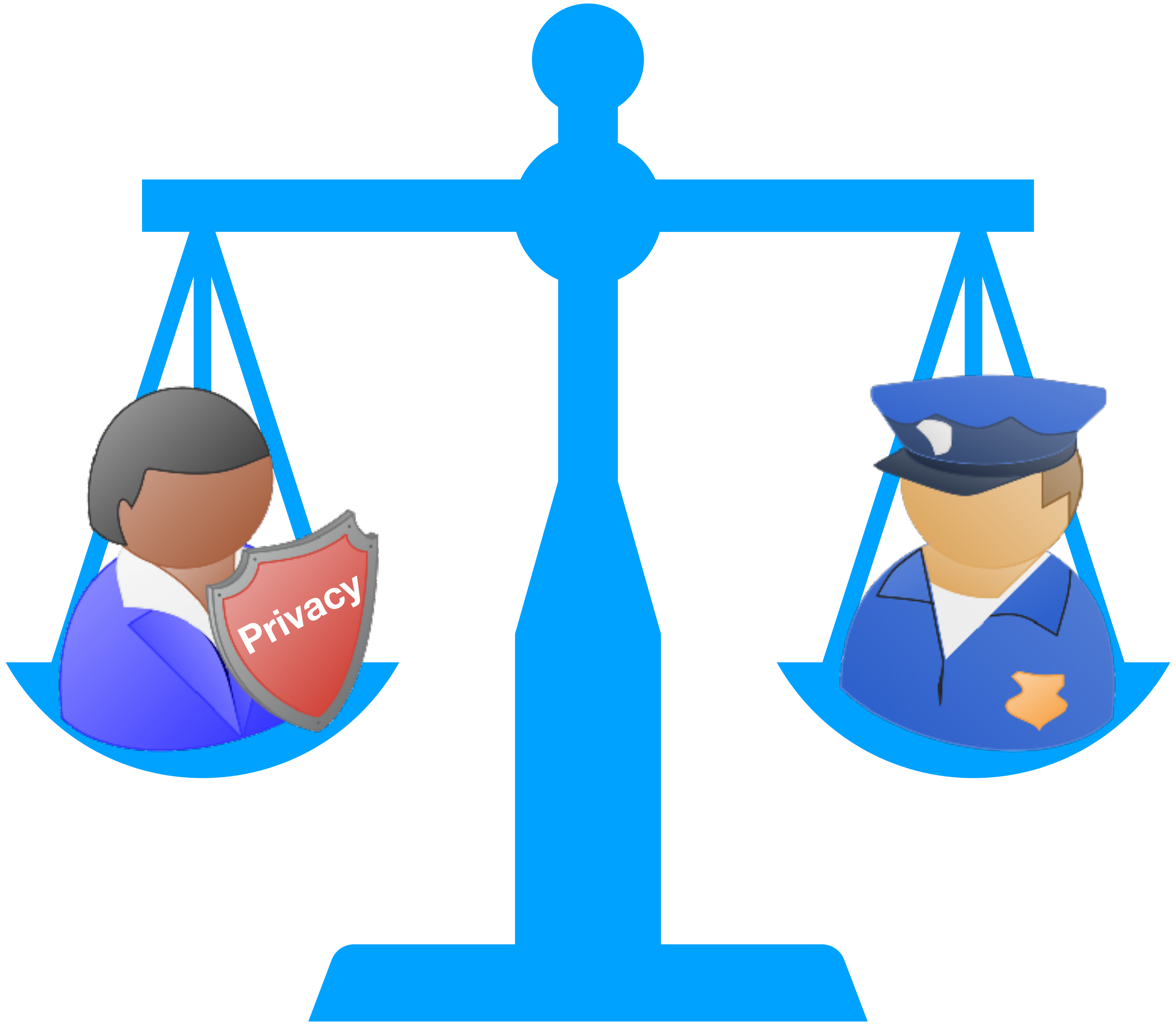
End-to-End Secure Messaging with Traceability Only for *Illegal* Content

James Bartusek, Sanjam Garg, Abhishek Jain, and Guru Vamsi Policharla

The debate over encryption

Privacy is a fundamental right!



E2EE interferes with the prosecution of criminals!

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

If you want to learn more, RWC talks are good starting points:

- An evaluation of the risks of client-side scanning [GTSST22]
- Reactionary Authoritarianism, Encryption, and You! [Portnoy23]

No good way to implement backdoors

- All proposed systems are susceptible to abuse
- Surveillance and censorship is a real threat
- Assurances by companies is not sufficient!

Apple limits AirDrop on iPhones in China after filesharing feature was used by protesters

Censorship, Surveillance and Profits: A Hard Bargain for Apple in China

Apple built the world's most valuable business on top of China. Now it has to answer to the Chinese government.

And in its data centers, Apple's compromises have made it nearly impossible for the company to stop the Chinese government from gaining access to the emails, photos, documents, contacts and locations of millions of Chinese residents, according to the security experts and Apple engineers.

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

Negative impacts cannot be ignored

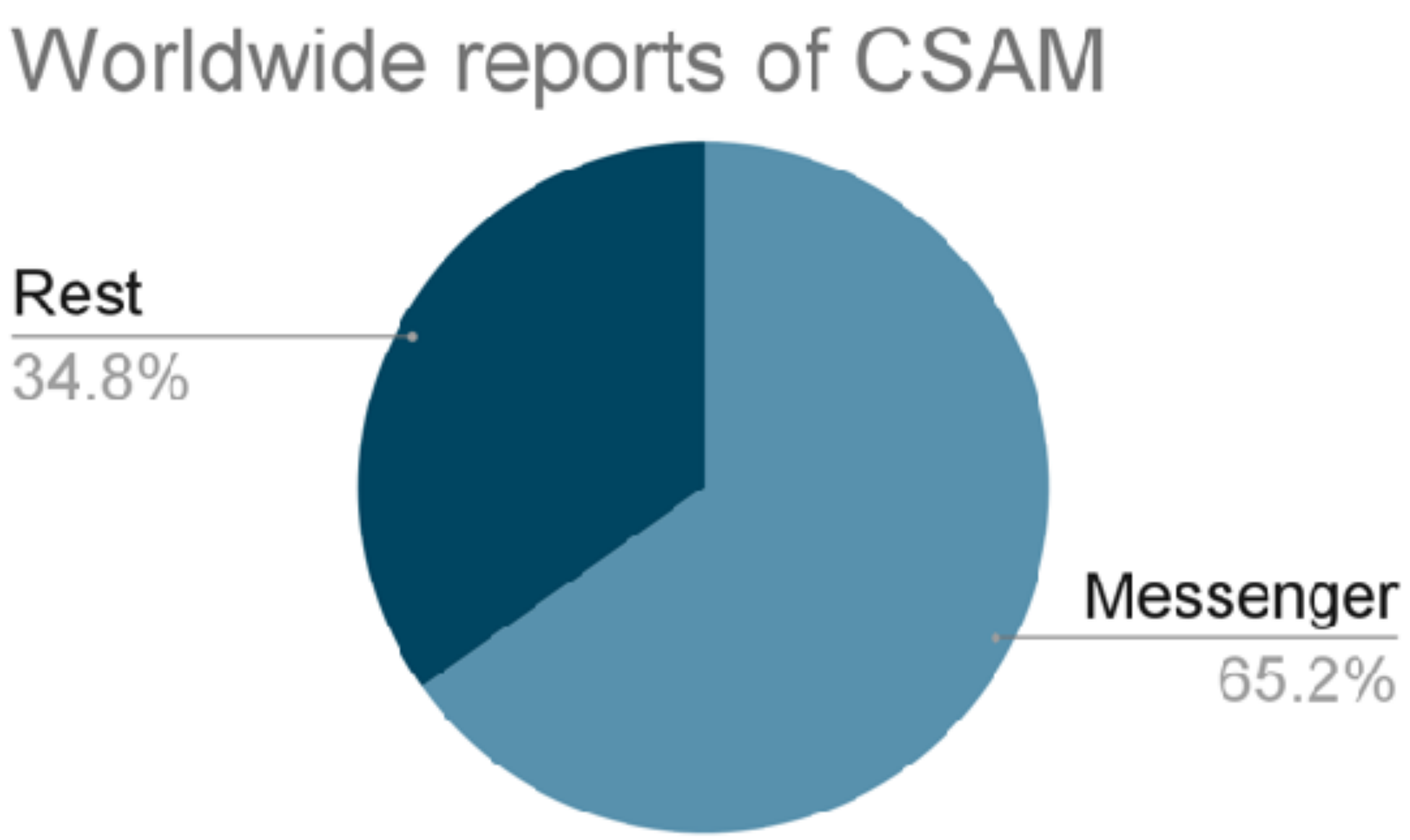
In response to Facebook deploying E2EE in messenger

FOR IMMEDIATE RELEASE — Sunday, October 11, 2020

International Statement: End-To-End Encryption and Public Safety

We, the undersigned, support strong encryption, which plays a crucial role in protecting personal data, privacy, intellectual property, trade secrets and cyber security. It also serves a vital purpose in repressive states to protect journalists, human rights defenders and other vulnerable people, as stated in the 2017 resolution of the UN Human Rights Council[1]. Encryption is an existential anchor of trust in the digital world and we do not support counter-productive and dangerous approaches that would materially weaken or limit security systems.

- Embed the safety of the public in system designs, thereby enabling companies to act against illegal content and activity effectively with no reduction to safety, and facilitating the investigation and prosecution of offences and safeguarding the vulnerable;
- Enable law enforcement access to content in a readable and usable format where an authorisation is lawfully issued, is necessary and proportionate, and is subject to strong safeguards and oversight; and
- Engage in consultation with governments and other stakeholders to facilitate legal access in a way that is substantive and genuinely influences design decisions.



SIGNATORIES

Rt Hon Priti Patel MP, United Kingdom Secretary of State for the

William P. Barr, Attorney General of the United States

The Hon Peter Dutton MP, Australian Minister for Home Affairs

Hon Andrew Little MP, Minister of Justice, Minister Responsible

The Honourable Bill Blair, Minister of Public Safety and Emergen

India

Japan

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

Negative impacts cannot be ignored

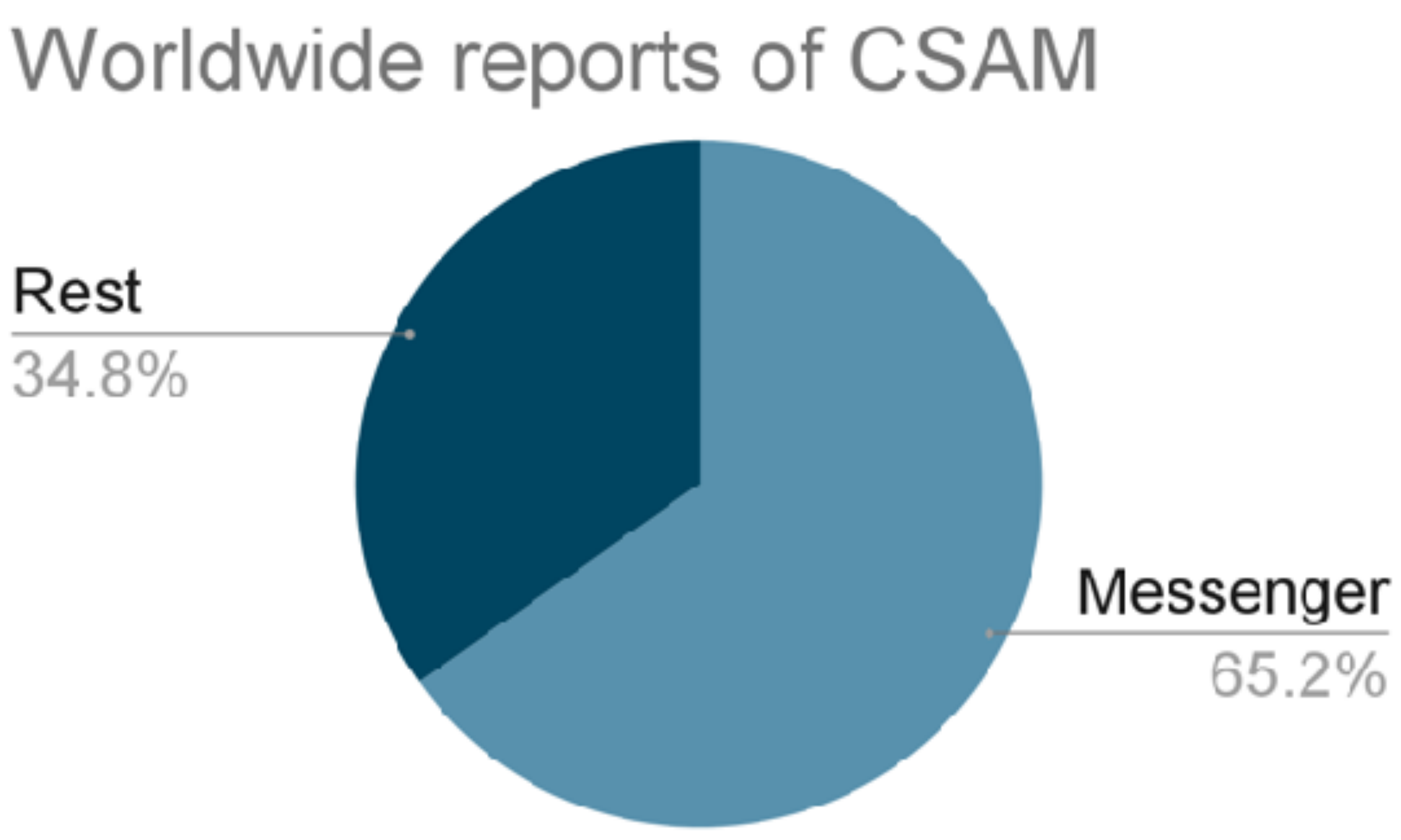
In response to Facebook deploying E2EE in messenger

FOR IMMEDIATE RELEASE   Sunday, October 11, 2020

International Statement: End-To-End Encryption and Public Safety

We, the undersigned, support strong encryption, which plays a crucial role in protecting personal data, privacy, intellectual property, trade secrets and cyber security. It also serves a vital purpose in repressive states to protect journalists, human rights defenders and other vulnerable people, as stated in the 2017 resolution of the UN Human Rights Council[1]. Encryption is an existential anchor of trust in the digital world and we do not support counter-productive and dangerous approaches that would materially weaken or limit security systems.

- Embed the safety of the public in system designs, thereby enabling companies to act against illegal content and activity effectively with no reduction to safety, and facilitating the investigation and prosecution of offences and safeguarding the vulnerable;
- Enable law enforcement access to content in a readable and usable format where an authorisation is lawfully issued, is necessary and proportionate, and is subject to strong safeguards and oversight; and
- Engage in consultation with governments and other stakeholders to facilitate legal access in a way that is substantive and genuinely influences design decisions.



SIGNATORIES

Rt Hon Priti Patel MP, United Kingdom Secretary of State for the

William P. Barr, Attorney General of the United States

The Hon Peter Dutton MP, Australian Minister for Home Affairs

Hon Andrew Little MP, Minister of Justice, Minister Responsible

The Honourable Bill Blair, Minister of Public Safety and Emergen

India

Japan

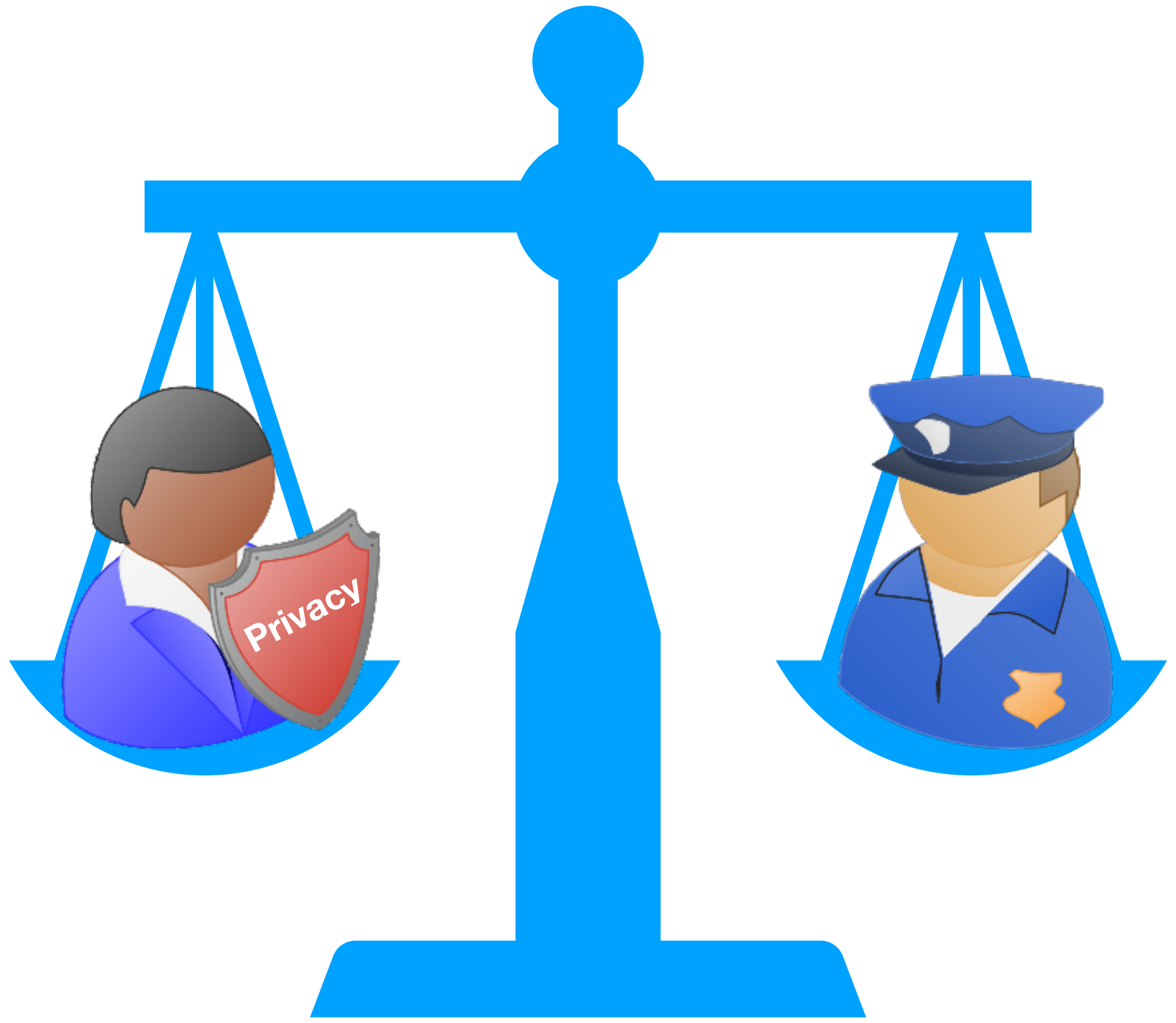
Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

The debate over encryption

Can we find a middle ground making both sides happy?

