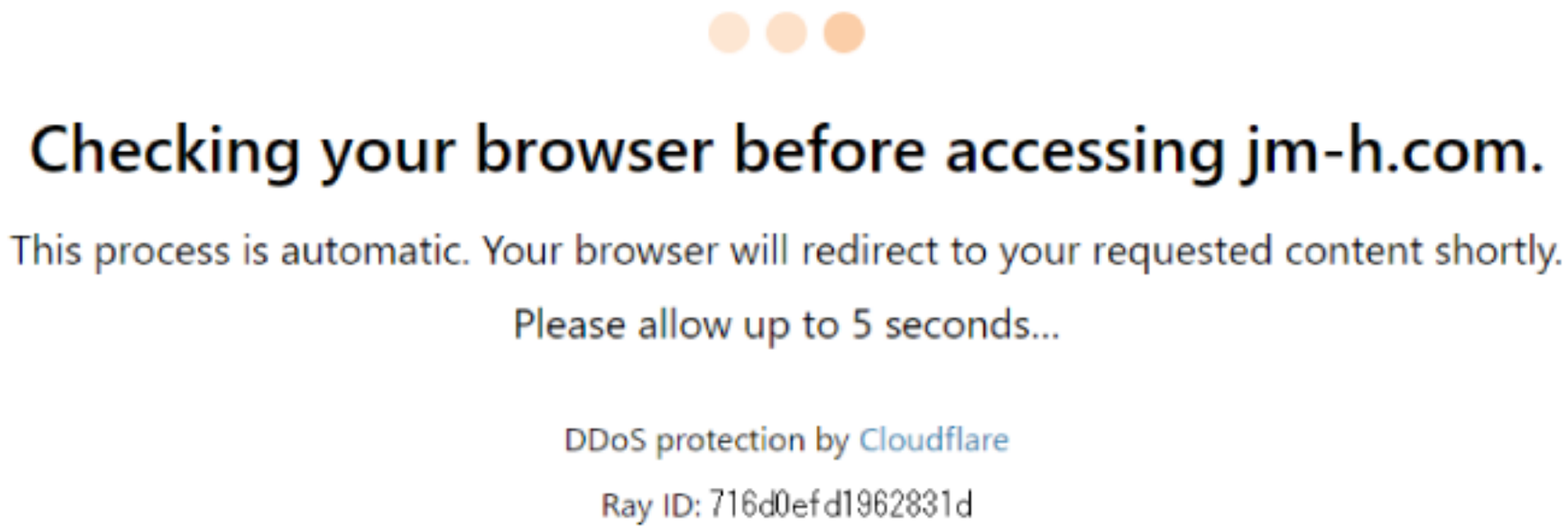
Lecture 16:
Authentication
(Pt 3: Anonyomous Credentials)

