only focus on the case where the adversary corrupts a single party

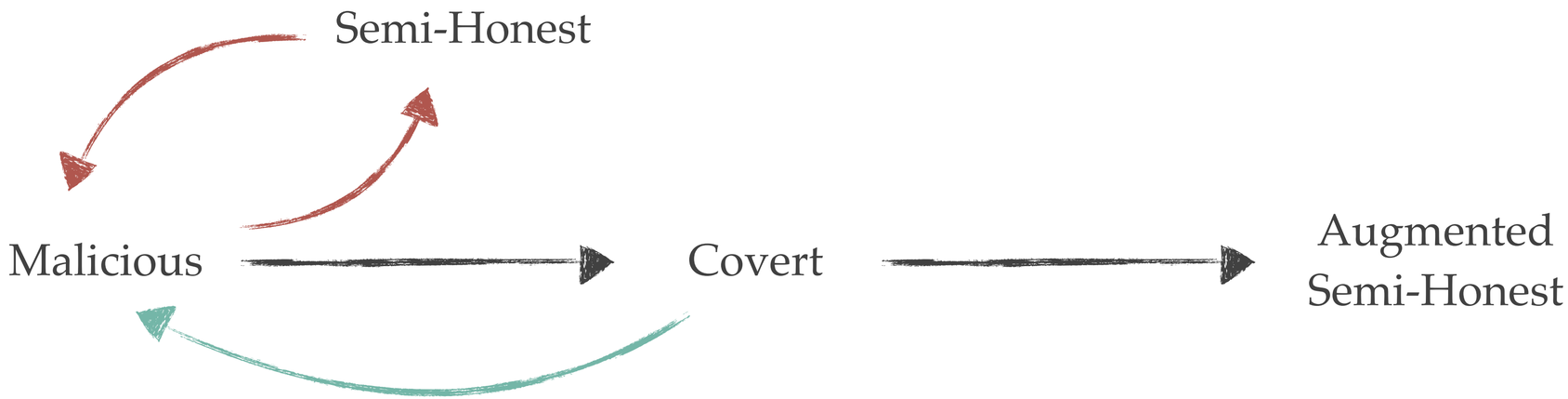
Security Notions

[AL07, HL10]

Semi-honest — Even the party corrupted by the adversary must follow the protocol and use its local input

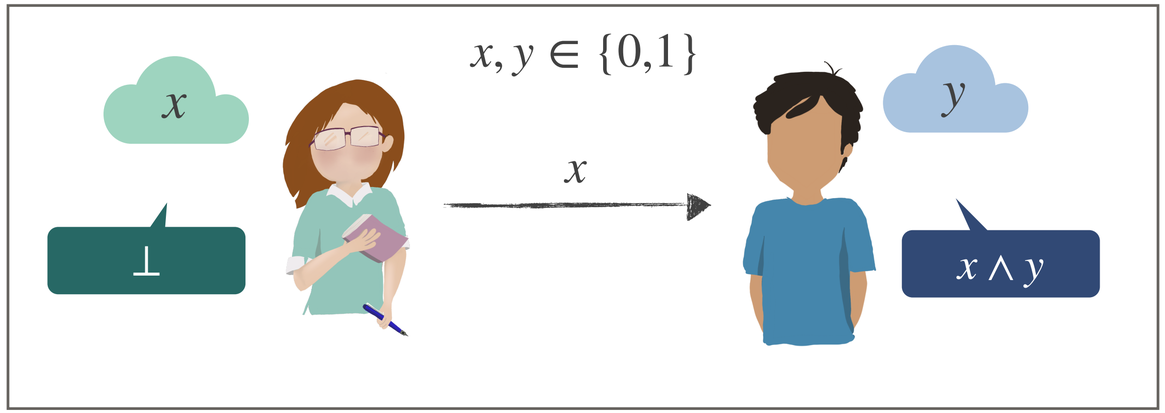
Malicious — The party corrupted by the adversary can deviate from the protocol and change the input it uses

