Introduction and Course Overview

CS/ECE 407

Today's objectives

Understand the course objectives

Describe setting for MPC

A puzzle

Course Website

https://courses.grainger.illinois.edu/cs598dh/sp2024/

What is this course about?

Use cryptography to run computer programs on “encrypted” data.

Study cryptographic protocols and their security

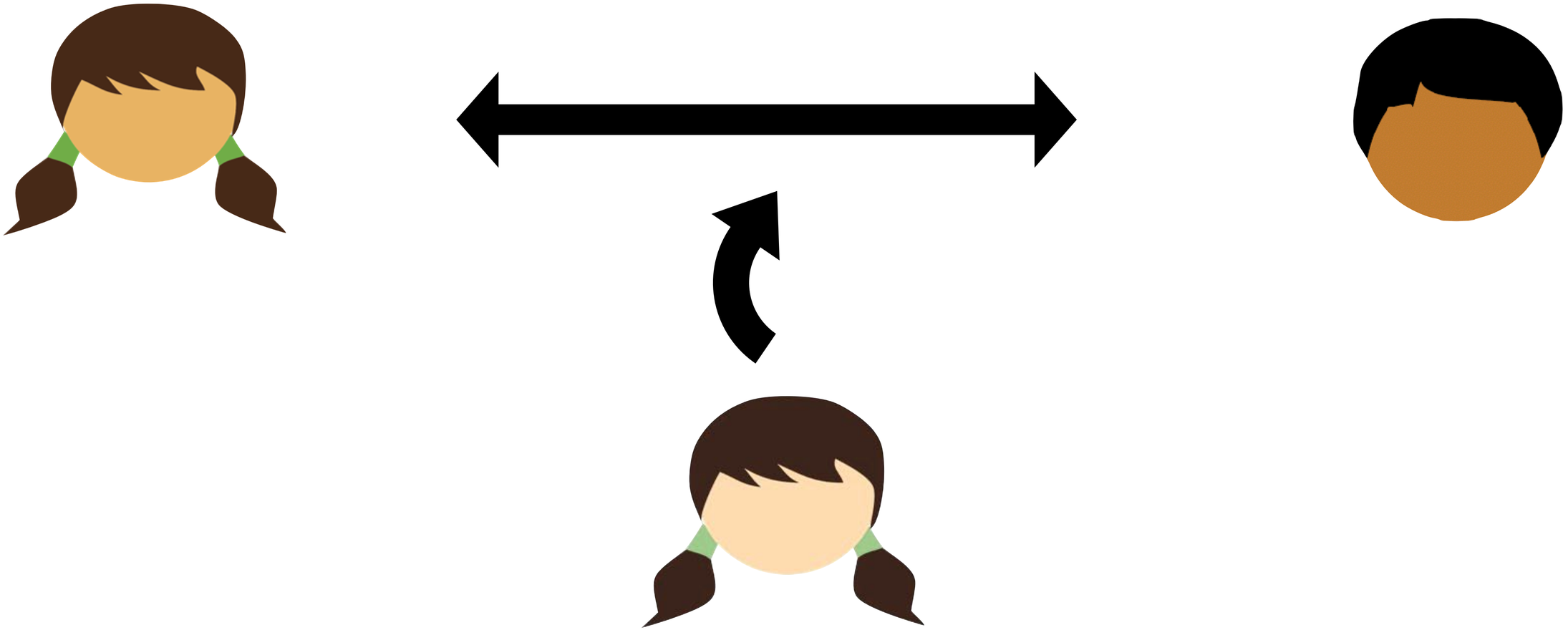
What is this course about?

Use cryptography to run computer programs on “encrypted” data.

By doing so, we can solve problems while keeping the underlying data private.

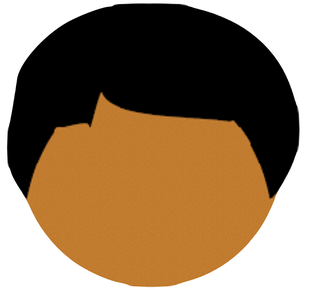
What is this course *not* about?