CPSC 4440/5440, 2026 Spring

Instructor: Fan Zhang

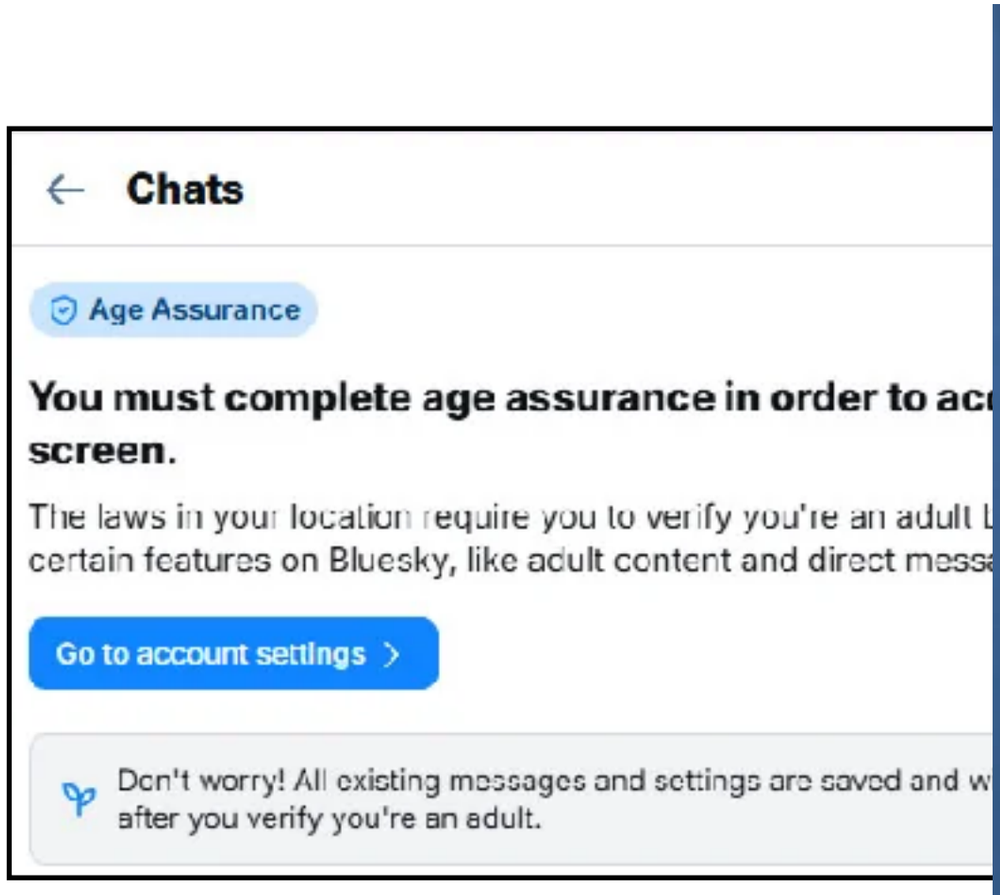
Many forms of authentication

- Some require **explicit logins**: with “something you {know|have|are}”, as we’ve seen.
- Some happens **implicitly**



Many forms of authentication

- Some allows you to create **pseduanonymous** accounts
- Some requires sensitive information (e.g., age , name, address)



The general trend

- Websites *used* to be happy to allow somewhat-anonymous usage
- Over the past couple of years this pattern has changed
  - sites like to collect data (e.g., knowing your identity makes you more lucrative as an advertising target)
  - push for legal age verification (required by 25 states in US and a dozen of other counties)
- The depressing future: users of *most sites* (e.g., even Wikipedia) will need to identify themselves by actual government ID.

How do we live in a world with routine age-verification and human identification, without completely abandoning our privacy?

Anonymous Credential

- Back in the 1980s, David Chaum foresaw our soon-to-be future, and he didn’t much like it.
- Long before the web or smartphones existed, Chaum recognized that users would need to routinely present (electronic) credentials to live their daily lives.
- He also saw that this would have enormous negative privacy implications.
- To address life in that world, he proposed a new idea: the **anonymous credential**.

SECURITY WITHOUT IDENTIFICATION: TRANSACTION SYSTEMS TO MAKE BIG BROTHER OBSOLETE

The large-scale automated transaction systems of the near future can be designed to protect the privacy and maintain the security of both individuals and organizations.

Non-anonymous authentication

- For Alice to prove she is over 18, an non-anonymous way is to reveal her government IDs.

Privacy Risks of non-anonymous authentication

- Privacy-cautious sites may delete Alice's ID after processing but no way to enforce.
- Curious sites (and ad networking) always probably know a lot about who we are and what we're browsing through tracking.
- They can now precisely link Alice's browsing history on the site to an actual human in the world.
- As AI bots become the main users of Internet, the value of precise human identity is huge, because bots won't read ads!
- Also think about the implications of security breaches.

Chaum's idea

- Breaking the linkage between the **issuance** and **usage** of a credential
- When Alice shows her credential to the website, all the site learns is that Alice has been given a valid credential. Not who issued the credentials, not other credentials associated with her ID.
- hold even if the website *colludes* with the issuer of the credentials.

Analogy: wristbands

3/4" Tyvek Wristbands Pre-Printed Over 21

(17 reviews)