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

The debate over encryption

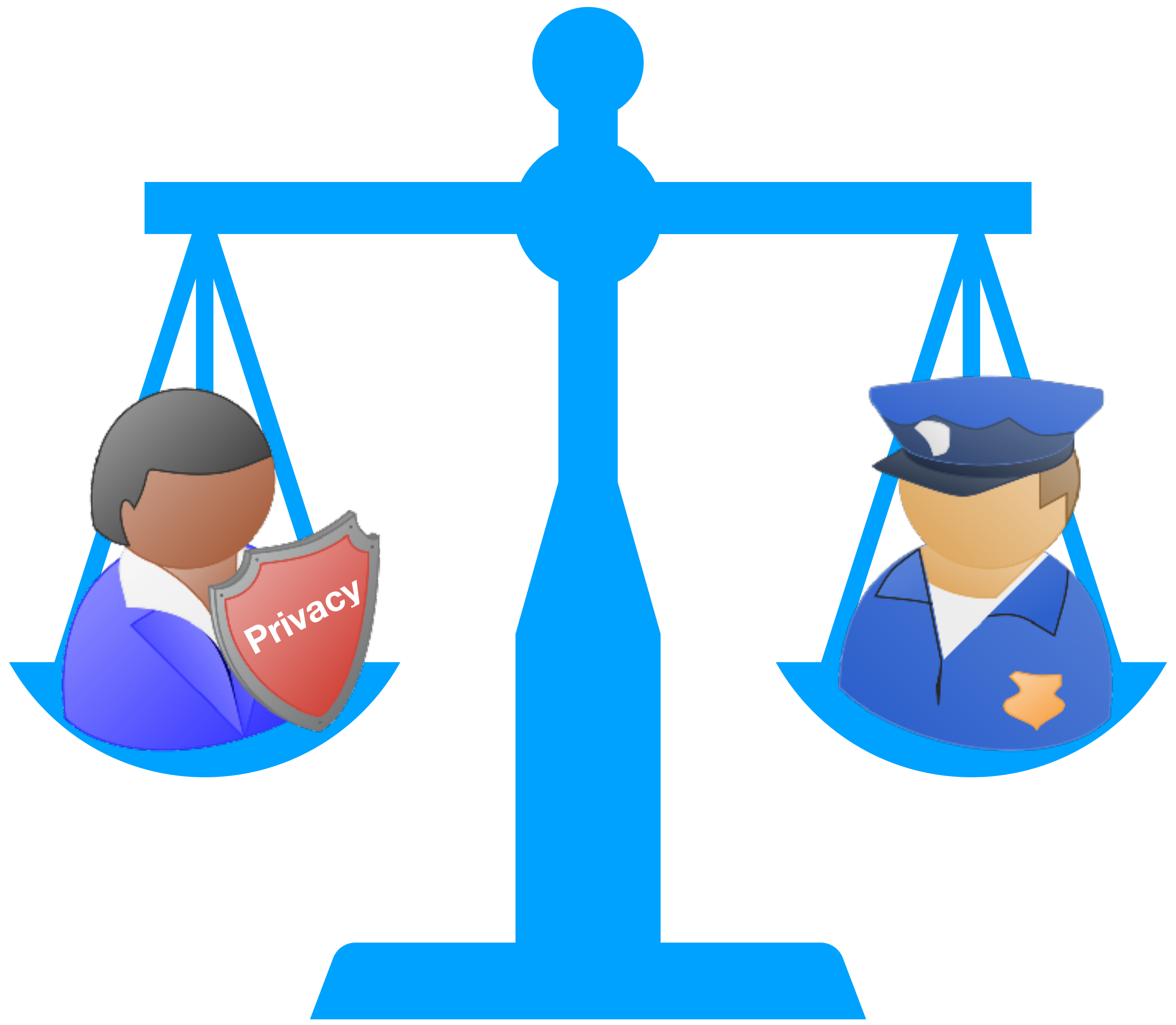


Can we find a middle ground making both sides happy?

Identify bad actors while preserving privacy of honest users

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

The debate over encryption



Can we find a middle ground making both sides happy?

Identify bad actors while preserving privacy of honest users

<u>Disclaimer: We do not think any proposal is safe for deployment yet</u>

Disclaimer: This is a much more nuanced debate than I have the time or expertise to talk about

Content moderation today

Moderation without E2EE

- Server given a database — <u>hashes</u> of “illegal” images

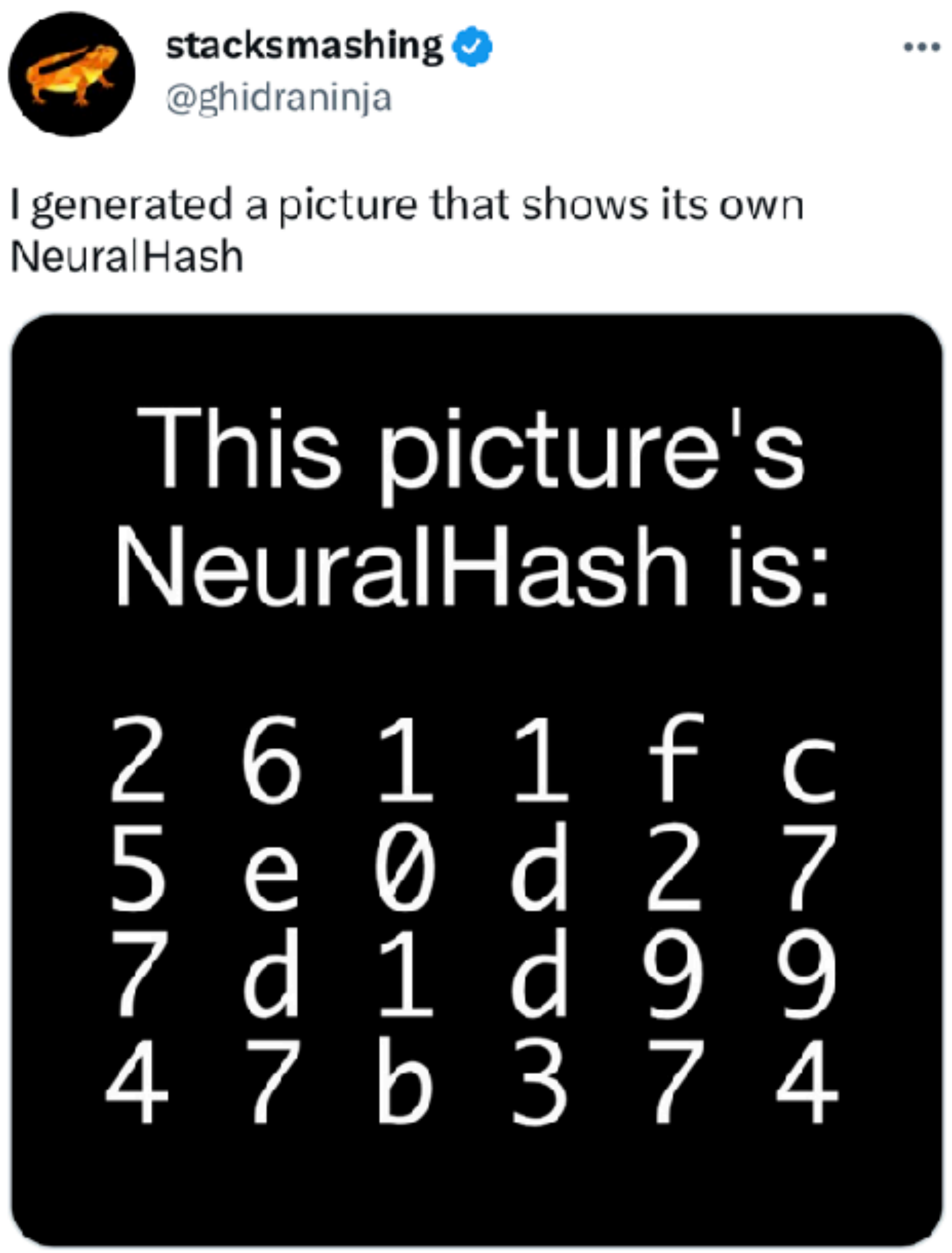
Moderation without E2EE

- Server given a database — hashes of “illegal” images

Resistant to small changes in image cropping, rotation etc.
Not collision resistant!

Moderation without E2EE

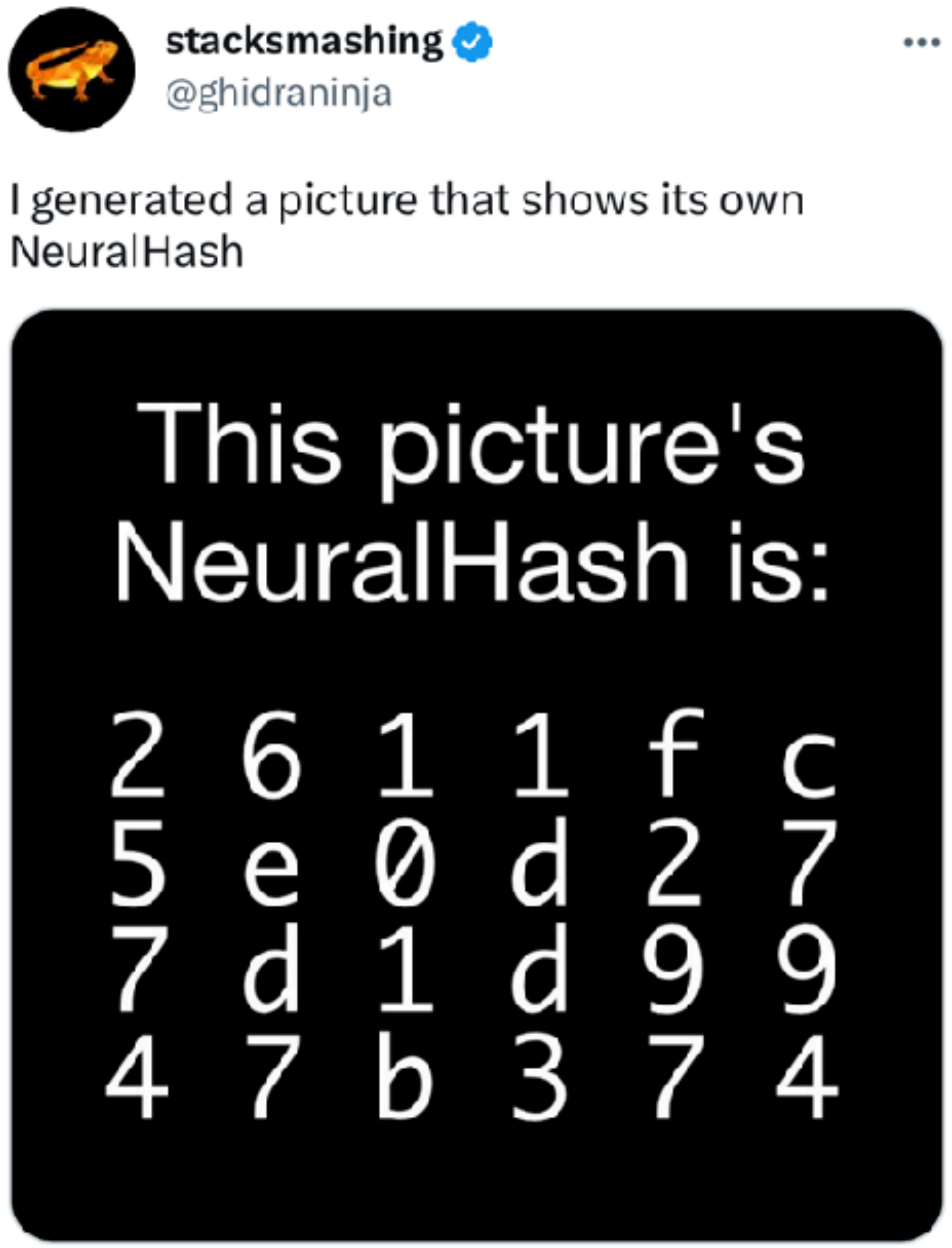
- Server given a database — hashes of “illegal” images



Moderation without E2EE

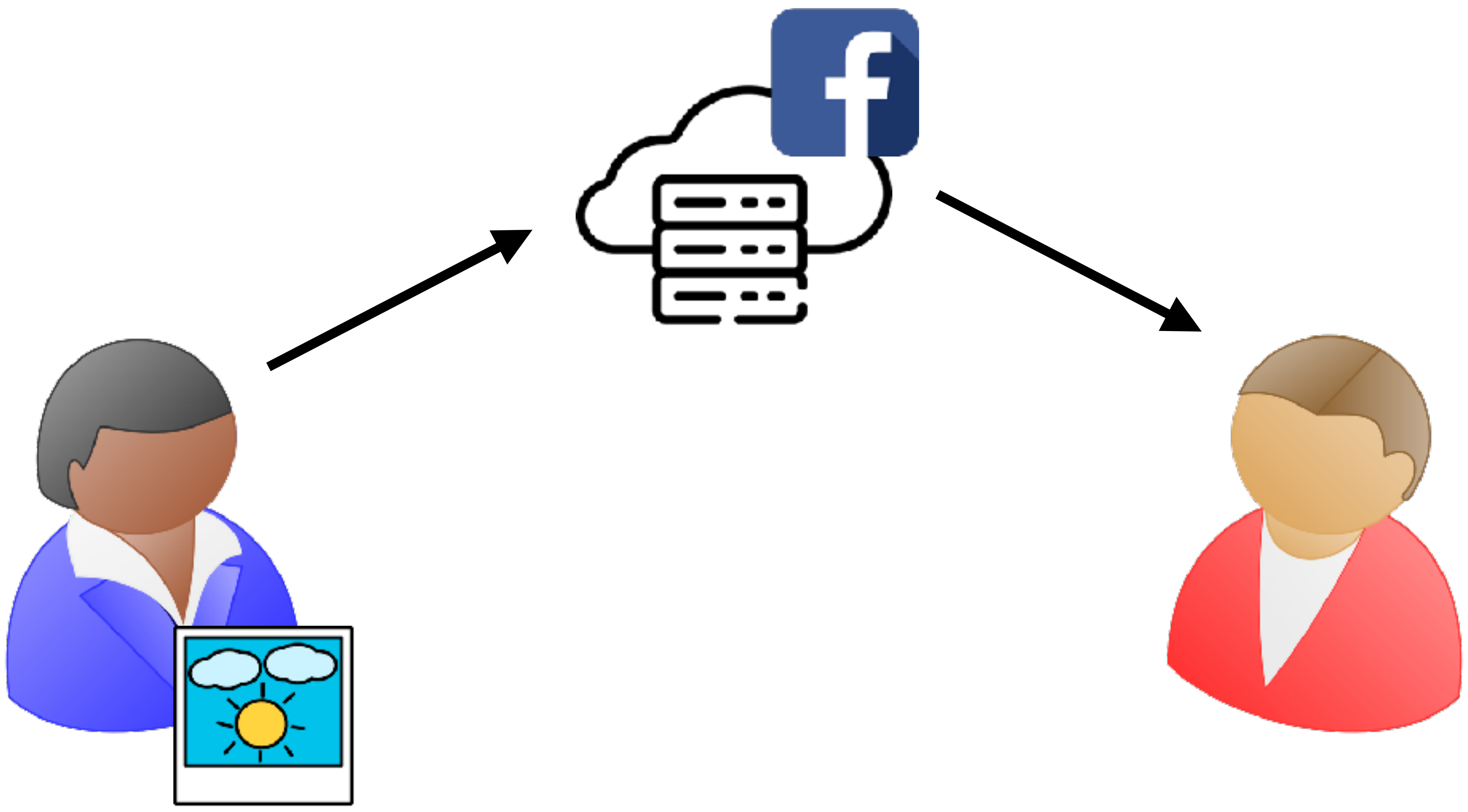
- Server given a database — hashes of “illegal” images

Resistant to small changes in image cropping, rotation etc.
Not collision resistant!