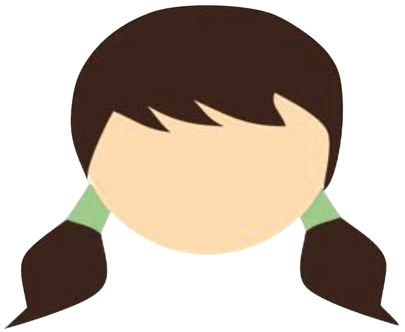
classic cryptography setting



What is this course *not* about?

classic cryptography setting





Privacy
Authenticity

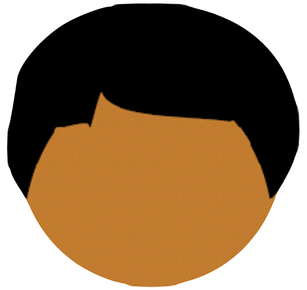
Privacy (informal)

Eve cannot understand
the conversation between
Alice and Bob

Authenticity (informal)

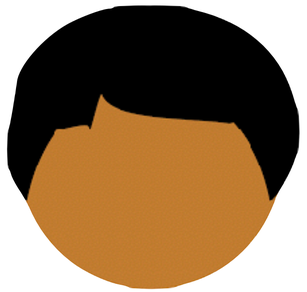
Eve cannot pretend to be Alice or Bob

This Class



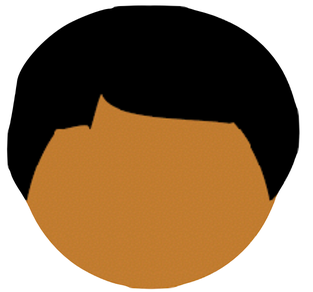
Privacy
Authenticity

This Class



Privacy
Authenticity

This Class



willingness to collaborate

≠

complete trust

You make a new account online

You make a new account online

Some time passes…

You make a new account online

Some time passes…

Google Chrome tells you that your username/password was leaked

You 

“CS598DH // Password123”

…

“SuzieQ // Password1”

“DonnyK // Secret”

“AlanT // MyDogIsGreat”

“CS598DH // Password123”

…

You

“CS598DH // Password123”

…
“SuzieQ // Password1”
“DonnyK // Secret”
“AlanT // MyDogIsGreat”
“CS598DH // Password123”
…

Option 1

You

“CS598DH // Password123”

…
“SuzieQ // Password1”
“DonnyK // Secret”
“AlanT // MyDogIsGreat”
“CS598DH // Password123”
…

Option 2

You

…

“SuzieQ // Password1”

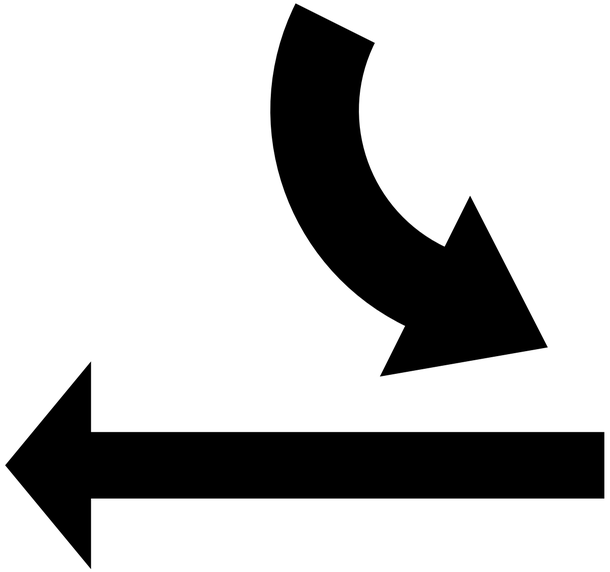
“DonnyK // Secret”

“AlanT // MyDogIsGreat”

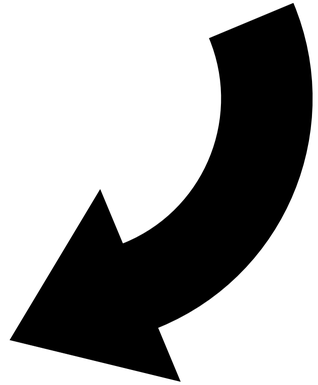
“CS598DH // Password123”