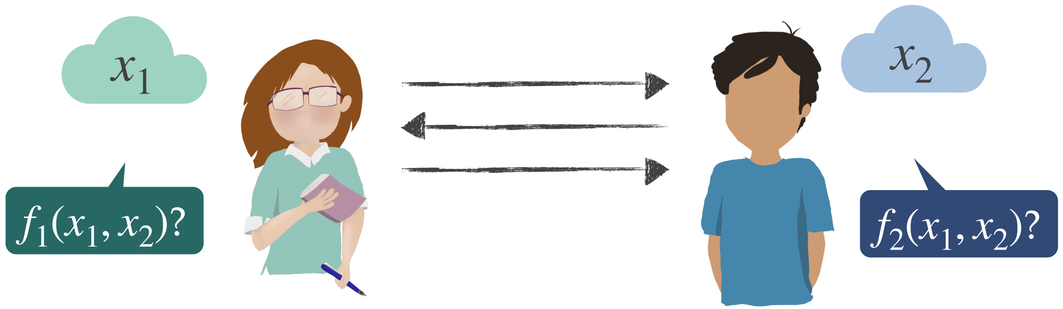
Covert — The party corrupted by the adversary can deviate from the protocol and change the input it uses. Deviations will be detected by the honest party with some given probability ϵ



Security Notions

Achieving security against malicious adversaries is great, but what that really means is complicated!





Suppose $f_1(x_1, x_2) = x_2$ and $f_2(x_1, x_2) = x_1$

Then there really is no privacy being maintained in real terms, yet it satisfies the notion of privacy we considered

Applications using PSI

Ad-conversion

[IKNPSSY17, IKNPRSSSY19]

Google Ads

Set of people ads were shown to

would like to know more about how well the ads are working

Revealing the input set:

- other party can use the set to contact users directly, and stop using ads

amazon

Details of people who bought something

would like to know more about people who bought things after seeing ads

Revealing the input set:

- other party can use the set to advertise to more prolific buyers, and command higher prices

Private Intersection Sum with Cardinality

Private contact discovery

List of people stored as contacts would like to start talking to contacts who are on Signal

Revealing the input set:

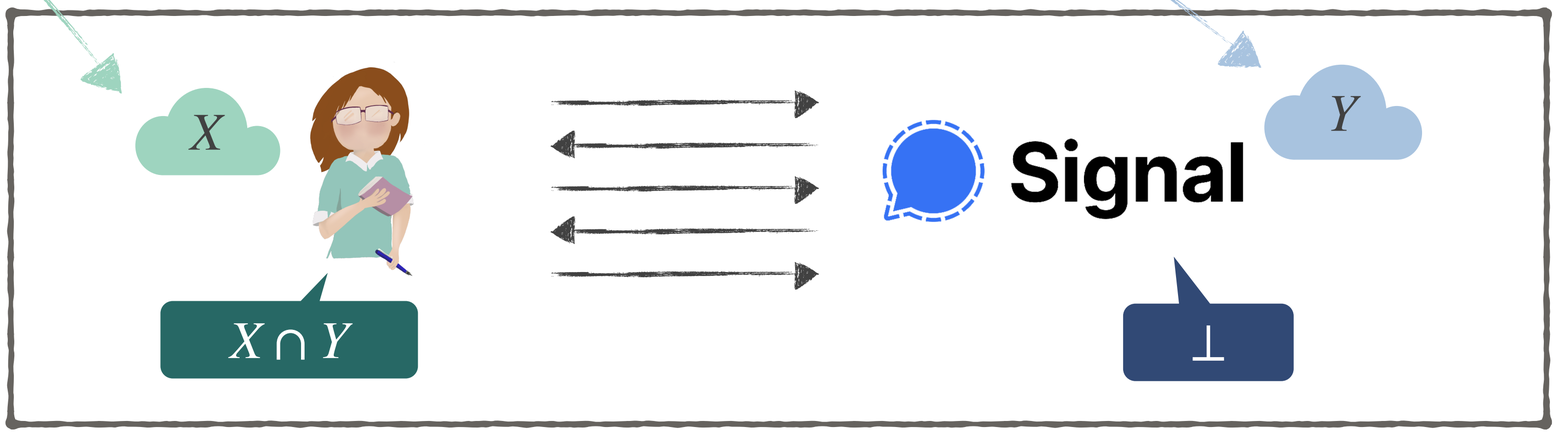
- The service provider could use this information to spam contacts that are not registered
- could be used to build a relationship graph for the user, followed by enhanced tracking and targeting