$7.00

Shipping calculated at checkout

Pack Count: 100

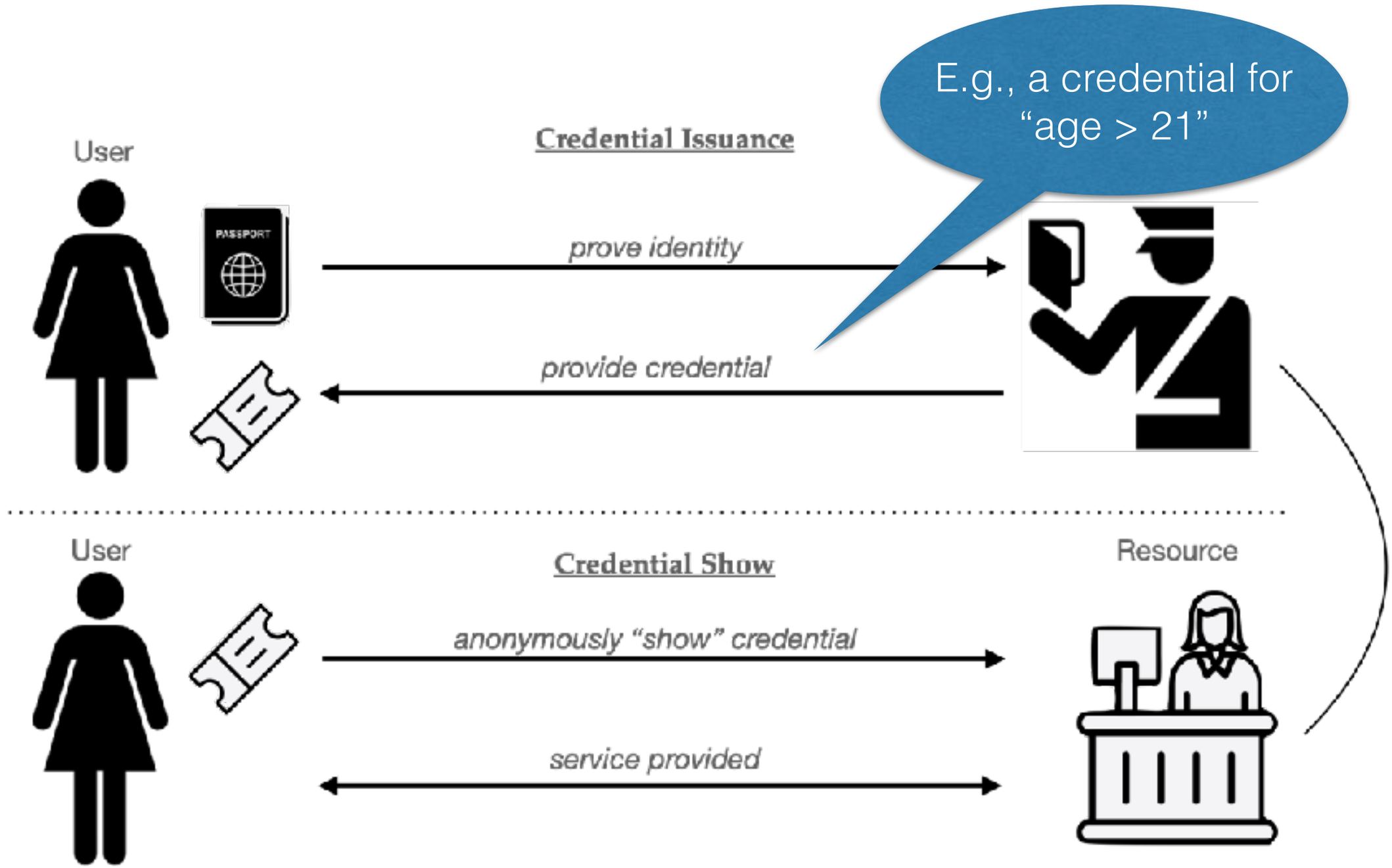
100

Color: Neon Blue

Neon Blue

Add to Cart

Wristbands are easy to “forge”.
We need cryptography.



https://blog.cryptographyengineering.com/2026/03/02/anonymous-credentials-an-illustrated-primer/

Strawman solution

- Idea: Issue anyone older than 21 a *deterministic* signature
  $\text{Sig}(sk_{\text{Issuer}}, \text{age} \geq 21)$
- Nice properties
  - Forgery is prevented by signatures
  - Unlinkability as credentials are identical
- Obvious problems?

Preventing duplication

- **Single-use (one-show) credentials**: each credential can be *shown* exactly one time.
  - If a user wants to access the website fifty times, then she needs to obtain fifty separate credentials from the Issuer.
  - A hacker is limited to only a bounded number of website accesses.
  - This approach is used by credentials like PrivacyPass, which is used by sites like CloudFlare.

Preventing duplication

- Revocable credentials:
  - when a particular anonymous user does something bad (posts spam, runs a DOS attack against a website), the website can ban that specific user's credential — blocking future usage of it, without otherwise learning who they are.
- Challenge: credentials are unlinkable!