…

“CS598DH // Password123”



Crypto Magic



Google Password Manager

Protecting accounts from credential stuffing with password breach alerting

Kurt Thomas[*] *Jennifer Pullman*[*] *Kevin Yeo*[†] *Ananth Raghunathan*[†]
Patrick Gage Kelley[*] *Luca Invernizzi*[†] *Borbala Benko*[*] *Tadek Pietraszek*[*]
Sarvar Patel[*] *Dan Boneh*[◇] *Elie Bursztein*[*]

Google[*] *Stanford*[◇]

Abstract

Protecting accounts from credential stuffing attacks remains burdensome due to an asymmetry of knowledge: attackers have wide-scale access to billions of stolen usernames and passwords, while users and identity providers remain in the dark as to which accounts require remediation. In this paper, we propose a privacy-preserving protocol whereby a client can query a centralized breach repository to determine whether a specific username and password combination is publicly exposed, but without revealing the information queried. Here, a client can be an end user, a password manager, or an identity provider. To demonstrate the feasibility of our protocol, we implement a cloud service that mediates access to over 4 billion credentials found in breaches and a Chrome extension serving as an initial client. Based on anonymous telemetry from nearly 670,000 users and 21 million logins, we find that 1.5% of logins on the web involve breached credentials. By alerting users to this breach status, 26% of our warnings result in users migrating to a new password, at least as strong as the original. Our study illustrates how secure, democratized access to password breach alerting can help mitigate one dimension of account hijacking.

1 Introduction

The wide-spread availability of usernames and passwords exposed by data breaches has trivialized criminal access to billions of accounts. In the last two years alone, breach compilations like Antipublic (450 million credentials), Exploit.in (600 million credentials), and Collection 1-5 (2.2 billion credentials) have steadily grown as their creators aggregated material shared on underground forums [24,28]. Despite the public nature of this data, it remains no less potent. Previous studies have shown that 6.9% of breached credentials remain valid due to reuse, even multiple years after their initial exposure [54]. Absent defense in depth techniques that expand authentication to include a user's location and device details [15,20], hijackers need only conduct a credential stuffing attack—attempting to log in with every breached credential—to isolate vulnerable accounts.

While users (or identity providers) can mitigate this hijacking risk by resetting an account's password, in practice, discovering which accounts require attention remains a critical barrier. This has given rise to breach alerting services like HaveIBeenPwned and PasswordPing that actively source breached credentials to notify affected users [29, 46]. At present, these services make a variety of tradeoffs spanning user privacy, accuracy, and the risks involved with sharing ostensibly private account details through unauthenticated public channels. One consequence of these tradeoffs is that users may receive inaccurate remediation advice due to false positives. For example, both Firefox and LastPass check the breach status of usernames to encourage password resetting [16,45], but they lack context for whether the user's password was actually exposed for a specific site or whether it was previously reset. Equally problematic, other schemes implicitly trust breach alerting services to properly handle plaintext usernames and passwords provided as part of a lookup. This makes breach alerting services a liability in the event they become compromised (or turn out to be adversarial).

In this paper, we present the design, implementation, and deployment of a new privacy-preserving protocol that allows a client to learn whether their username and password appears in a breach without revealing the information queried. Our protocol offers two main advantages compared to existing schemes. First, our design takes into account the threat of both an adversarial client (e.g., an attacker attempting to steal usernames and passwords from our service) and an adversarial server (e.g., an attacker harvesting usernames and passwords sent to the service). We address these risks using a combination of computationally expensive hashing, k-anonymity, and private set intersection. Second, these privacy requirements allow us to check a client's exact username and password against a database of breached credentials (versus only usernames, or only passwords currently), thus reducing false positives that lead to warning fatigue.

To demonstrate the feasibility of our protocol, we publicly

Better password protections in Chrome - How it works

December 10, 2019

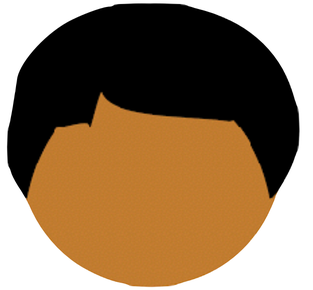
Posted by Patrick Nepper, Kiran C. Nair, Vasilii Sukhanov and Varun Khaneja, Chrome Team

Today, we announced better password protections in Chrome, gradually rolling out with release M79. Here are the details of how they work.