Signal

List of people who are registered on the app would like to make it easy for new users to start using Signal for conversations

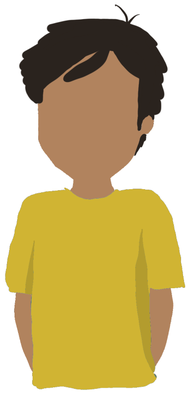
Revealing the input set:

- Loss of reputation as a privacy-conscious app
- Allows rivals to piggy-back on the network



Asymmetric Private Set Intersection

Contact tracing



List of people in close proximity in past 14 days

would like to know if testing / quarantine is needed

Revealing the input set:

- Surveillance concerns — both about who one meets and where one travels

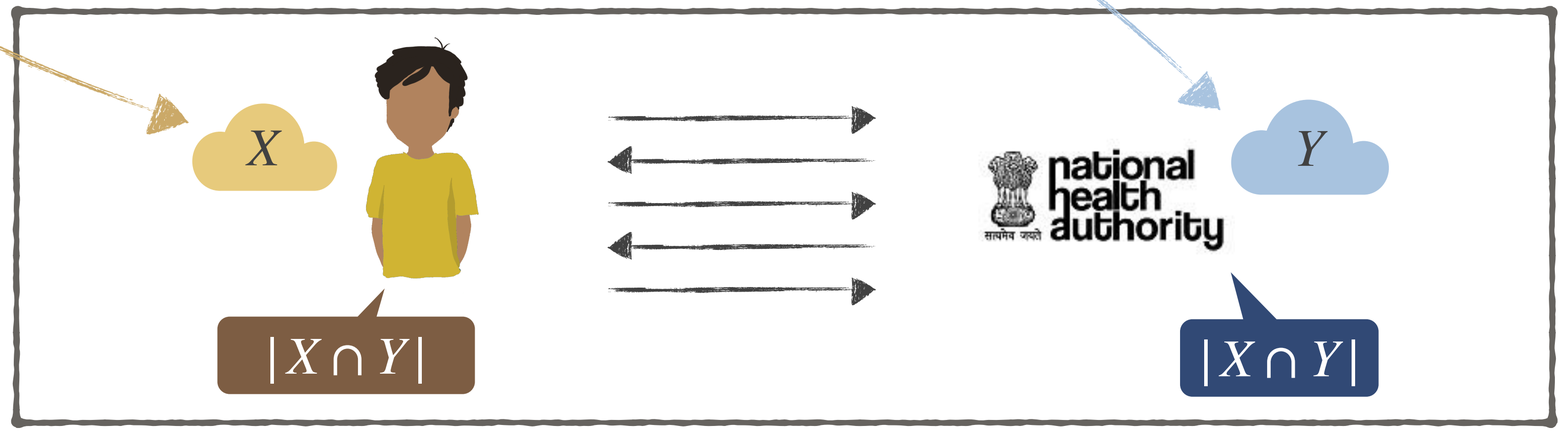
national health authority

List of people who have tested positive

would like to be able to follow-up to see if users have taken the test

Revealing the input set:

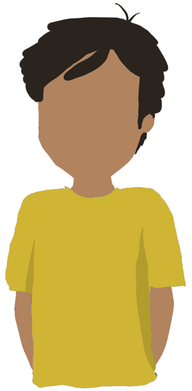
- Stigma associated with being diagnosed positive
- Healthy residents might harm those who have tested positive



Symmetric PSI-CA

Genomics

[BBDGT11]