Moderation without E2EE

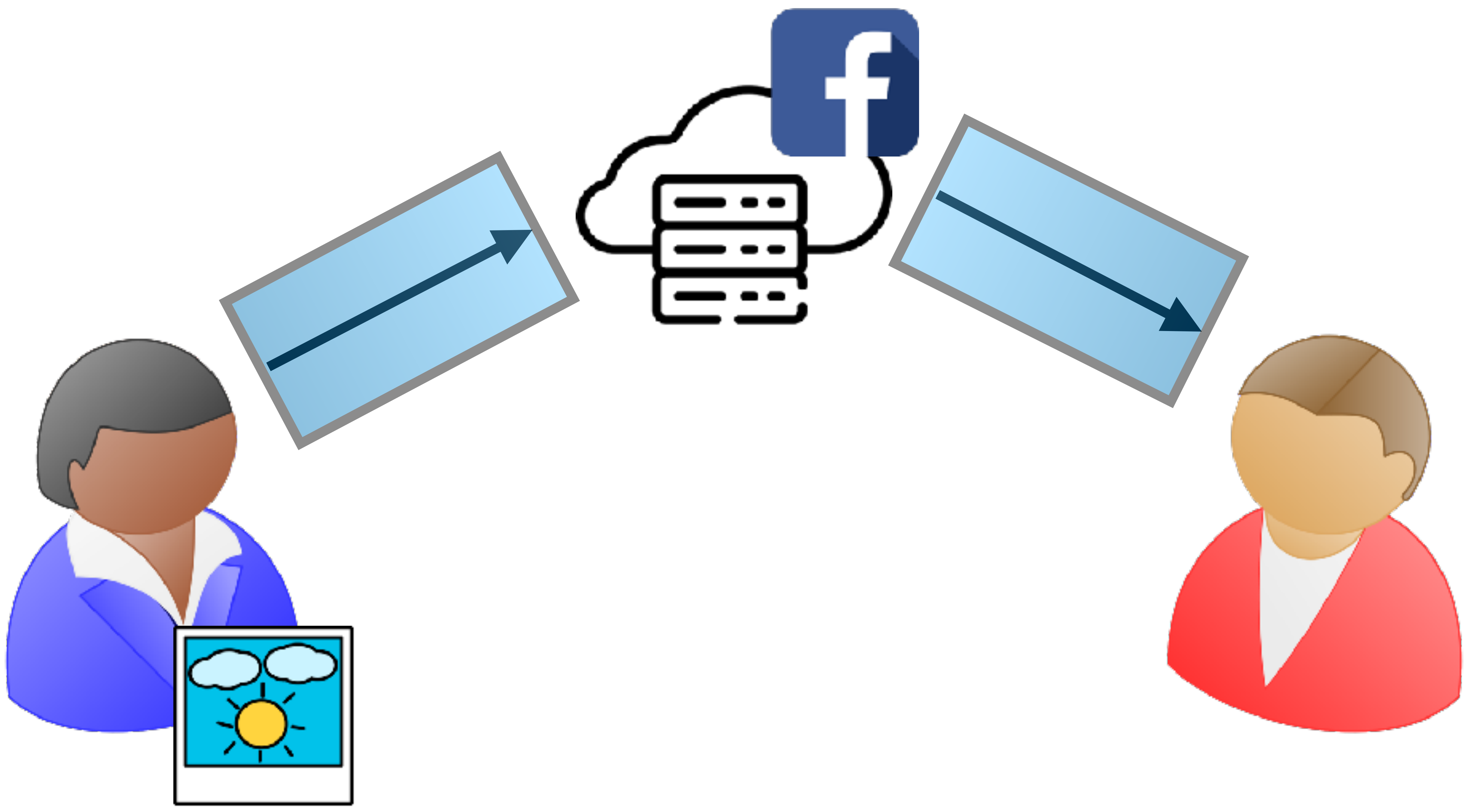
- Server given a database — hashes of “illegal” images
- Server can view all messages being exchanged



Moderation without E2EE

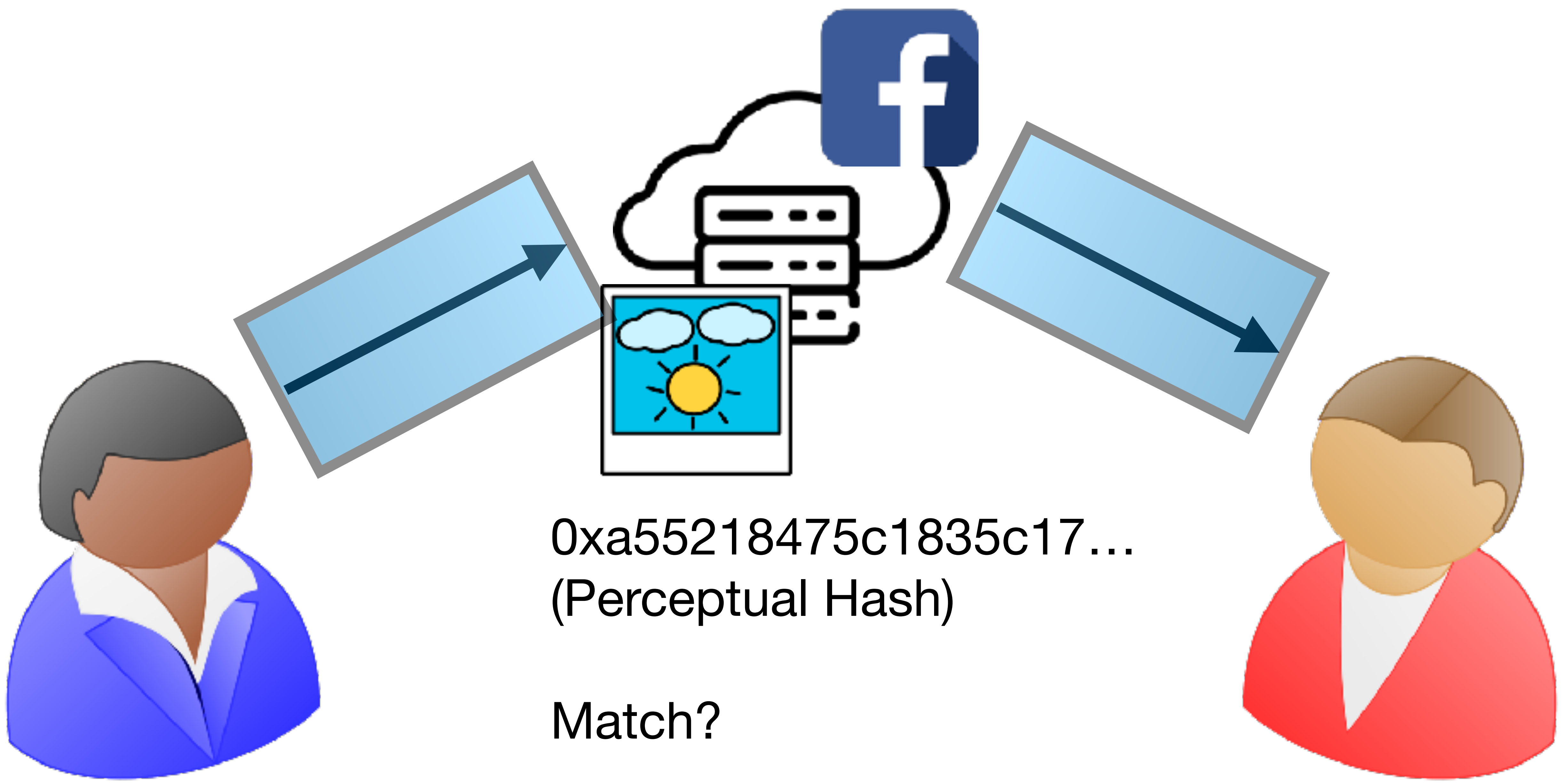
- Server given a database — <u>hashes</u> of "illegal" images

- Server can view all messages being exchanged



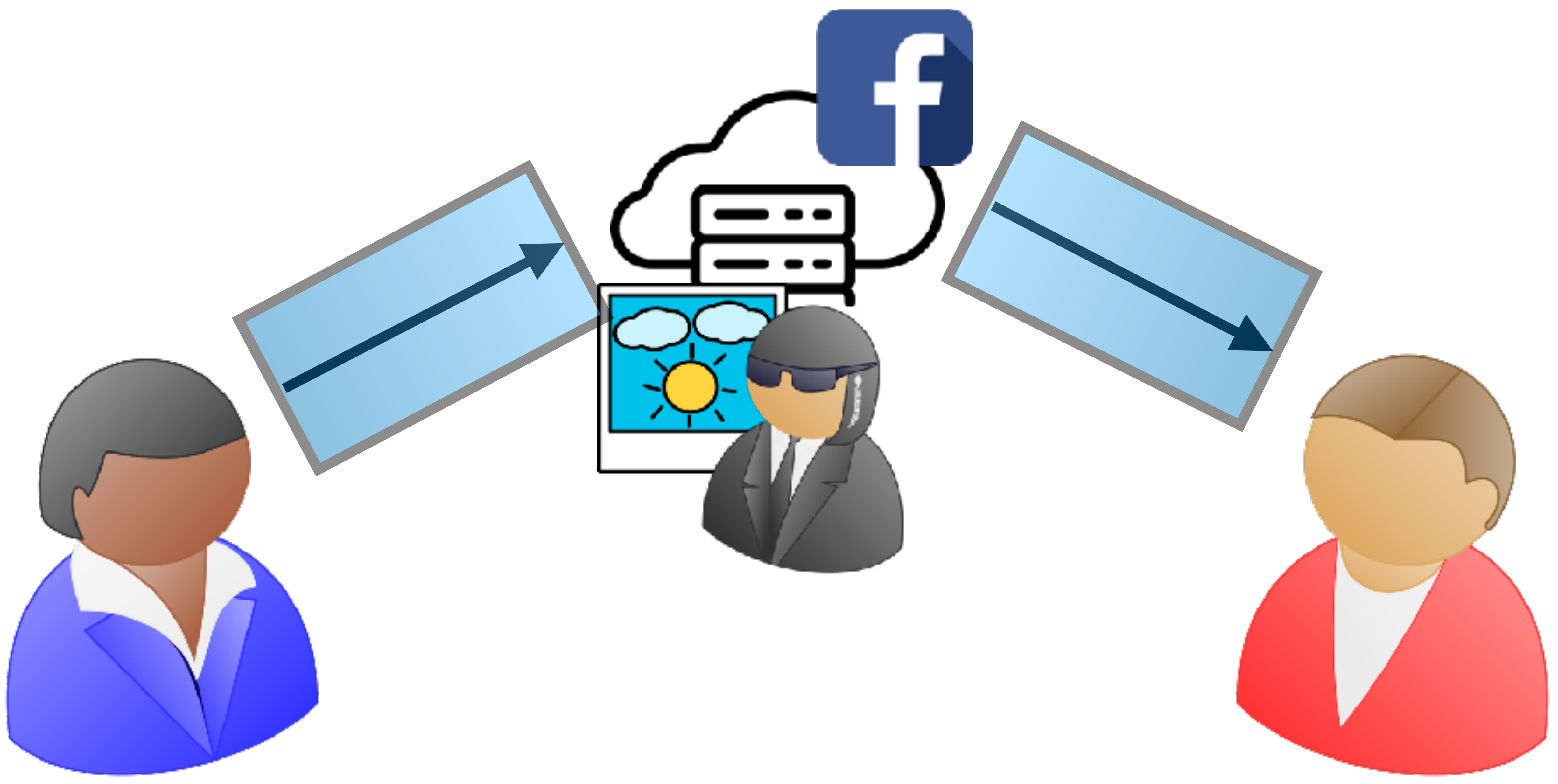
Moderation without E2EE

- Server given a database — <u>hashes</u> of “illegal” images
- Server can view all messages being exchanged



Moderation without E2EE

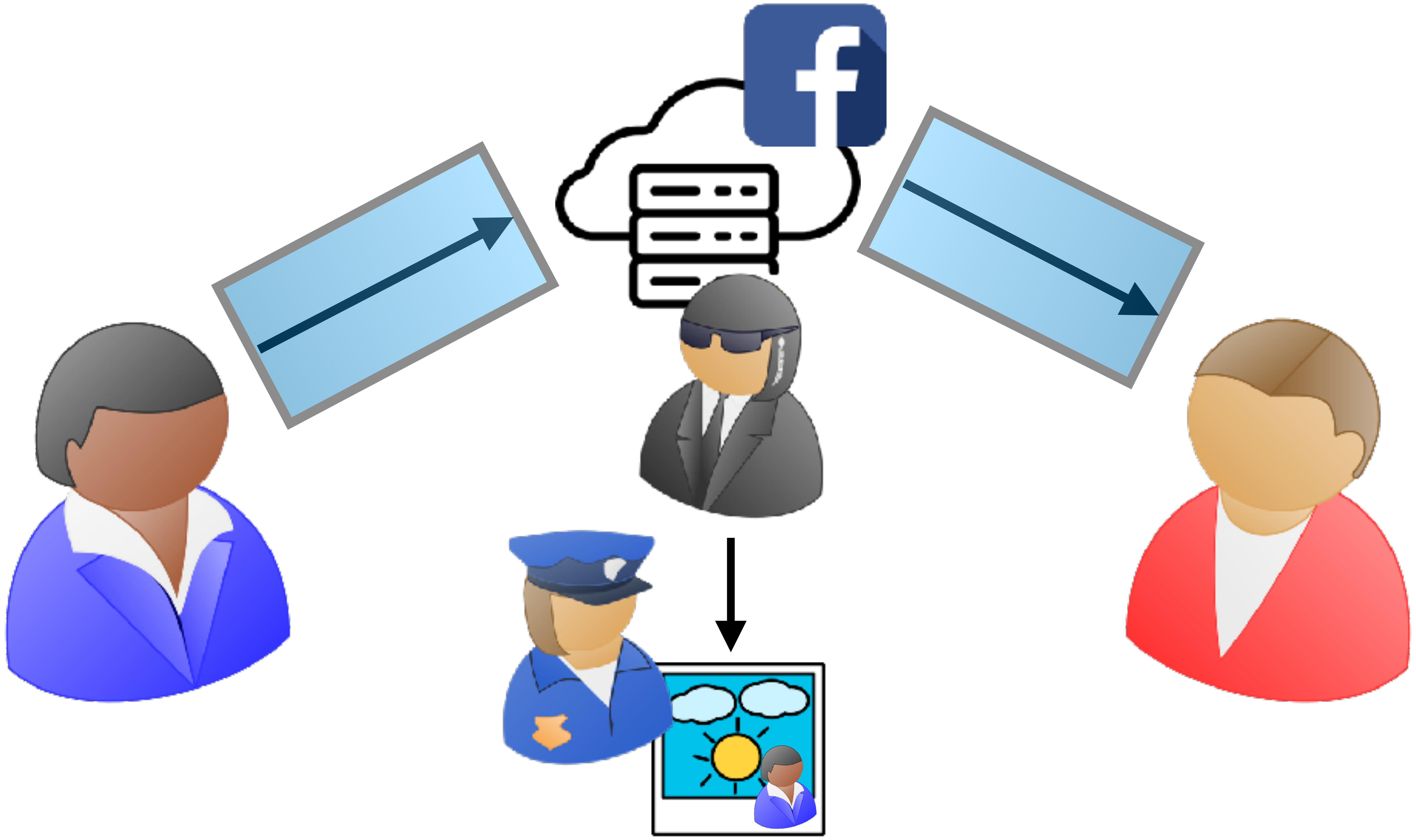
- Server given a database — hashes of “illegal” images
- Server can view all messages being exchanged



Moderation without E2EE

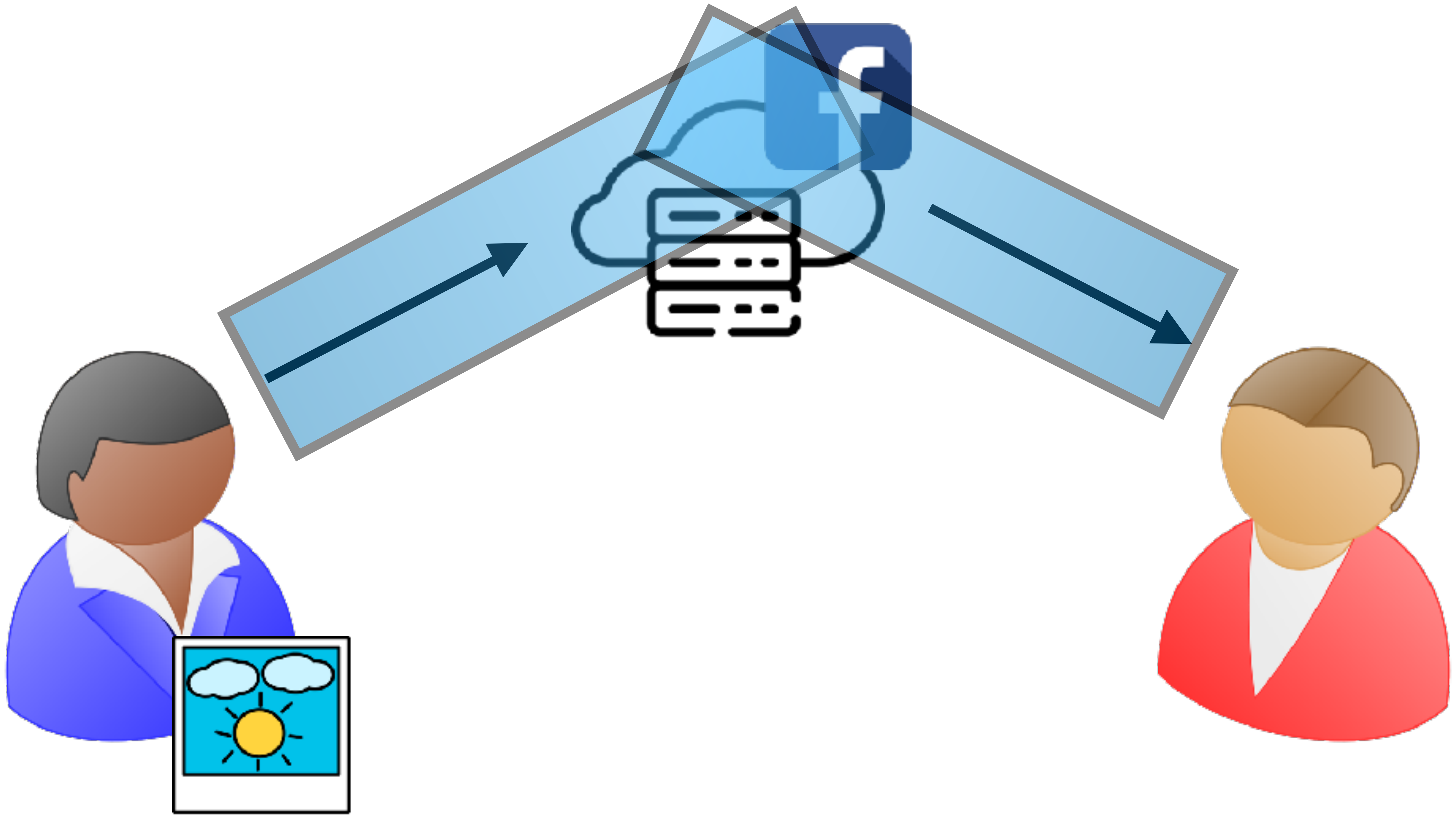
- Server given a database — <u>hashes</u> of “illegal” images