- In order to determine if your username and password appears in any breach, we use a technique called private set intersection with blinding that involves multiple layers of encryption. This allows us to compare your encrypted username and password with all of the encrypted breached usernames and passwords, without revealing your username and password, or revealing any information about any other users' usernames and passwords. In order to make this computation more efficient, Chrome sends a 3-byte SHA256 hash prefix of your username to reduce the scale of the data joined from 4 billion records down to 250 records, while still ensuring your username remains anonymous.

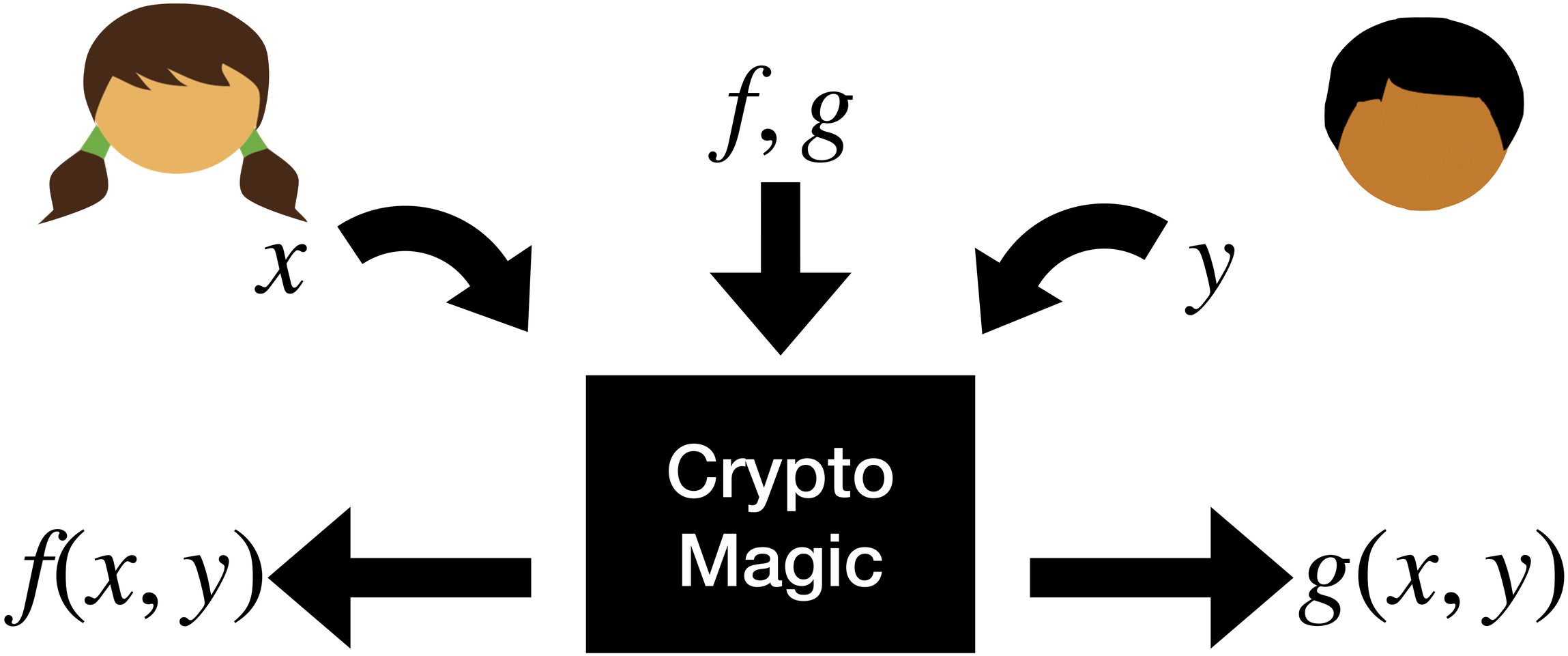
x

f, g

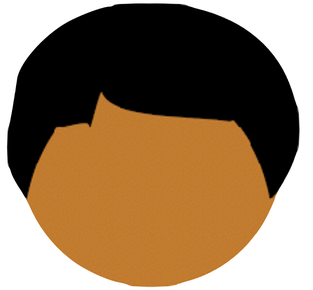


y

Crypto
Magic


f, g
x
y
Crypto
Magic
$f(x, y)$
$g(x, y)$

f, g



x

y

Crypto Magic

$f(x, y)$

$g(x, y)$

Privacy

Authenticity

Privacy (informal)