Fully sequenced genome

Fully sequenced genome

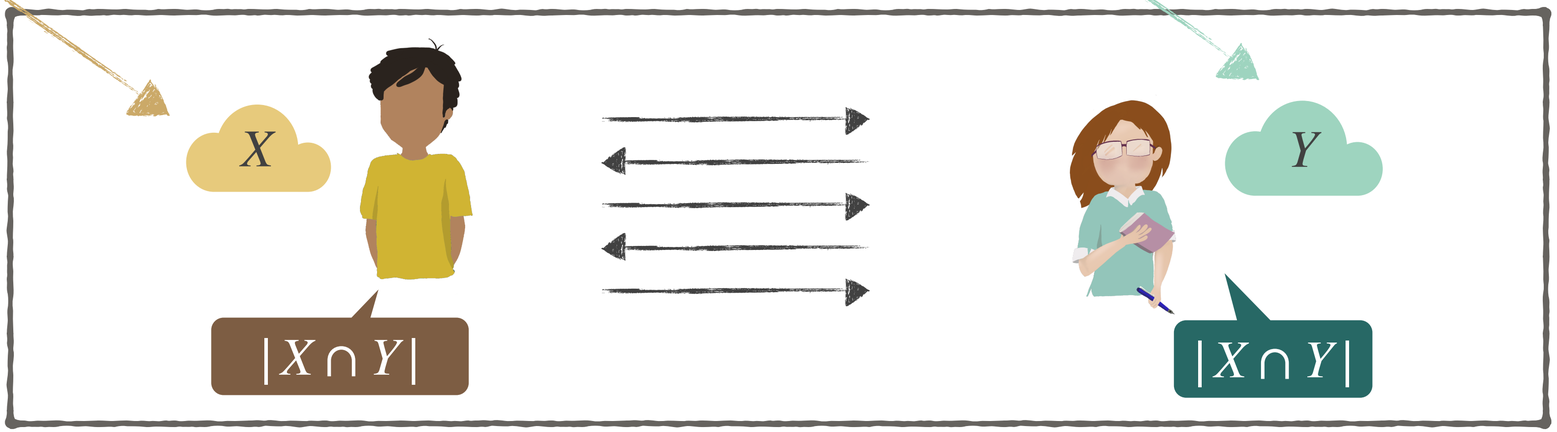
Revealing the input set:

- Disclosure of sensitive personal and medical information

would like to know the result of a paternity test (for example)

Revealing the input set:

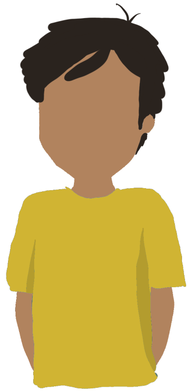
- Disclosure of sensitive personal and medical information

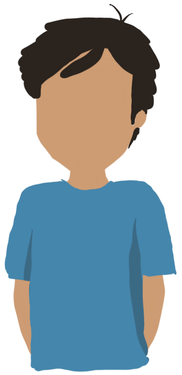


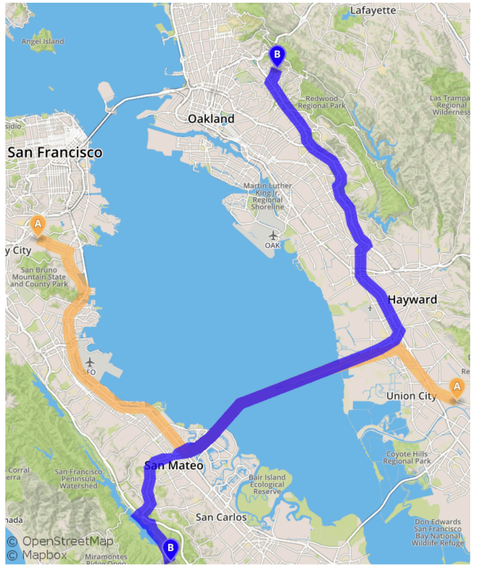
Symmetric PSI-CA

Ride Sharing

[HOS17]