- Server can view all messages being exchanged



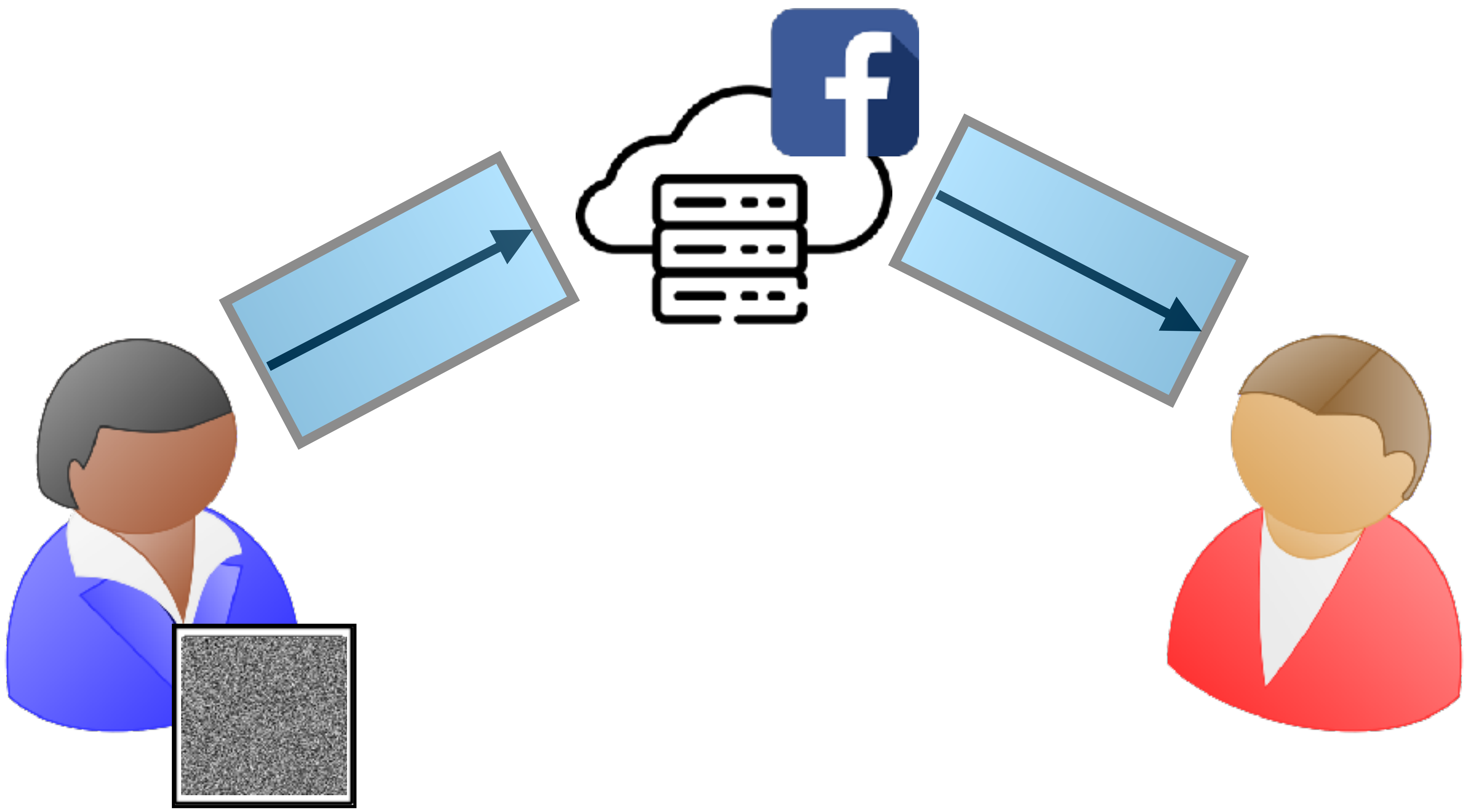
Moderation with~~out~~ E2EE?

- Server given a database — <u>hashes</u> of “illegal” images
- ~~Server can view all messages being exchanged~~



Some inherent limitations

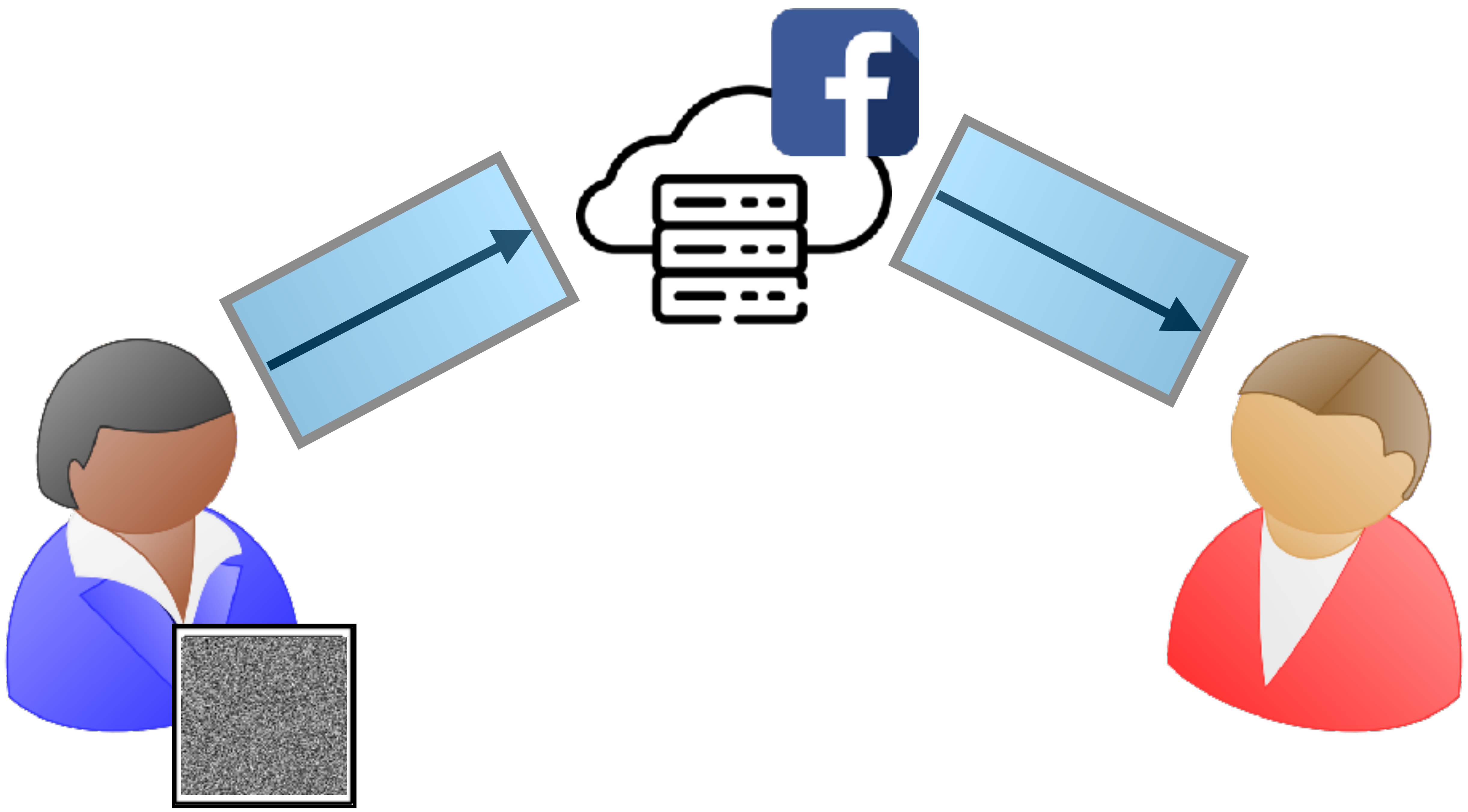
Malicious users can use steganography to hide content.



Some inherent limitations

Malicious users can use steganography to hide content.

Will persist even with cryptography. So who is moderation really targeting?

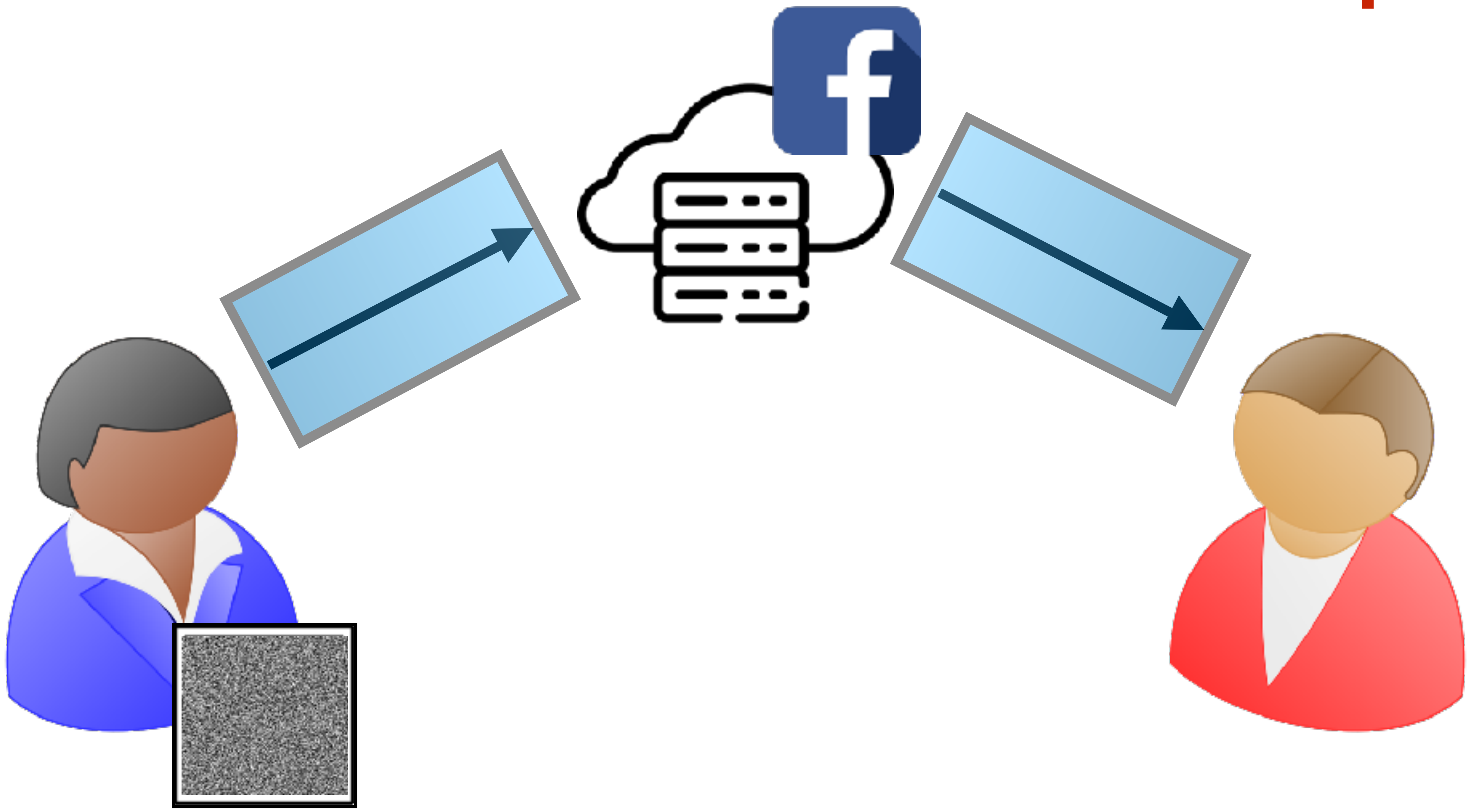


Some inherent limitations

Malicious users can use steganography to hide content.

Will persist even with cryptography. So who is moderation really targeting?

18m+ reports every year



What do we want from E2EE with moderation?

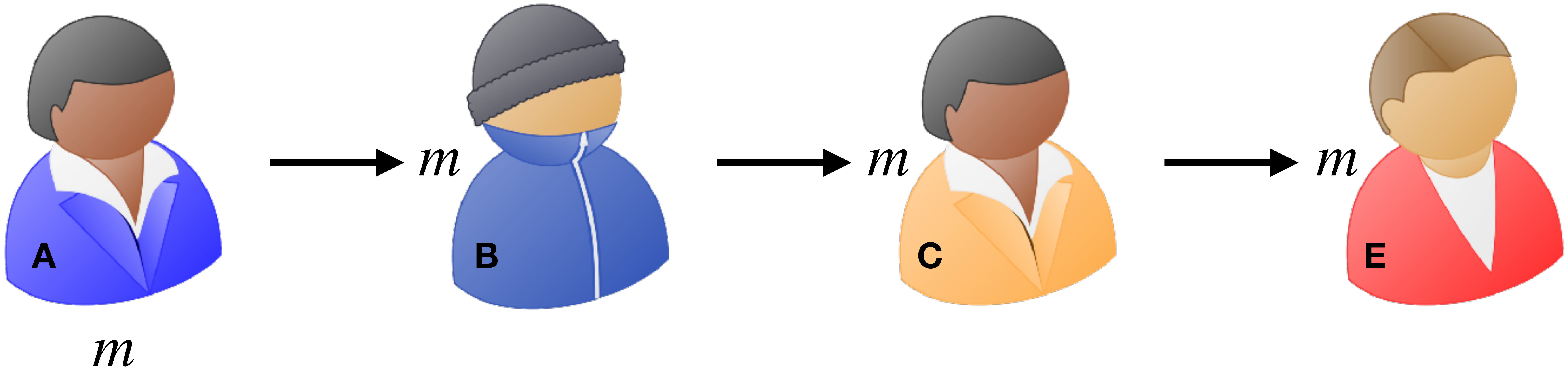
Minimum Requirements

1. Server learns no information about messages exchanged
2. Originator of “forwarded” messages remains anonymous

Minimum Requirements

1. Server learns no information about messages exchanged

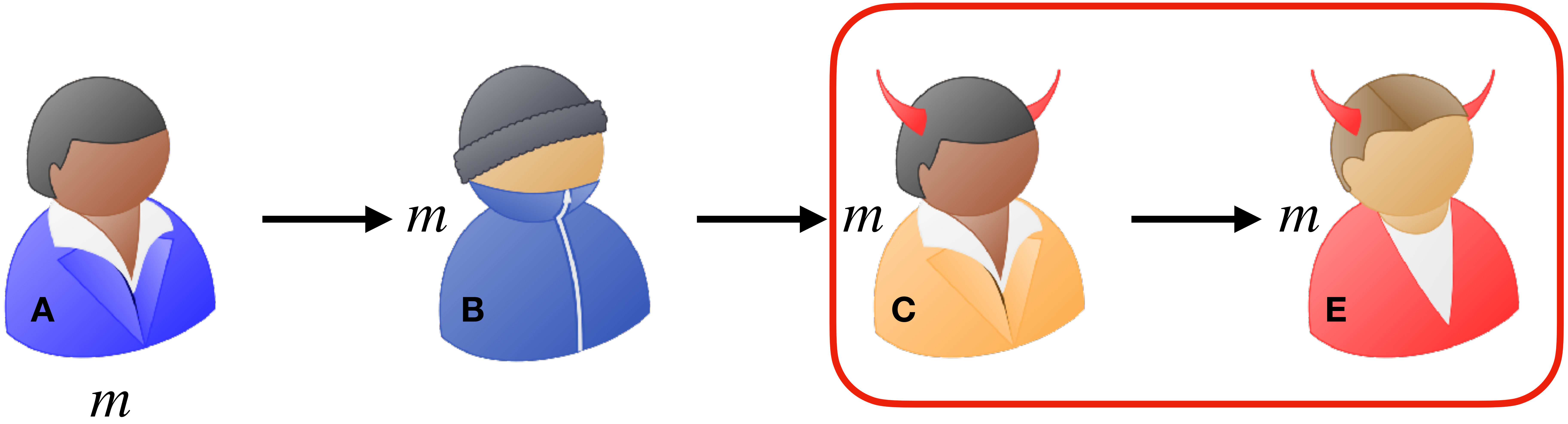
2. Originator of “forwarded” messages remains anonymous



Minimum Requirements

1. Server learns no information about messages exchanged

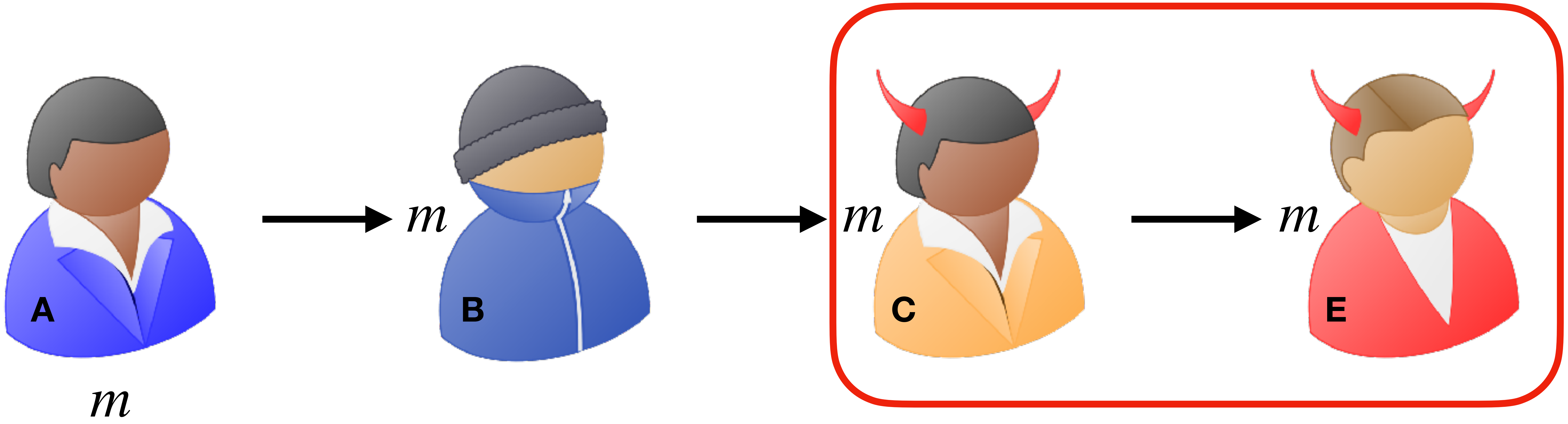
2. Originator of “forwarded” messages remains anonymous