Alice learns nothing about
Bob's input
(except for what is implied
by the function output)

Authenticity (informal)

Alice cannot force Bob to output something different than $g(x, y)$

This Class

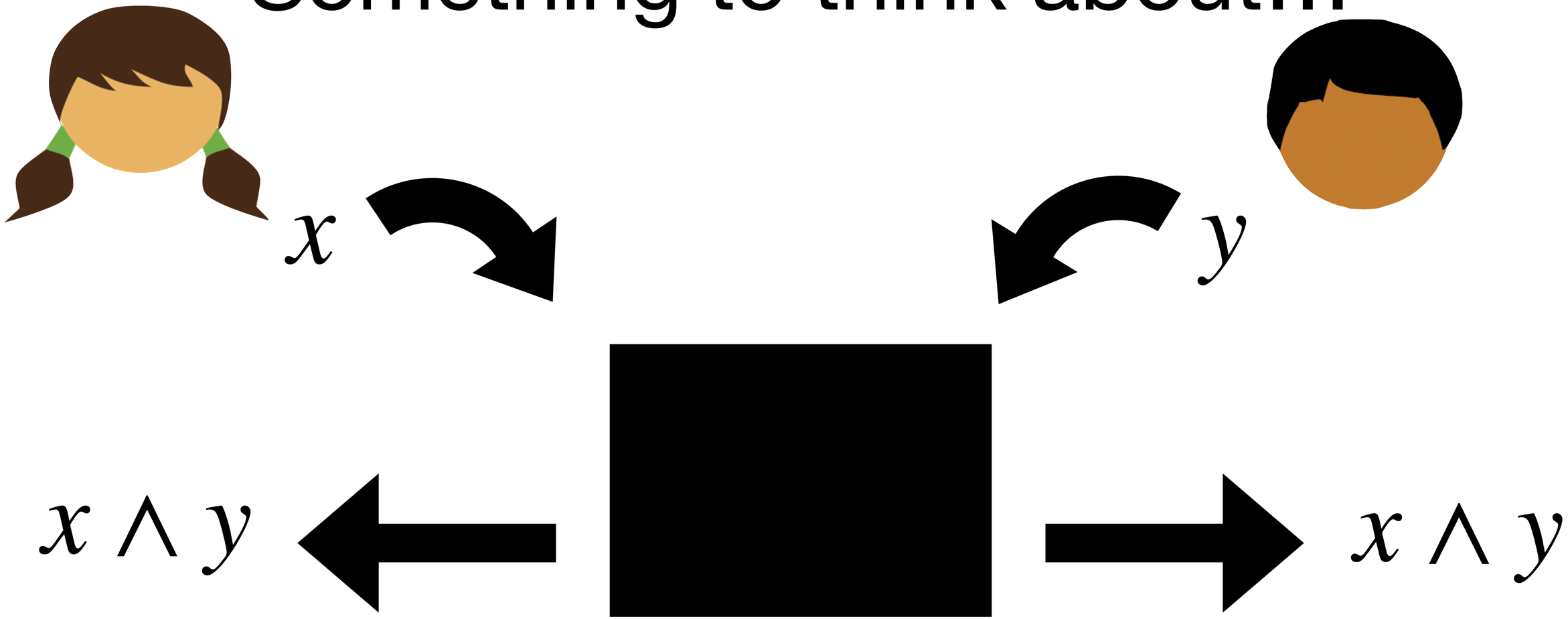
What is privacy?

What is authenticity?

What are the basic tools of this magic?

What is the cutting edge of this magic?

Something to think about…



Something to think about…