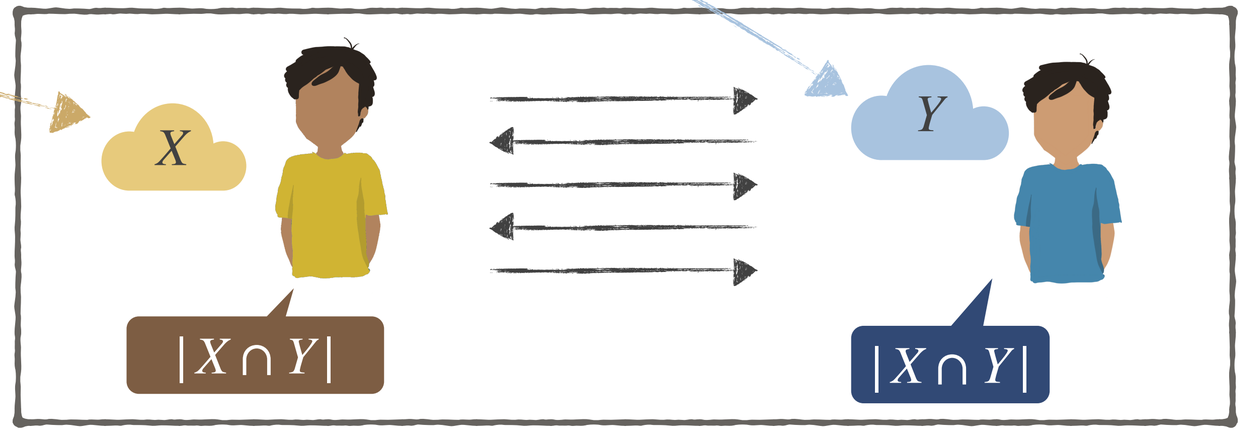
Source, Destination and **Route for trip**

Source, Destination and **Route for trip**

would like to know if they can pool together for their respective trips

Revealing the input set:

- Snooping concerns - neither party would like to reveal their start and end points to the other user



Symmetric PSI-CA

Roadmap

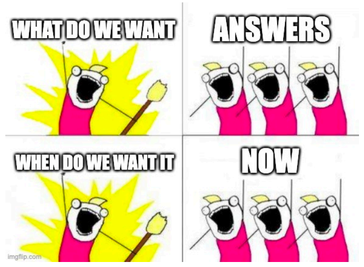
- Multiparty Computation, Private Set Intersection and Variants
- Notions of Security
- Application Descriptions

Security is almost always only considered in the semi-honest model!

- weak
- requires trust in parties

Is restricting to semi-honest security reasonable?

What are the reasons for this restriction?



Are there alternatives to malicious / covert security?

Categorizing the participants

[Kam20]

People

have access to devices like smartphones and computers

have small to medium-sized data sets

Meta Google NETFLIX amazon Uber Signal

Power

have large servers with significant computational and storage capacity

data sets can be pretty large

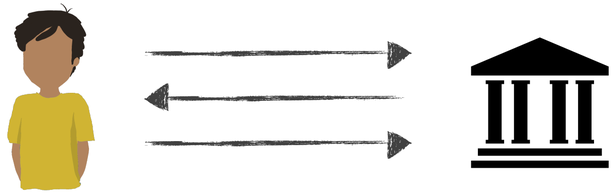
Categorizing the application protocols

People - People

Genomics

Ride Sharing

Proximity Testing



People - Power

Private Contact Discovery

Contact Tracing

Power - Power

Ad-conversion Systems

People

Most users would just download the app and use it as is → then semi-honest is sufficient!

However, some users might act maliciously → maybe by personally tweaking the source code

→ or by downloading the app from an untrustworthy source

[CD16, PST16]

Creates trusted execution environments

code runs in a secure enclave, cannot be tampered with

provides an attestation to the execution

side channels were not considered in Intel's threat model — developers are required to account for them

improper partitioning into trusted and untrusted sections of code can cause leakage of information

The Foreshadow Attack
[VMWGKPSWYS18]

Malicious security ✓

BUT… inefficient!

Covert security + Penalties

Power

Have access to a larger pool of resources, and also have more **power**

Organizations ← Laws and the State

Organizations ← Public opinion

Terms of Service

State ← strong separation of powers

ultimate coercing power

Strong incentive to gather data on individuals → surveillance

Strong incentive to gather data on individuals → profits

Strong incentive to gather data on individuals → improvement of services

---

Malicious security ✓

Code Audits → work mostly in the power-power protocol setting

[IKNPRSSSY19]