No information learnt about who sent m

Minimum Requirements

1. Server learns no information about messages exchanged
2. Originator of “forwarded” messages remains anonymous



No information learnt about who sent m

What if server also colludes?
No more than that revealed by aux info — graph of messages

Minimum Requirements

1. Server learns no information about messages exchanged

2. Originator of “forwarded” messages remains anonymous

“Standard” E2EE messaging already satisfies this

Minimum Requirements

1. Server learns no information about messages exchanged

2. Originator of “forwarded” messages remains anonymous

“Standard” E2EE messaging already satisfies this

But no “content moderation”

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Feasibility: Group signatures are good enough

Line of work on traceback for E2EE achieves this + nice properties

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Feasibility: Group signatures are good enough

Line of work on traceback for E2EE achieves this + nice properties

Message Franking

GLR17 **LZHY+23**

FB Whitepaper **DGRW18** **TGLMR19**

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Feasibility: Group signatures are good enough

Line of work on traceback for E2EE achieves this + nice properties

Message Franking

GLR17 **LZHY+23**
FB Whitepaper **DGRW18** **TGLMR19**

Traceback

PEB21
TMR19 **LRTY21** **IAV22**

Group Signatures

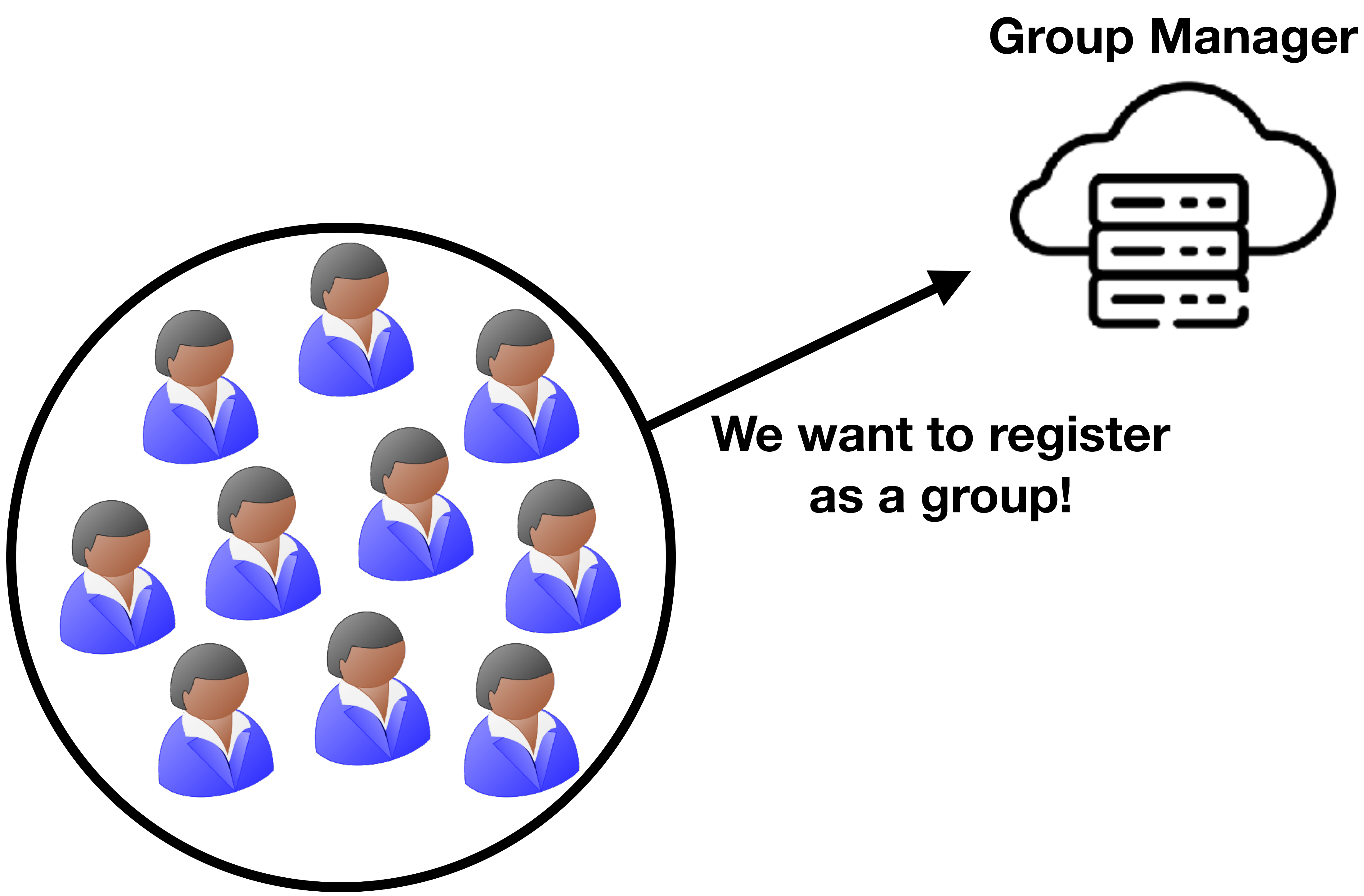
Member of a group can anonymously sign a message on behalf of the group

But there is a group manager who can identify signer of a message

Group Signatures

Member of a group can anonymously sign a message on behalf of the group

But there is a group manager who can identify signer of a message

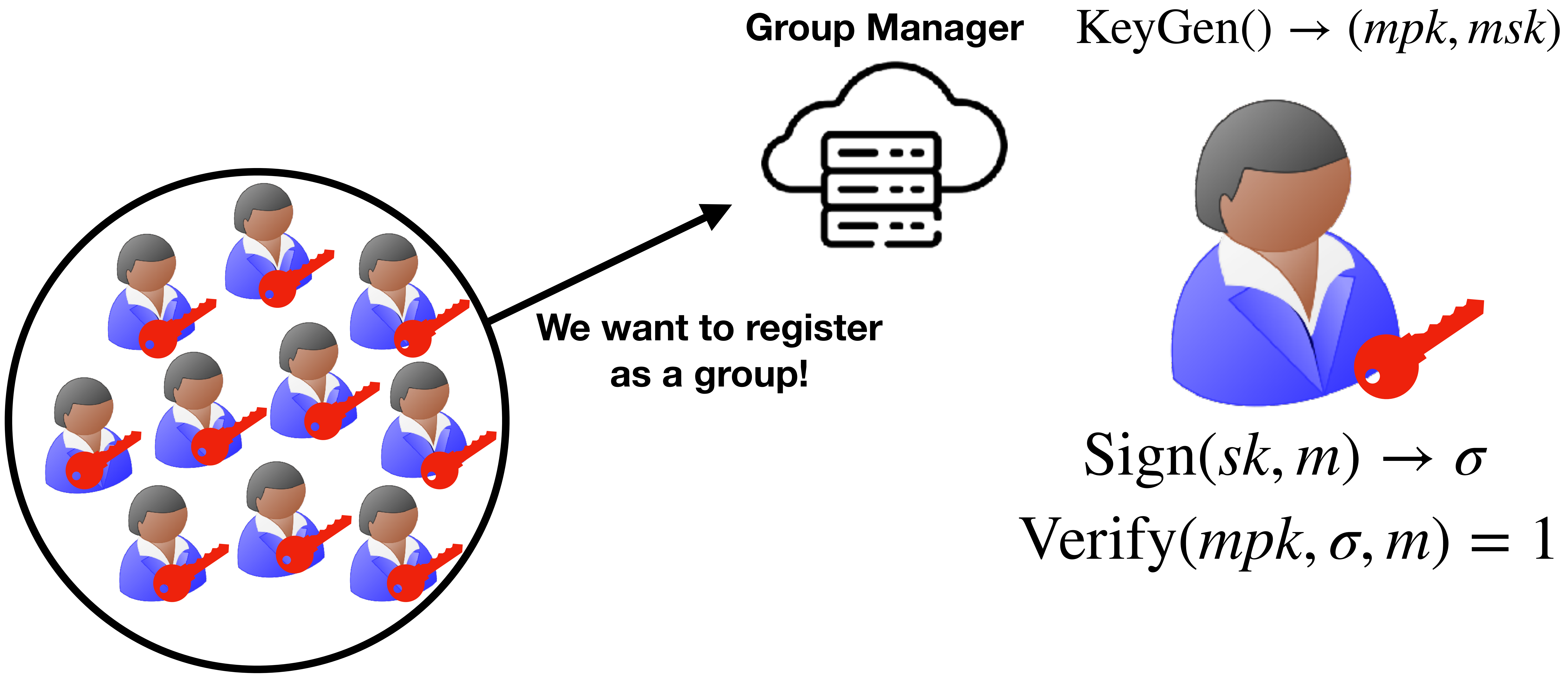


$\mathrm{KeyGen}() \rightarrow (mpk, msk)$

Group Signatures

Member of a group can anonymously sign a message on behalf of the group

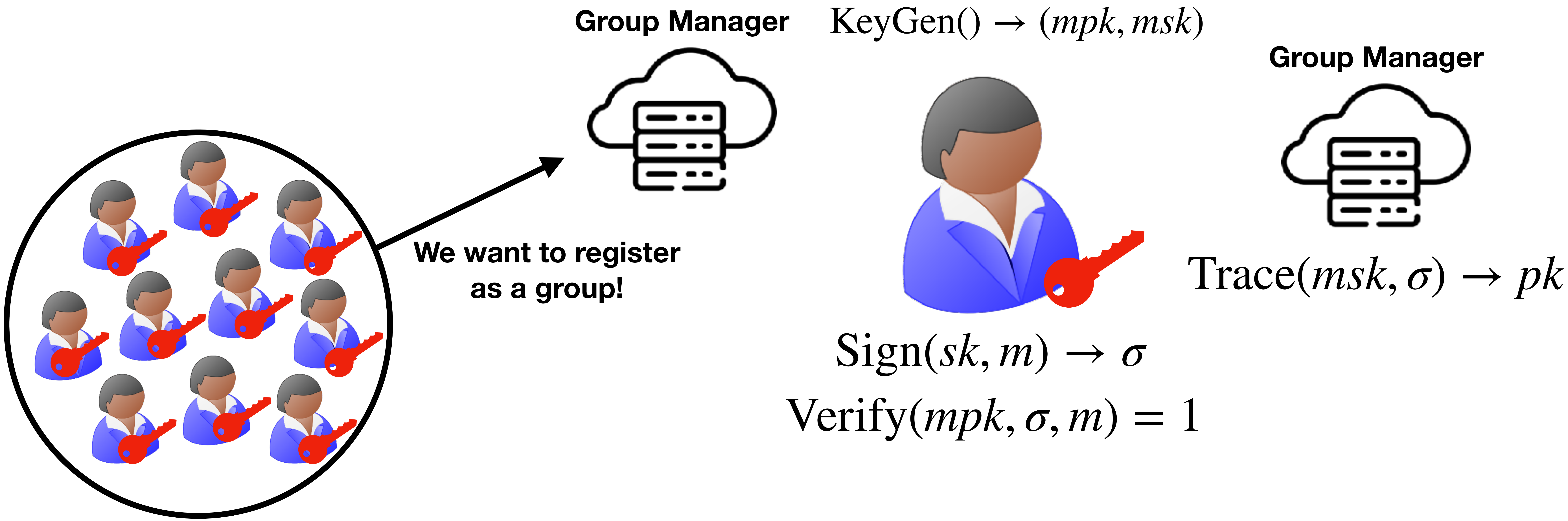
But there is a group manager who can identify signer of a message



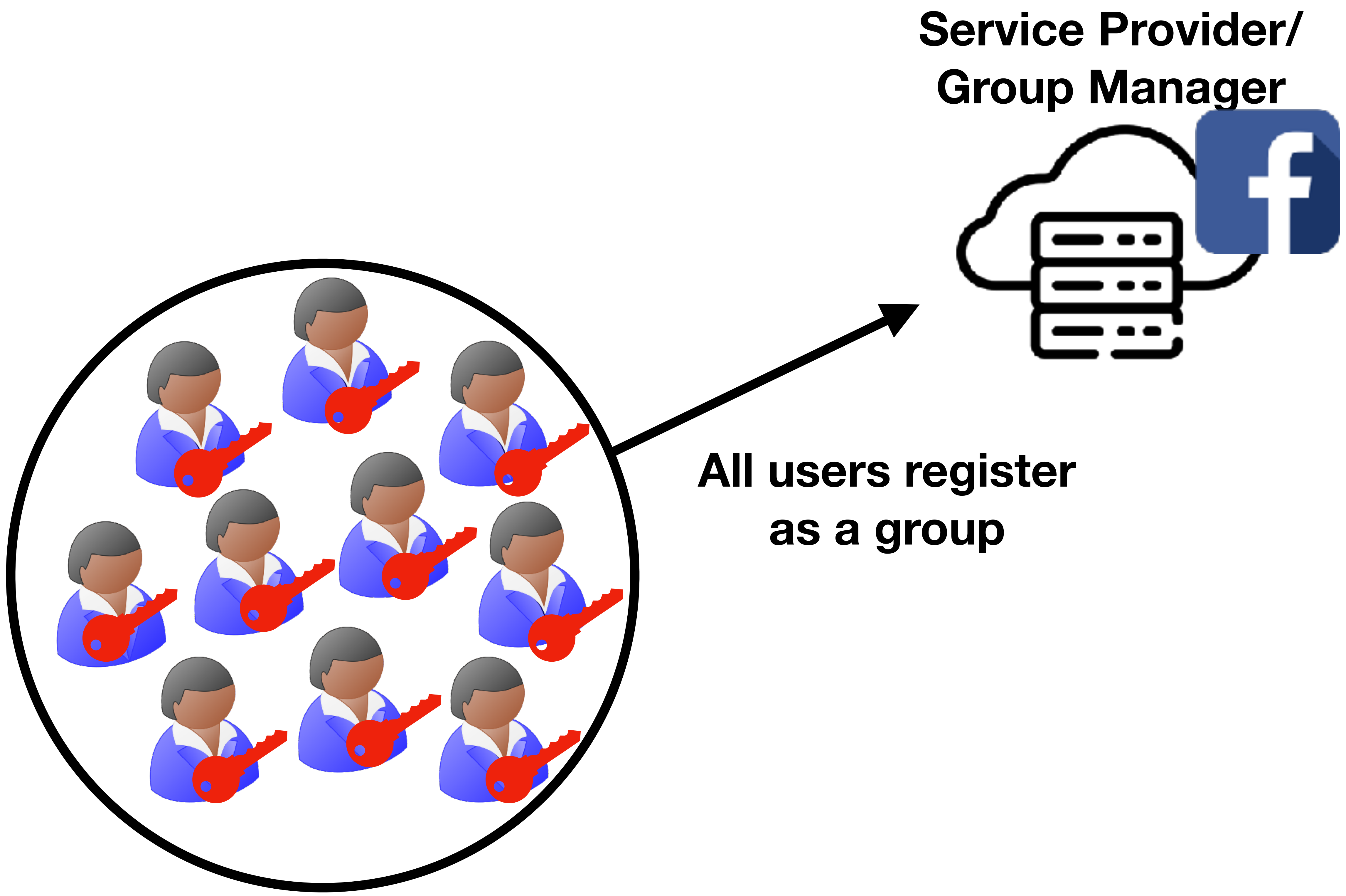
Group Signatures

Member of a group can anonymously sign a message on behalf of the group

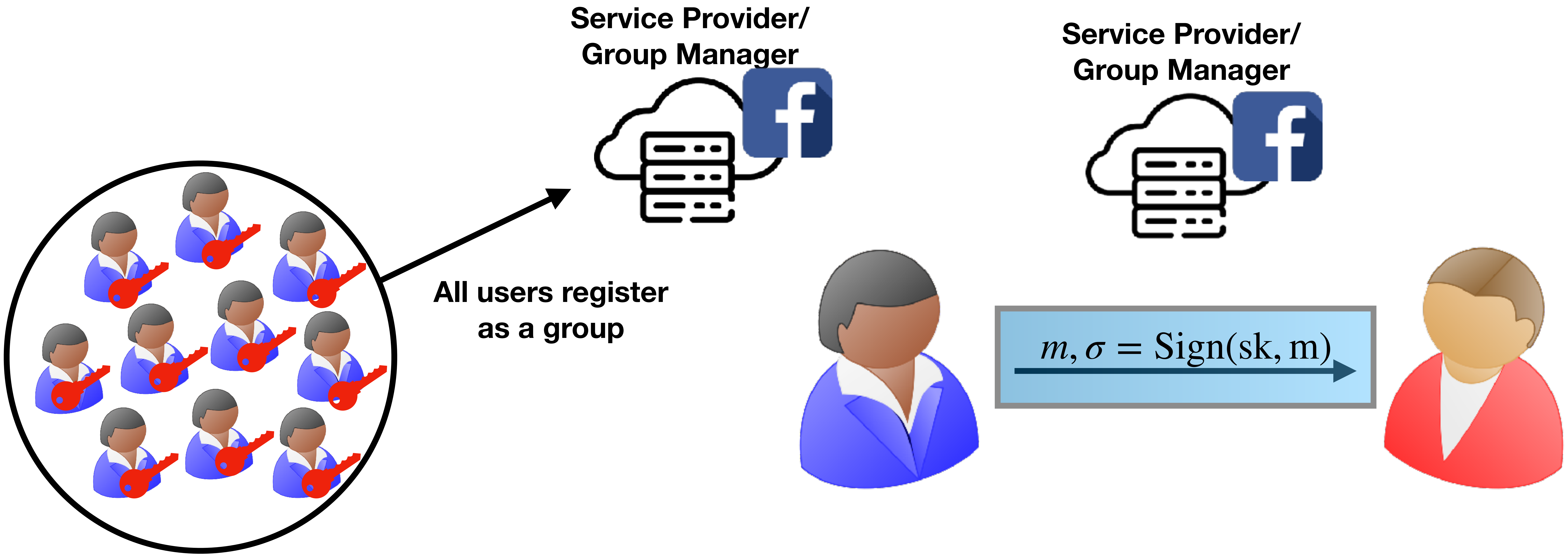
But there is a group manager who can identify signer of a message