Providing incorrect inputs

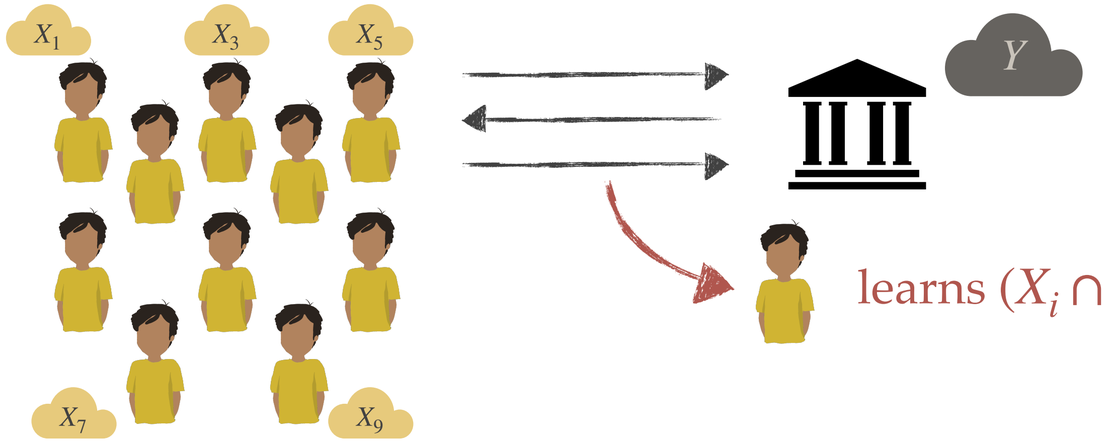
This is a limitation of the problem itself

Incentives for providing incorrect inputs

- subvert correctness — make the output different
- subvert privacy — learn more about the other party's input

[HWSDS21]

Enumeration attacks when the domain the sets are from is small enough



Mitigation : Rate-limiting

Providing incorrect inputs

This is a limitation of the problem itself

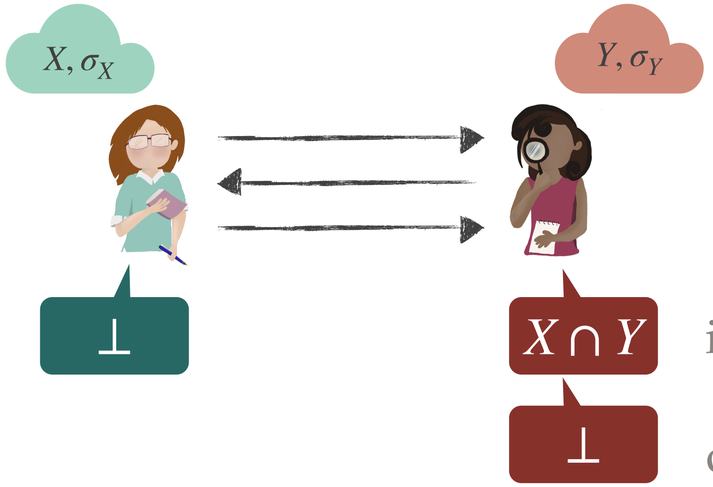
Incentives for providing incorrect inputs → subvert correctness — make the output different

→ subvert privacy — learn more about the other party's input

[HWSDS21]

Enumeration attacks when the domain the sets are from is small enough **Mitigation : Rate-limiting**

Authorized PSI [DKT10]



if σ_X and σ_Y verify

otherwise

Can be done for other variants too

Requires trust in a Certificate Authority!

Then why not just have a trusted party do the computation?

Trust — only that signatures are correctly generated

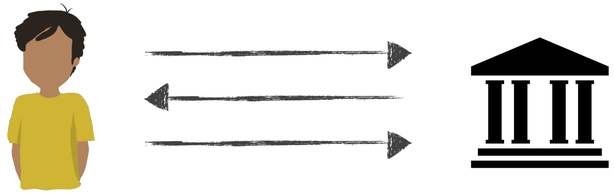
Categorizing the application protocols

People - People

Genomics

Ride Sharing

Proximity Testing



People - Power

Private Contact Discovery

Contact Tracing

Power - Power

Ad-conversion Systems

People - People

Genomics

Ride Sharing

Proximity Testing

- Incentive to subvert correctness
- Covert security to catch cheating — legal penalties?
- Authorized PSI could work for incorrect inputs

People - People

Genomics

Ride Sharing

Proximity Testing

Genomics

- Incentive to subvert correctness
- Covert security to catch cheating — legal penalties?
- Authorized PSI could work for incorrect inputs

- Incentive to subvert privacy
- Changing inputs — suboptimal results for ride share.
- Trusted execution would be useful
- Covert security — how to penalize?
- Malicious security — ideal, but efficiency?

People - People

Genomics

Ride Sharing

Proximity Testing

Genomics

- Incentive to subvert correctness
- Covert security to catch cheating — legal penalties?
- Authorized PSI could work for incorrect inputs

Ride Sharing

- Incentive to subvert privacy
- Changing inputs — suboptimal results for ride share.
- Trusted execution would be useful
- Covert security — how to penalize?
- Malicious security — ideal, but efficiency?

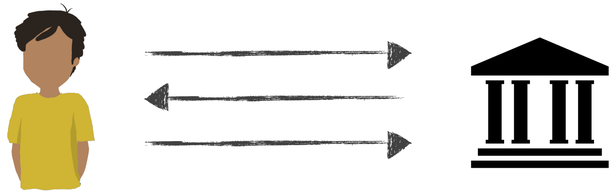
- Incentives to subvert privacy as well as correctness
- Trusted execution would be useful
- Covert security — how to penalize?
- Malicious security — ideal, but efficiency?

People - People

Genomics

Ride Sharing

Proximity Testing



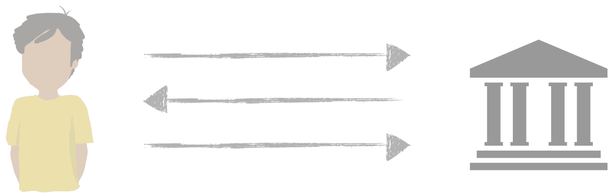
People - Power

Private Contact Discovery

Contact Tracing

Power - Power

Ad-conversion Systems

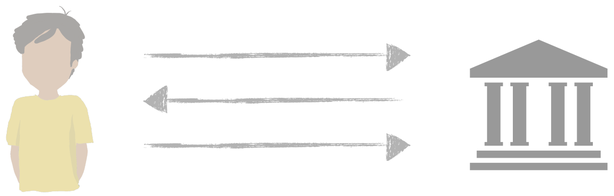


People - Power

Private Contact Discovery

Contact Tracing

- Incentive to subvert privacy
- Malicious or covert security could assure users their data is safe, and also guard from coercion by the State
- Trusted execution could work
- Enumeration attacks



People - Power

Private Contact Discovery

Contact Tracing

Private Contact Discovery

- Incentive to subvert privacy
- Malicious or covert security could assure users their data is safe, and also guard from coercion by the State
- Trusted execution could work
- Enumeration attacks

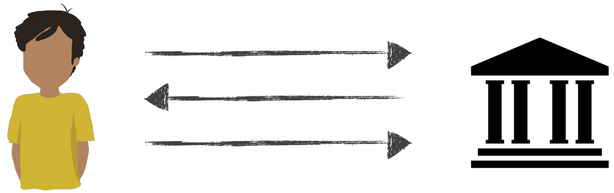
- Incentives to subvert privacy as well as correctness
- Malicious security would give users confidence the government is not snooping
- Covert — how to penalize the State?

People - People

Genomics

Ride Sharing

Proximity Testing



People - Power

Private Contact Discovery

Contact Tracing

Power - Power

Ad-conversion Systems

Power - Power

Ad-conversion Systems

- Incentive to subvert privacy
- Code audits
- Why not covert, or malicious security?
- How to prevent changing the input?

The [CM20] Protocol

Sender

Set X

$s \leftarrow \{0,1\}^w$

one bit at a time

1-out-of-2 OT

$\mathbf{OT}((A_i, B_i), s[i])$

one column at a time

C

k

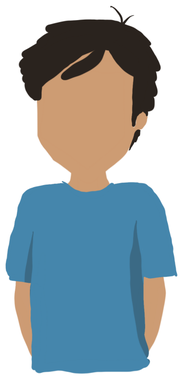
$k \leftarrow \{0,1\}^\lambda$

$D \leftarrow 1^{m \times w}$

For $y \in Y$:
  $v \leftarrow F_k(H_1(y))$
  For $I \in [w]$:
    $D_i[v[i]] \leftarrow 0$