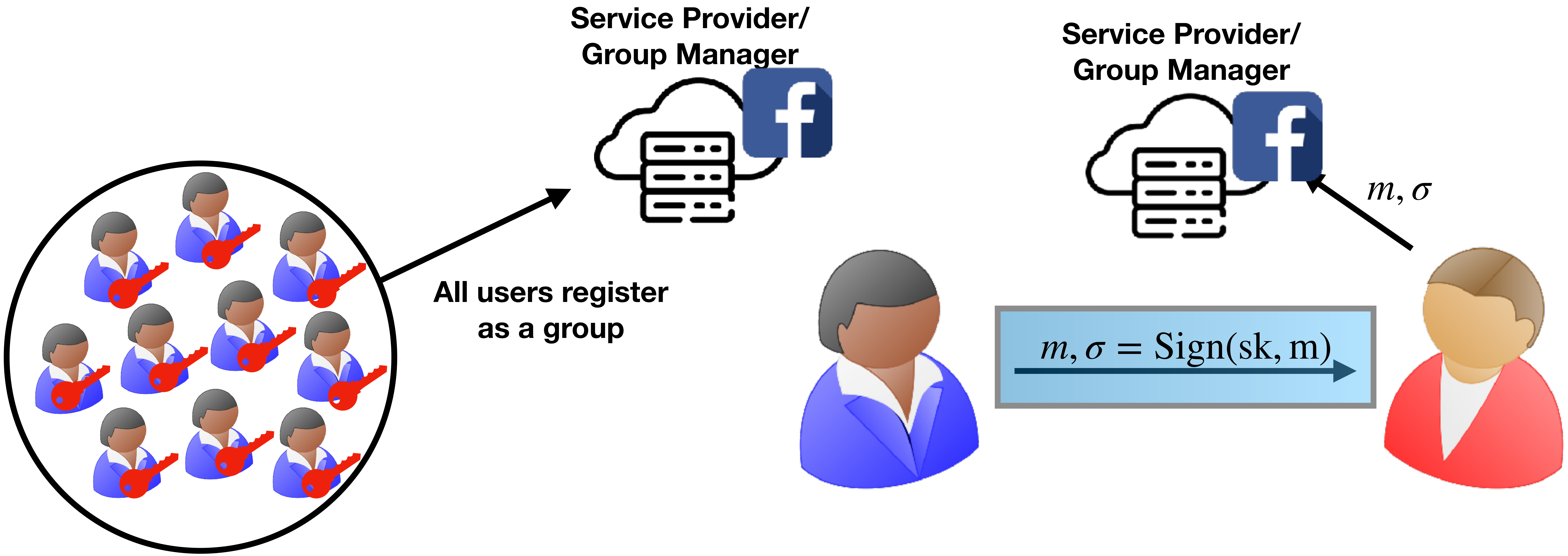
Group Signatures → Content Moderation



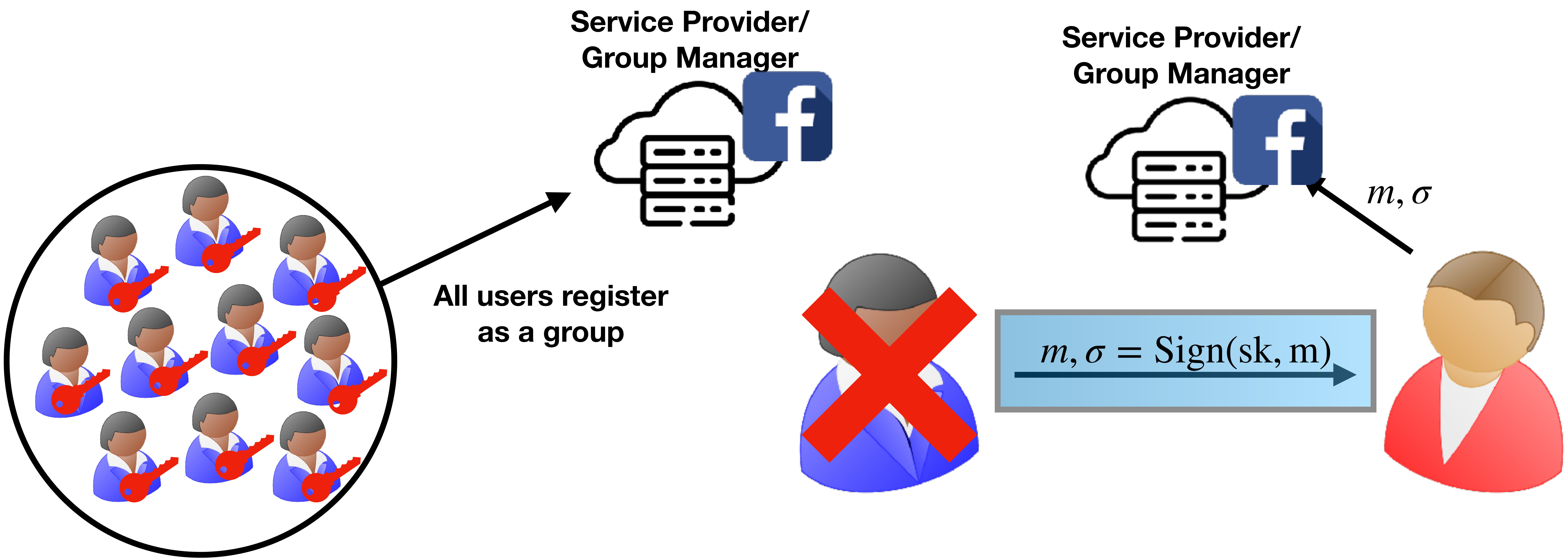
Group Signatures → Content Moderation



Group Signatures → Content Moderation



Group Signatures → Content Moderation



1. If no report, malicious server learns no information about messages exchanged
2. If no report, originator of "forwarded" messages remains anonymous
3. If a user receives some content and reports it, server can identify the originator.

Minimum Requirements

1. Server learns no information about messages exchanged (no report)
2. Originator of “forwarded” messages remains anonymous (no report)
3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Is this really sufficient?

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Is this really sufficient?

What happens when a malicious server and user collude??

Let’s try to strengthen this

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some *illegal* content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

2. Originator of “forwarded” messages remains anonymous (no report)

3. If a user receives some *illegal* content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

Need to define illegal content
We will use the “database” definition

Minimum Requirements

1. Server learns no information about messages exchanged (no report)

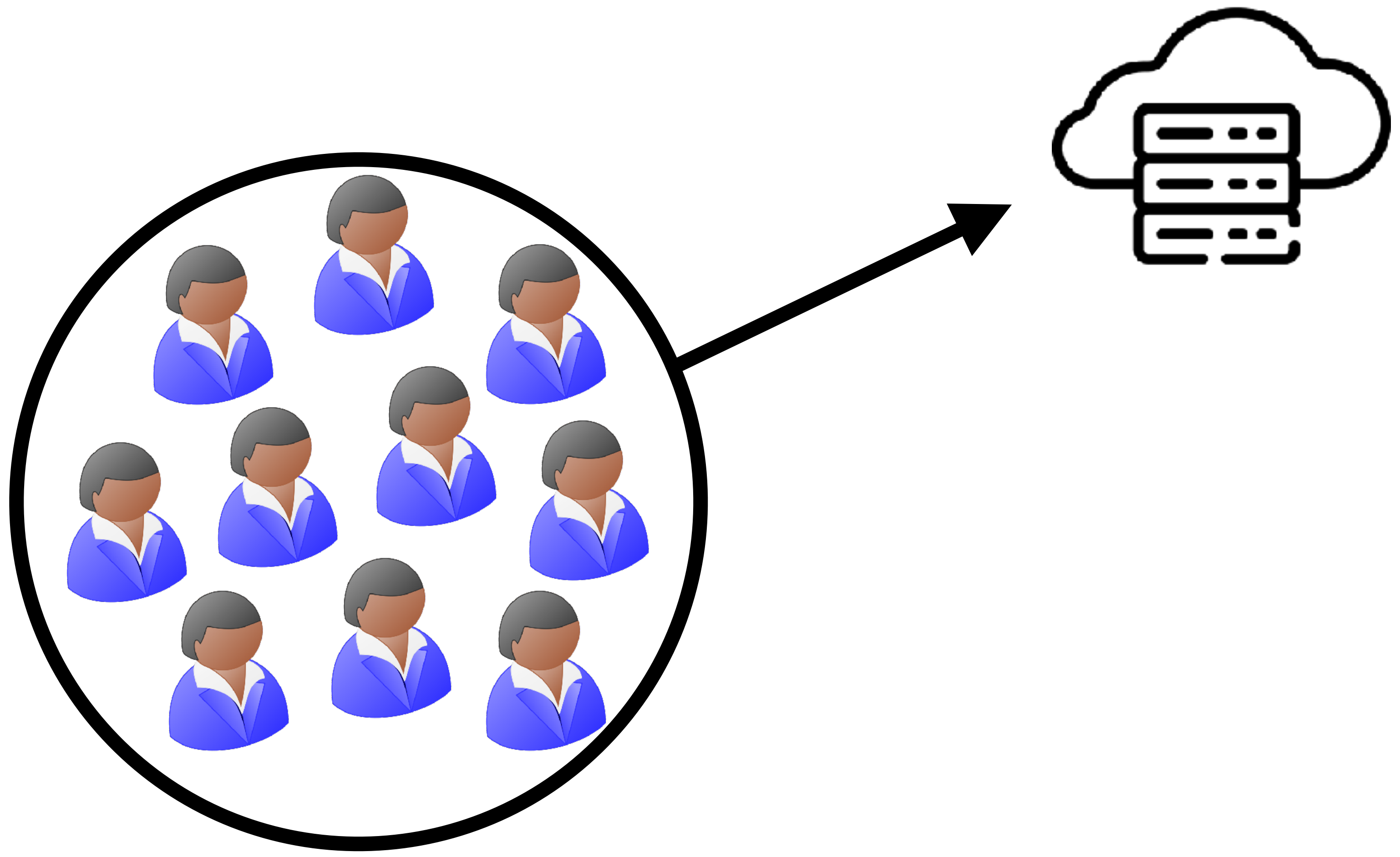
2. Originator of “forwarded” messages remains anonymous (no report)

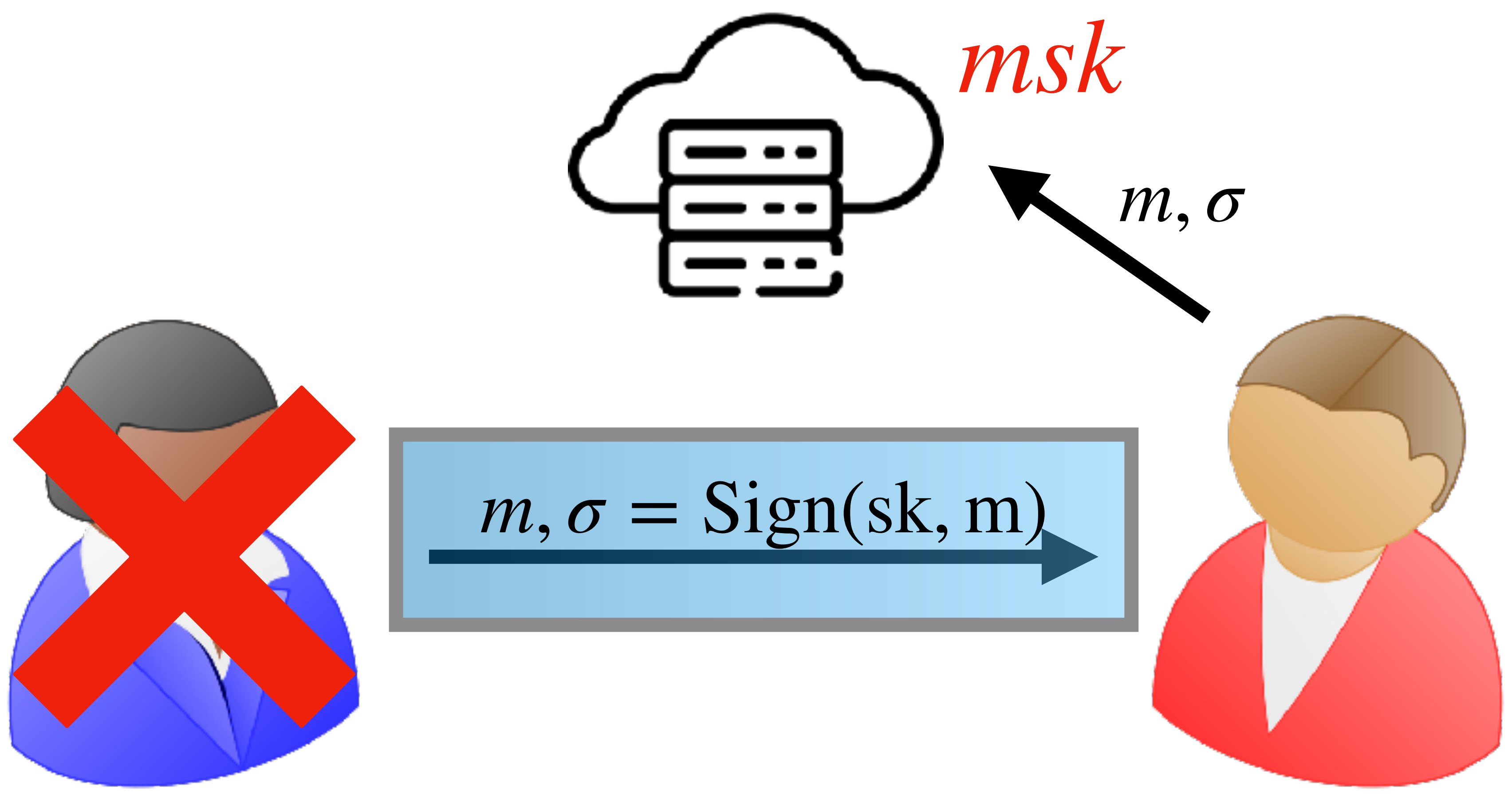
3. If a user receives some *illegal* content (even if forwarded) and reports it, server can identify the originator. No help needed from other users.

4. Originator of *harmless* content remains anonymous, even if a malicious user and server collude.

Achieving security against malicious servers

Group Signatures → Content Moderation





What's going wrong here?

Server has too much power as it has msk.

Let's tie its hands!

Design Philosophy

- Want to avoid a master secret key as there is no server accountability
- Server should only be able to deanonymize “bad” message signers
- Paradigm of “pre-constraining” encryption keys introduced in [AJJM22]
- We build on this and introduce Pre-Constrained Group Signatures

Pre-Constrained Group Signatures

Group Manager

$\mathrm{KeyGen}(D) \rightarrow (mpk, msk)$

Database of illegal images for which signers can be identified

Pre-Constrained Group Signatures

Group Manager

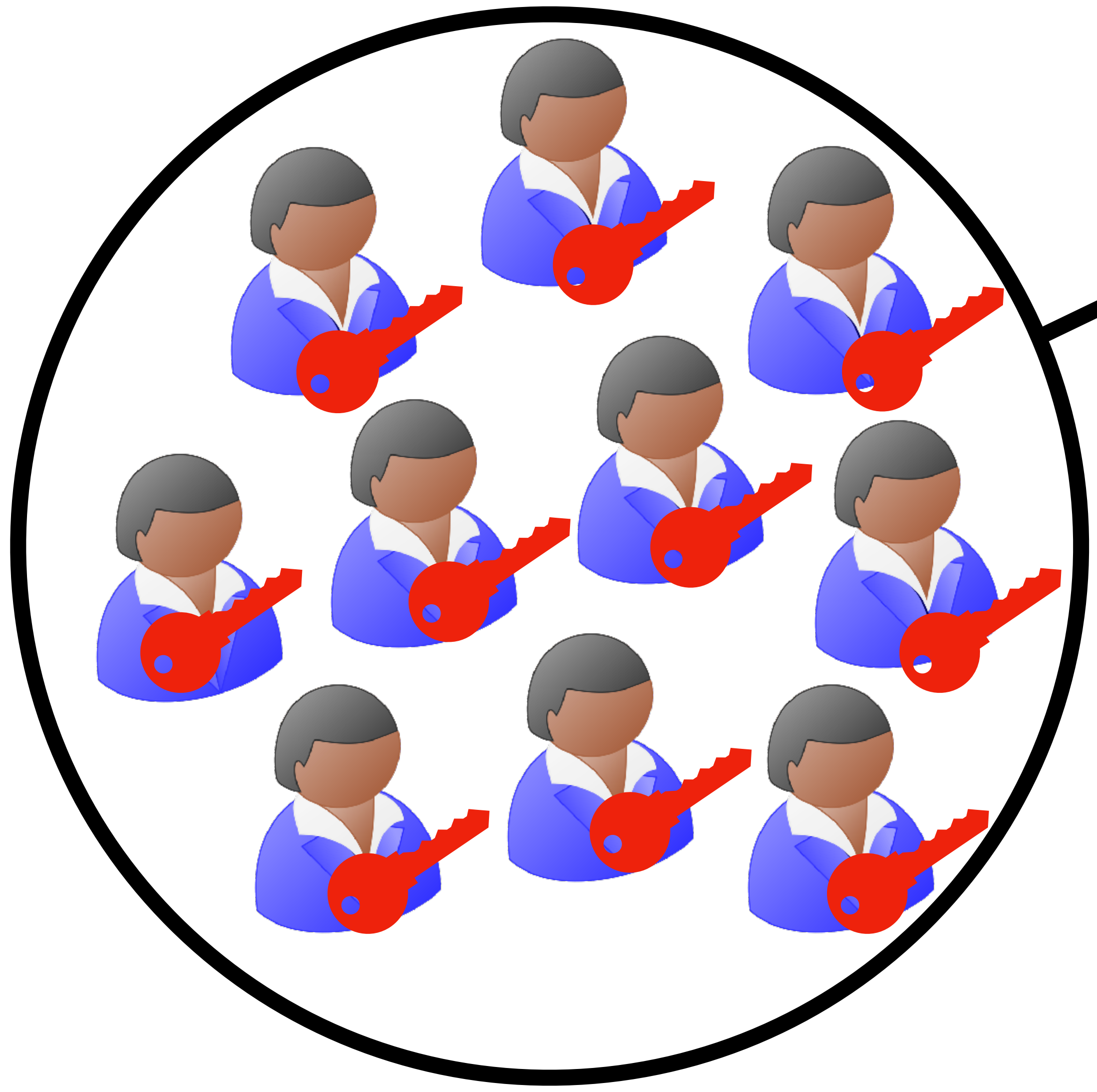
$\mathrm{KeyGen}(D) \rightarrow (mpk, msk)$

Database of illegal images for which signers can be identified

<u>Public key should not leak D</u>

Can enforce that D is signed by NCMEC

Pre-Constrained Group Signatures



Group Manager

$\boxed{\mathrm{KeyGen}(D) \to (mpk, msk)}$

Can be generated maliciously!

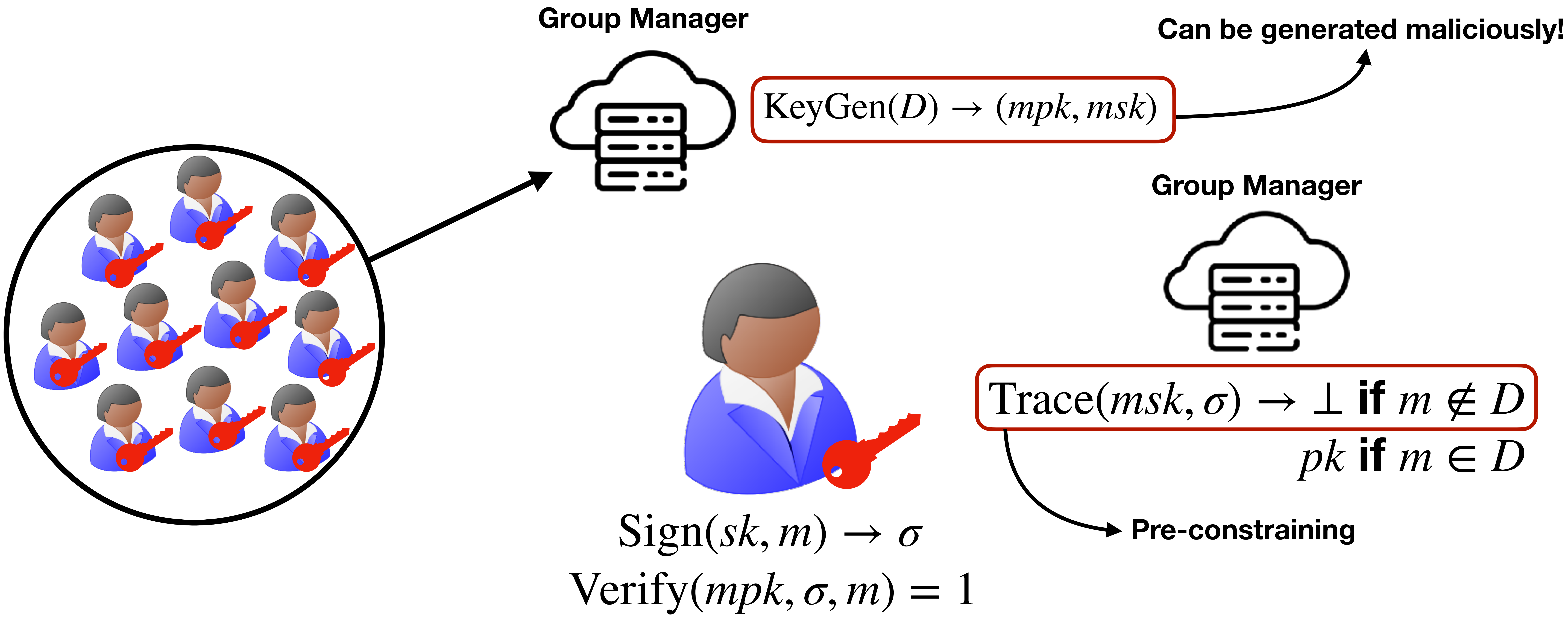
Group Manager

$$\mathrm{Trace}(msk, \sigma) \to \begin{cases} \perp \textbf{ if } m \notin D \\ pk \textbf{ if } m \in D \end{cases}$$

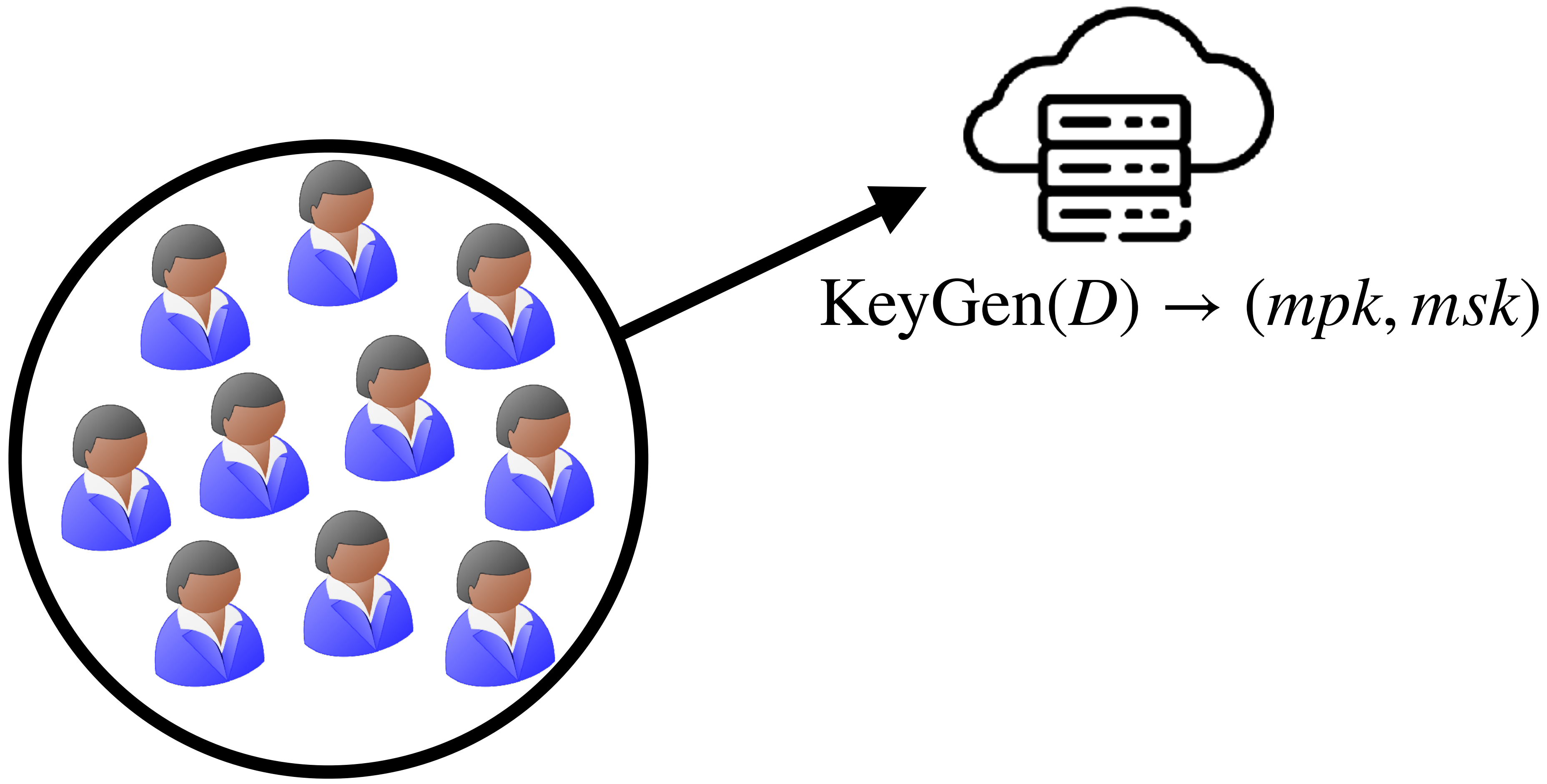
$$\mathrm{Sign}(sk, m) \to \sigma$$

$$\mathrm{Verify}(mpk, \sigma, m) = 1$$

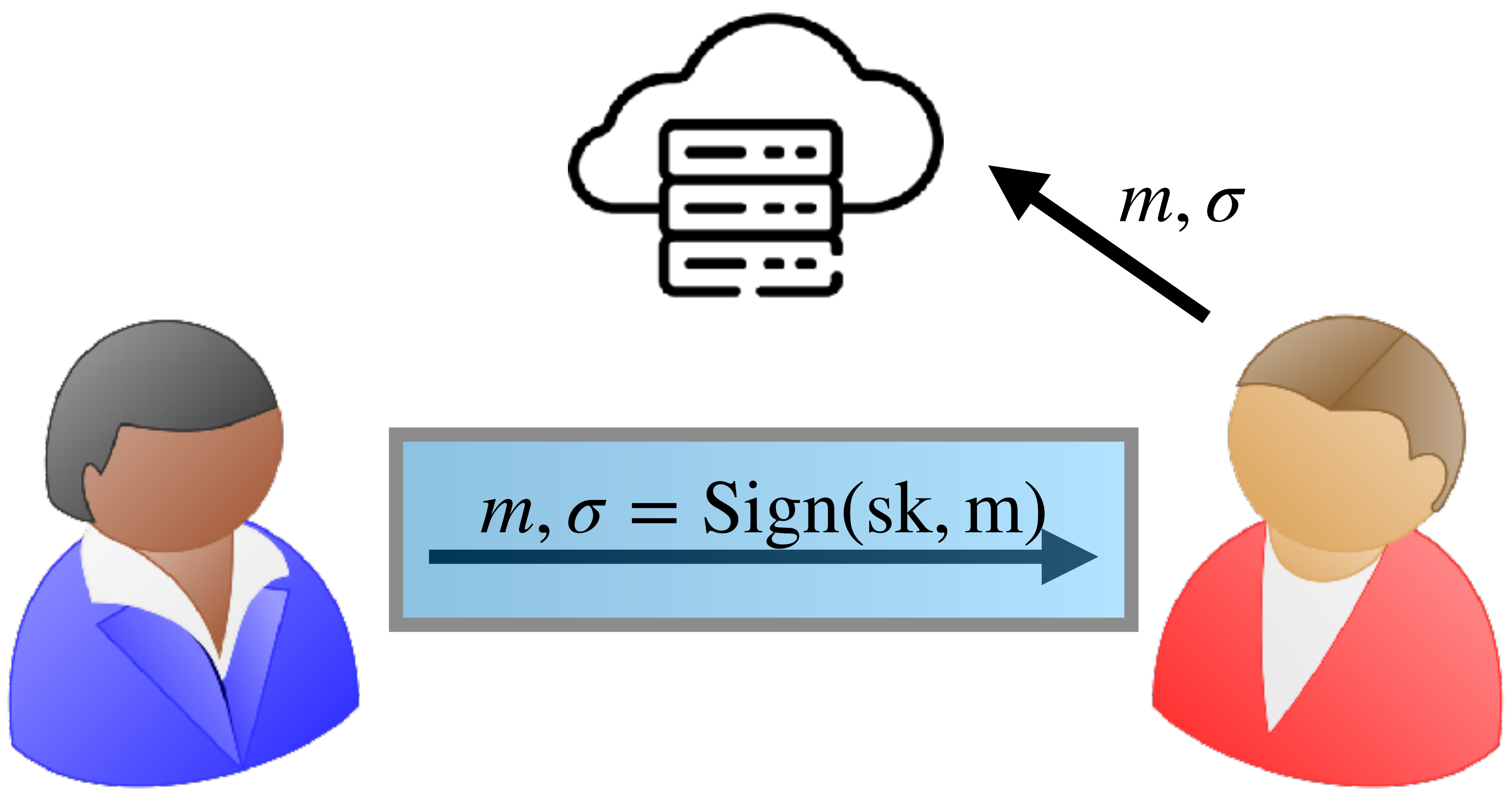
Pre-Constrained Group Signatures



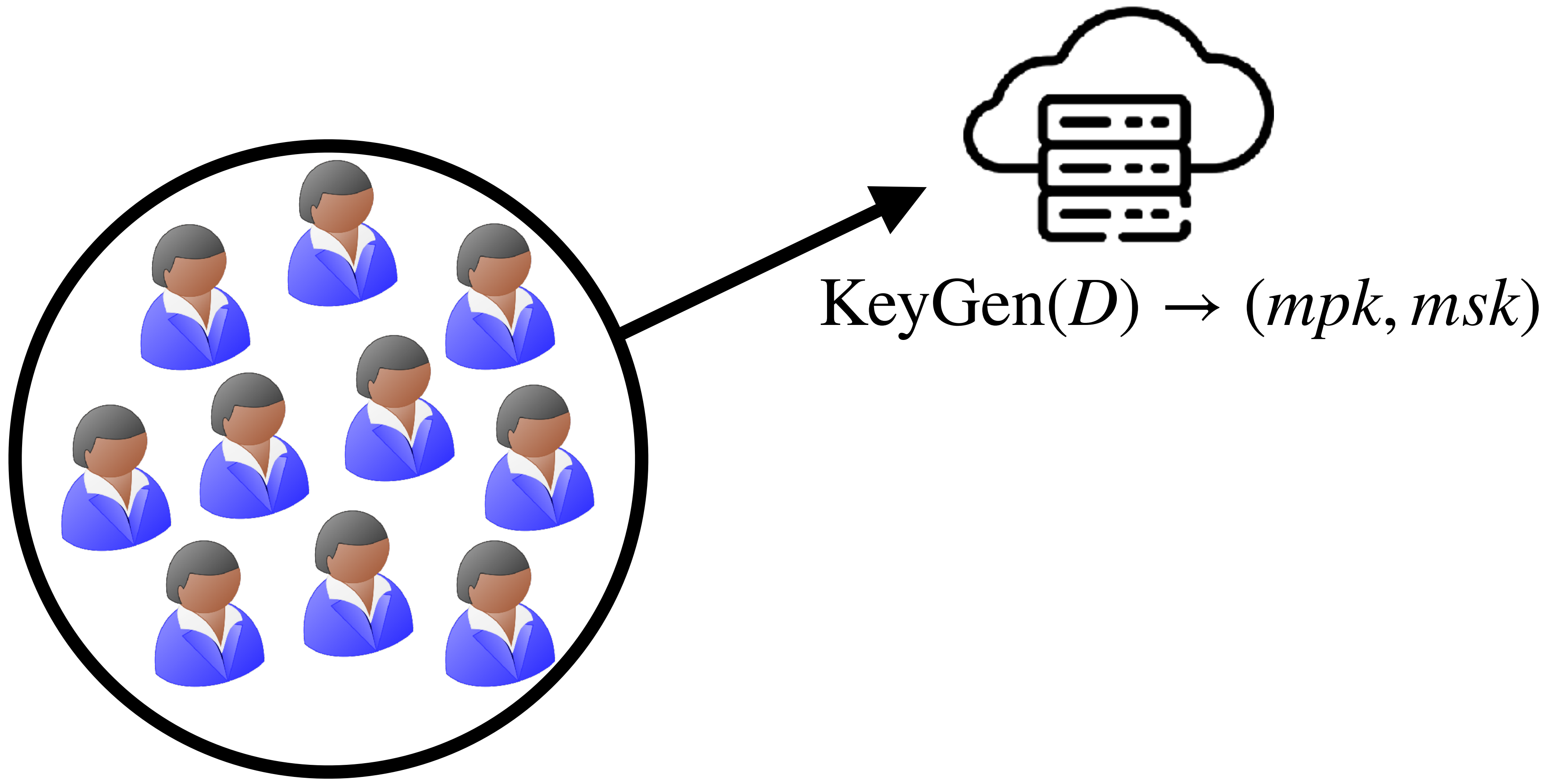
Pre-Constrained Group Signatures → Content Moderation



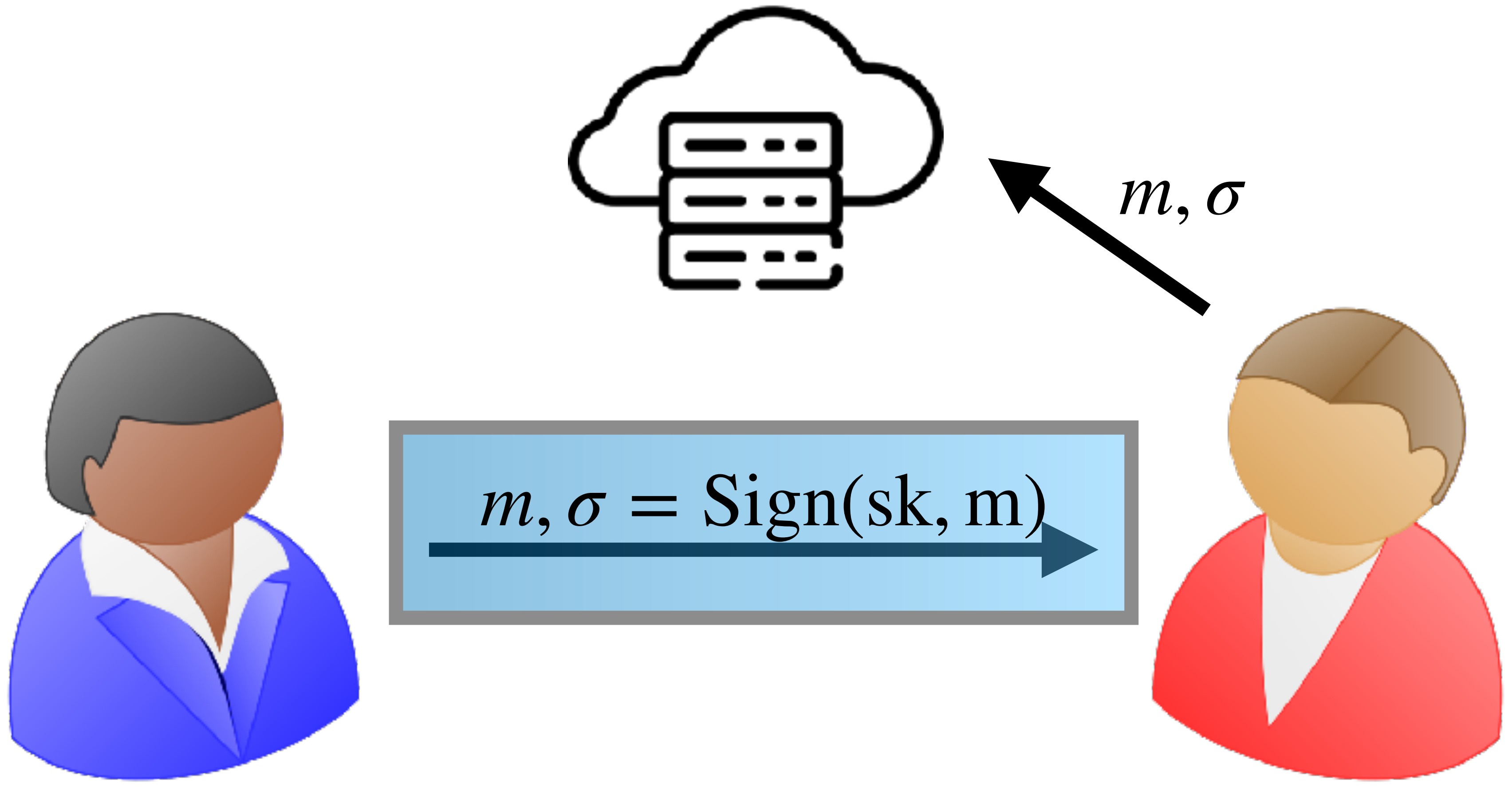
Can identify user only if m is "illegal"



Pre-Constrained Group Signatures → Content Moderation



Can identify user only if m is “illegal”



Privacy of honest users is unaffected!