$A \leftarrow \{0,1\}^{m \times w}$

$B \leftarrow A \oplus D$



Receiver

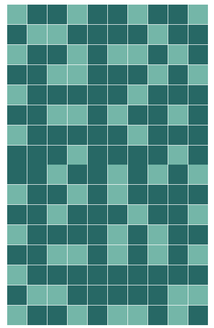
Set Y

The [CM20] Protocol

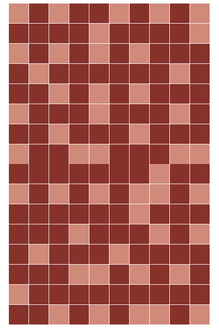
Sender

$s \leftarrow \{0,1\}^w$

k



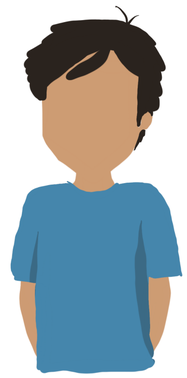
C



A

$k \leftarrow \{0,1\}^\lambda$

$A \leftarrow \{0,1\}^{m \times w}$



Receiver

Set X

Set Y

For $x \in X$:

$v \leftarrow F_k(H_1(x))$

$Z \leftarrow Z \cup \left\{ H_2\left(C_1[v[1]] \| \ldots C_w[v[w]]\right) \right\}$

Z

For $y \in Y$:

$v \leftarrow F_k(H_1(y))$

$h \leftarrow H_2\left(A_1[v[1]] \| \ldots A_w[v[w]]\right)$

If $h \in Z$ then

Return I

Attacking the [CM20] Protocol

Sender

Set X

$s \leftarrow \{0,1\}^w$

one bit at a time

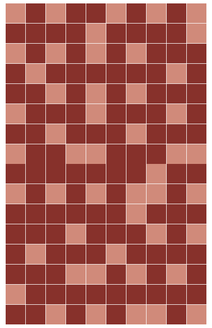
$D \leftarrow 0^{m \times w}$

$A \leftarrow \{0,1\}^{m \times w}$

$B \leftarrow A \oplus D$

Receiver

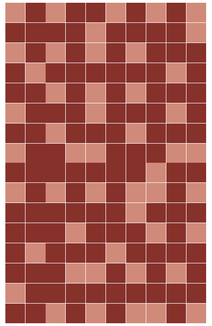
Set Y



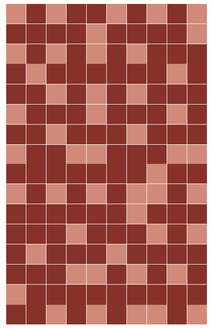
one column at a time

1-out-of-2 OT

$\mathbf{OT}((A_i, B_i), s[i])$



C



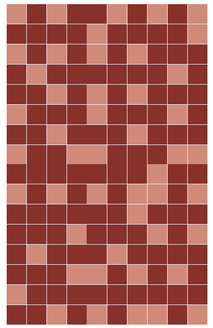
k

Attacking the [CM20] Protocol

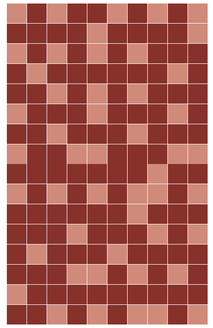
Sender

$s \leftarrow \{0,1\}^w$

k



C



A

$k \leftarrow \{0,1\}^\lambda$

$A \leftarrow \{0,1\}^{m \times w}$

Receiver

Set X

Set Y

For $x \in X$:

$v \leftarrow F_k(H_1(x))$

$Z \leftarrow Z \cup \left\{ H_2\left(C_1[v[1]] \| \ldots C_w[v[w]]\right) \right\}$

Receiver can extract all info about X from Z

Z

For $y \in Y$:

$v \leftarrow F_k(H_1(y))$

$h \leftarrow H_2\left(A_1[v[1]] \| \ldots A_w[v[w]]\right)$

If $h \in Z$ then

Return I

Conclusions

- Guarantees of privacy and security are essential for interactions between parties that involve the use of sensitive information.
- PSI protocols have the potential to be used for a number of different applications
  - Semi-honest security is insufficient
  - Malicious security is inefficient
- Trusted execution environments could allow semi-honest protocols to run as per the specifications
  - Currently no provable security guarantees

- The literature should be more objective about the use of semi-honest security in applications