How do we pre-constrain Group Signatures?

Compiler for Pre-Constrained Group Signatures

Compiler for Pre-Constrained Group Signatures

<u>Let's start with a generic construction of Group Signatures</u>

Group Signature: $mpk = (vk_s, pk_s)$

Public Key of a Public Key encryption scheme

Verification Key of a Signature Scheme

Compiler for Pre-Constrained Group Signatures

Let's start with a generic construction of Group Signatures

Group Signature: $mpk = (vk_s, pk_s)$

- $ct = \text{Enc}_{pk_s}(pk_c; r)$

Client's public key

Compiler for Pre-Constrained Group Signatures

<u>Let's start with a generic construction of Group Signatures</u>

Group Signature: $mpk = (vk_s, pk_s)$

- $ct = \mathrm{Enc}_{pk_s}(pk_c; r)$
- Simulation Extractable NIZK:

  A. I know a server signature σ on my public key pk_c

  B. I encrypted my public key pk_c using randomness r

  C. I know the secret key sk_c corresponding to pk_c

  D. m is a tag in the NIZK

$$ct = \mathrm{Enc}(pk_c; r), \Pi = \{sk_c, r, \sigma \mid \boxed{\mathrm{Verify}_{vk_s}(pk_c, \sigma) = 1} \wedge \boxed{ct = \mathrm{Enc}(pk_c; r)} \wedge \boxed{(sk_c, pk_c) \in \mathscr{K}} \wedge m)\}$$

Compiler for Pre-Constrained Group Signatures

<u>Let's start with a generic construction of Group Signatures</u>

Group Signature: $mpk = (vk_s, pk_s)$

- $ct = \mathrm{Enc}_{pk_s}(pk_c; r)$
- Simulation Extractable NIZK:
    A. I know a server signature σ on my public key pk_c [Only group members sign]
    B. I encrypted my public key pk_c using randomness r [Group manger can trace]
    C. I know the secret key sk_c corresponding to pk_c [Unforgeability]
    D. m is a tag in the NIZK

$$ct = \mathrm{Enc}(pk_c; r), \Pi = \{sk_c, r, \sigma \mid \boxed{\mathrm{Verify}_{vk_s}(pk_c, \sigma) = 1} \wedge \boxed{ct = \mathrm{Enc}(pk_c; r)} \wedge \boxed{(sk_c, pk_c) \in \mathscr{K}} \wedge m)\}$$

Compiler for Pre-Constrained Group Signatures

<u>Let's start with a generic construction of Group Signatures</u>

Group Signature: $mpk = (vk_s, pk_s)$

- $ct = \boxed{\mathrm{Enc}_{pk_s}(pk_c; r)} \longrightarrow$ **Pre-constrain here!**
- Simulation Extractable NIZK:
  A. I know a server signature σ on my public key pk_c
  B. I encrypted my public key pk_c using randomness r
  C. I know the secret key sk_c corresponding to pk_c
  D. m is a tag in the NIZK

$$ct = \mathrm{Enc}(pk_c; r), \Pi = \{sk_c, r, \sigma \mid \boxed{\mathrm{Verify}_{vk_s}(pk_c, \sigma) = 1} \wedge \boxed{ct = \mathrm{Enc}(pk_c; r)} \wedge \boxed{(sk_c, pk_c) \in \mathcal{K}} \wedge m)\}$$

Compiler for Pre-Constrained Group Signatures

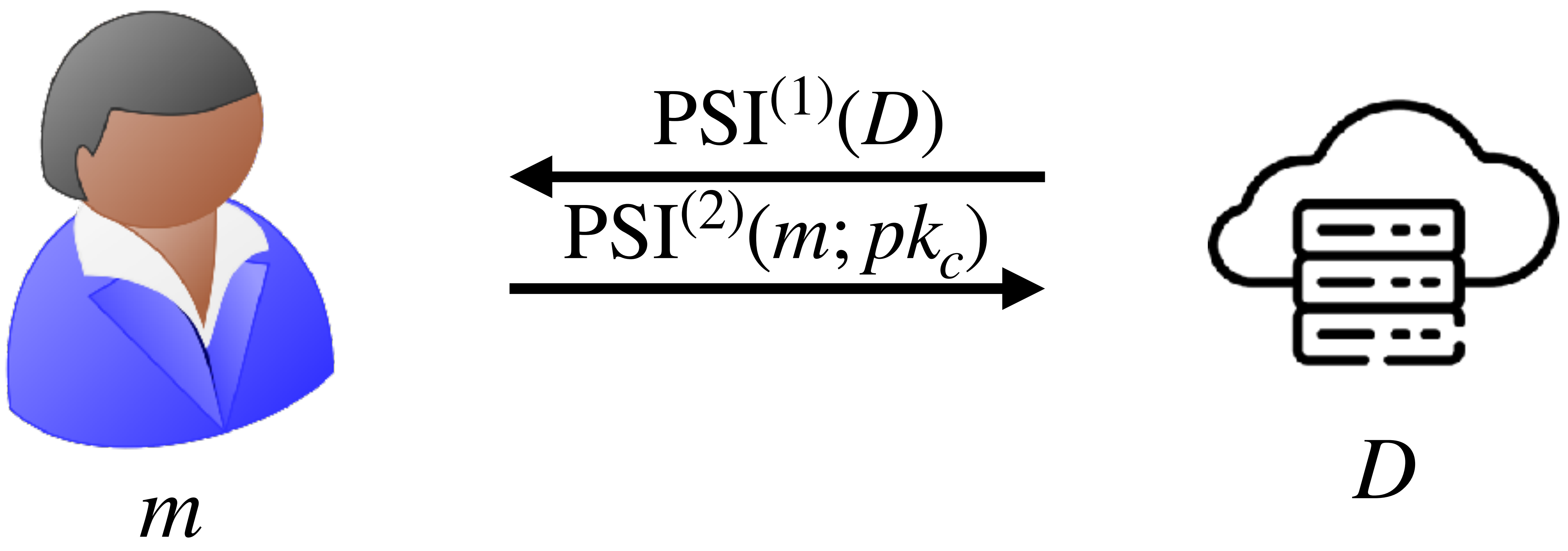
Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$

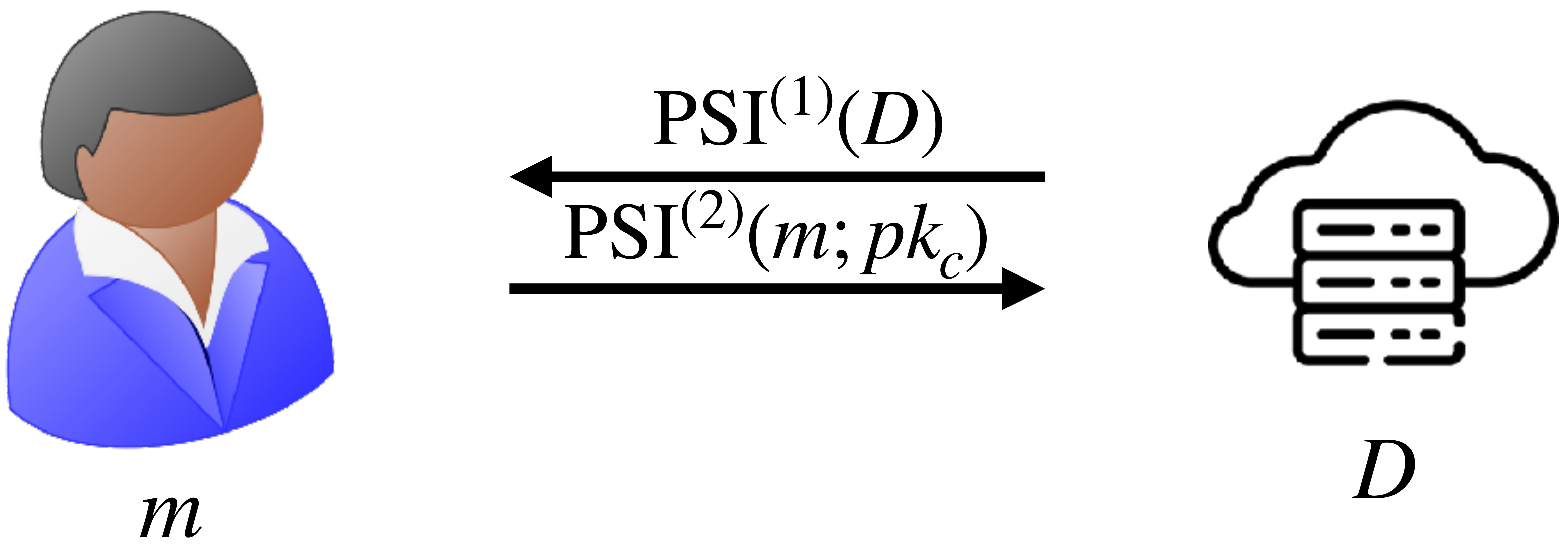


- Server learns pk_c if $m \in D$
- Two round — first round reusable
- Desirable to have $|ct| = O(1)$ and $T(\mathrm{PSI}^{(2)}) = O(1)$

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$

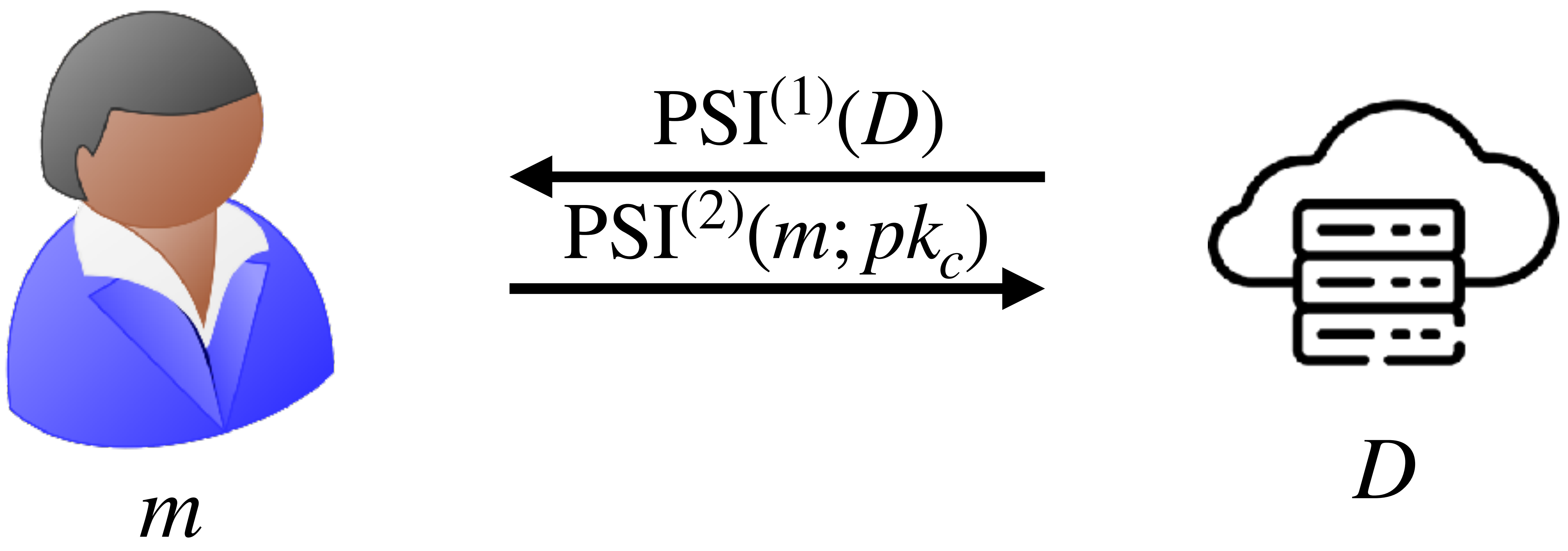


- $\boxed{\text{Server learns } pk_c \text{ if } m \in D}$ → **We achieved pre-constraining!!**
- Two round — first round reusable
- Desirable to have $|ct| = O(1)$ and $T(\mathrm{PSI}^{(2)}) = O(1)$

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$

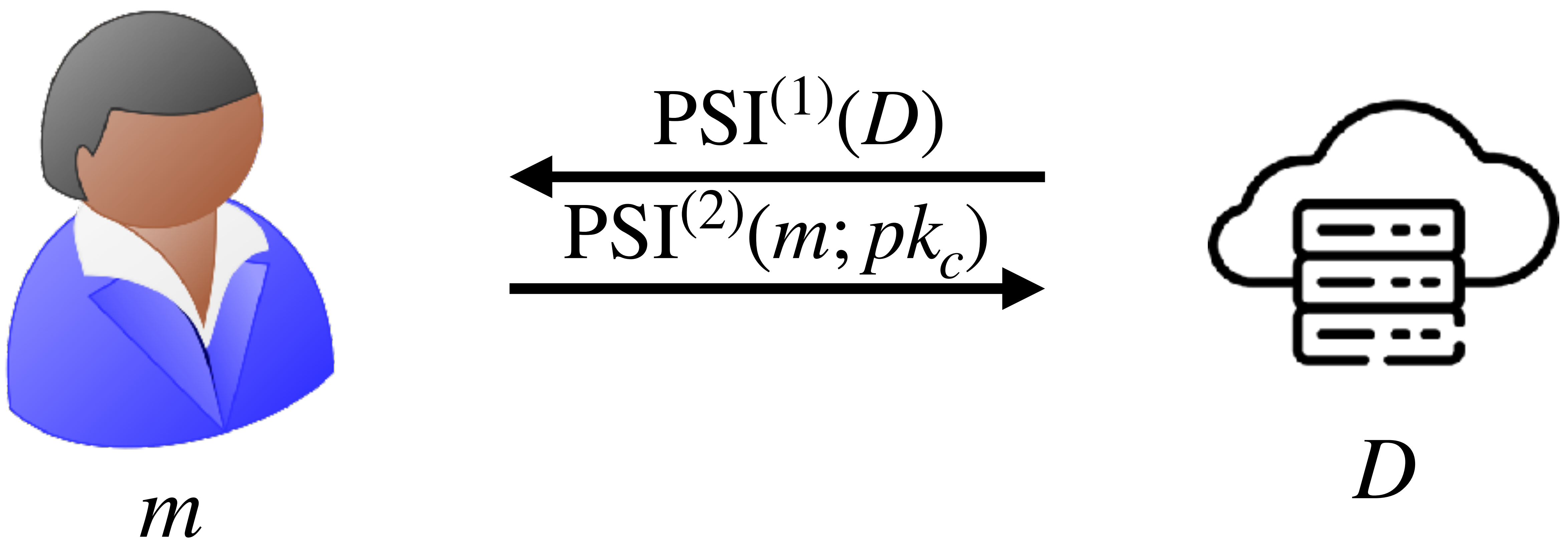


Do we have such a PSI scheme?

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$



Do we have such a PSI scheme?

Apple PSI [BDMTT21]

Caveat: $|mpk| = O(|D|)$

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$
- Simulation Extractable NIZK:
  A. I know a server signature σ on my public key pk_c
  B. ct was computed correctly
  C. I know the secret key sk_c corresponding to pk_c
  D. m is a tag in the NIZK

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}\ ct = \mathrm{PSI}^{(2)}(m; pk_c)$
- Simulation Extractable NIZK:
  A. I know a server signature σ on my public key pk_c
  B. ct was computed correctly
  C. I know the secret key sk_c corresponding to pk_c
  D. m is a tag in the NIZK

Final touches: Pick the right signature scheme and proof system.

Compiler for Pre-Constrained Group Signatures

Pre-Constrained Group Signature: $mpk = (vk_s, \cancel{pk_s}\ \mathrm{PSI}^{(1)}(D))$

- $\cancel{ct = \mathrm{Enc}_{pk_s}(pk_c; r)}$ $ct = \mathrm{PSI}^{(2)}(m; pk_c)$
- Simulation Extractable NIZK:
  A. I know a server signature σ on my public key pk_c
  B. ct was computed correctly
  C. I know the secret key sk_c corresponding to pk_c
  D. m is a tag in the NIZK

Final touches: Pick the right signature scheme and proof system.

We use structure preserving signatures + Groth-Sahai Proof System

How do we perform?

Structure preserving signatures + Groth-Sahai Proof System

Signing: ~10ms

Verification: ~40ms

Takeaways

- Constructions are exciting but take a step back.
- **Question the definition!**
- Talk to both sides of the debate. Need formal requirements.
- Being “secure” according to the “wrong” definition is meaningless.

Thank you!

eprint: 2022/1643