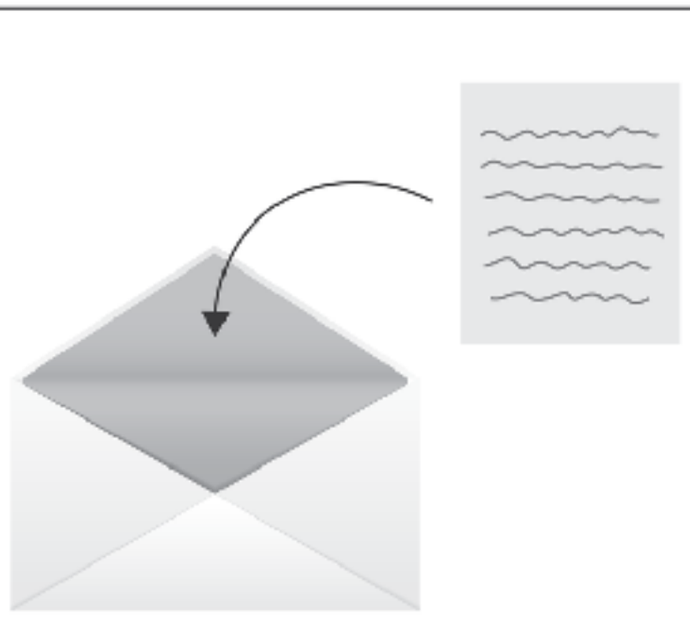
Preventing duplication

- Hardware-tied credentials
  - “bind” credentials to a piece of hardware, such as the **secure enclave** in Android phones. Used by real-world proposals like Google’s LongFellow ZK.
  - A hacker will need to “crack” the hardware to clone the credentials.
  - But a single security breach into secure hardware can have big consequences that can undermine the security of the *whole* system.

Single-use credentials

Chaumian credentials

- Blind signatures



Seal a message with carbon copy paper

Sign the envelope pressing into the carbon paper

Signs the message without revealing content

Chaumian Credentials from Blind Signatures

- Setup: Issuer samples (sk, pk) for blind signature
- Issuer:
  - Alice presents ID to Issuer (e.g., proving age over 21).
  - Alice chooses a random serial number sn
  - Bank **blindly** signs over sn and returns $\sigma = \text{BSign}(sk, sn, \text{age} \geq 21)$
- Show:
  - Alice presents (sn, σ) to website
  - Website stores sn to prevent “double spending” or duplicating
- Variants of it still used after 40 years!

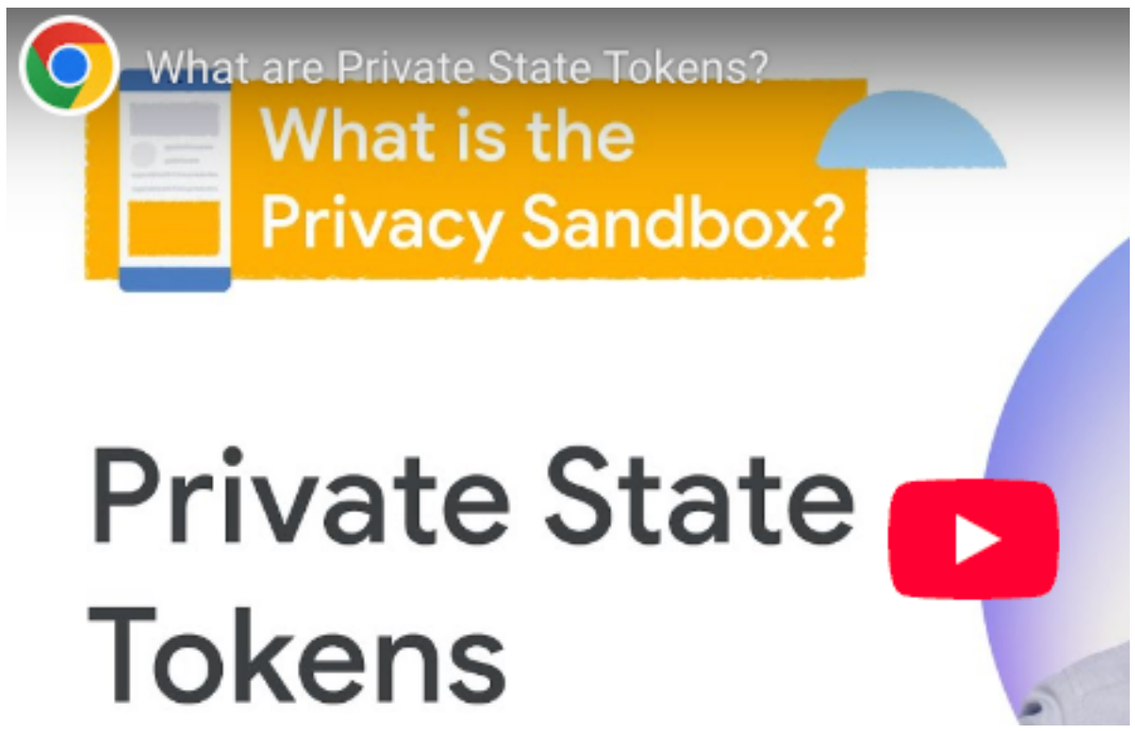
DE GRUYTER OPEN Proceedings on Privacy Enhancing Technologies ; 2018 (3):1–17

Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda

Privacy Pass: Bypassing Internet Challenges Anonymously

Privacy Pass

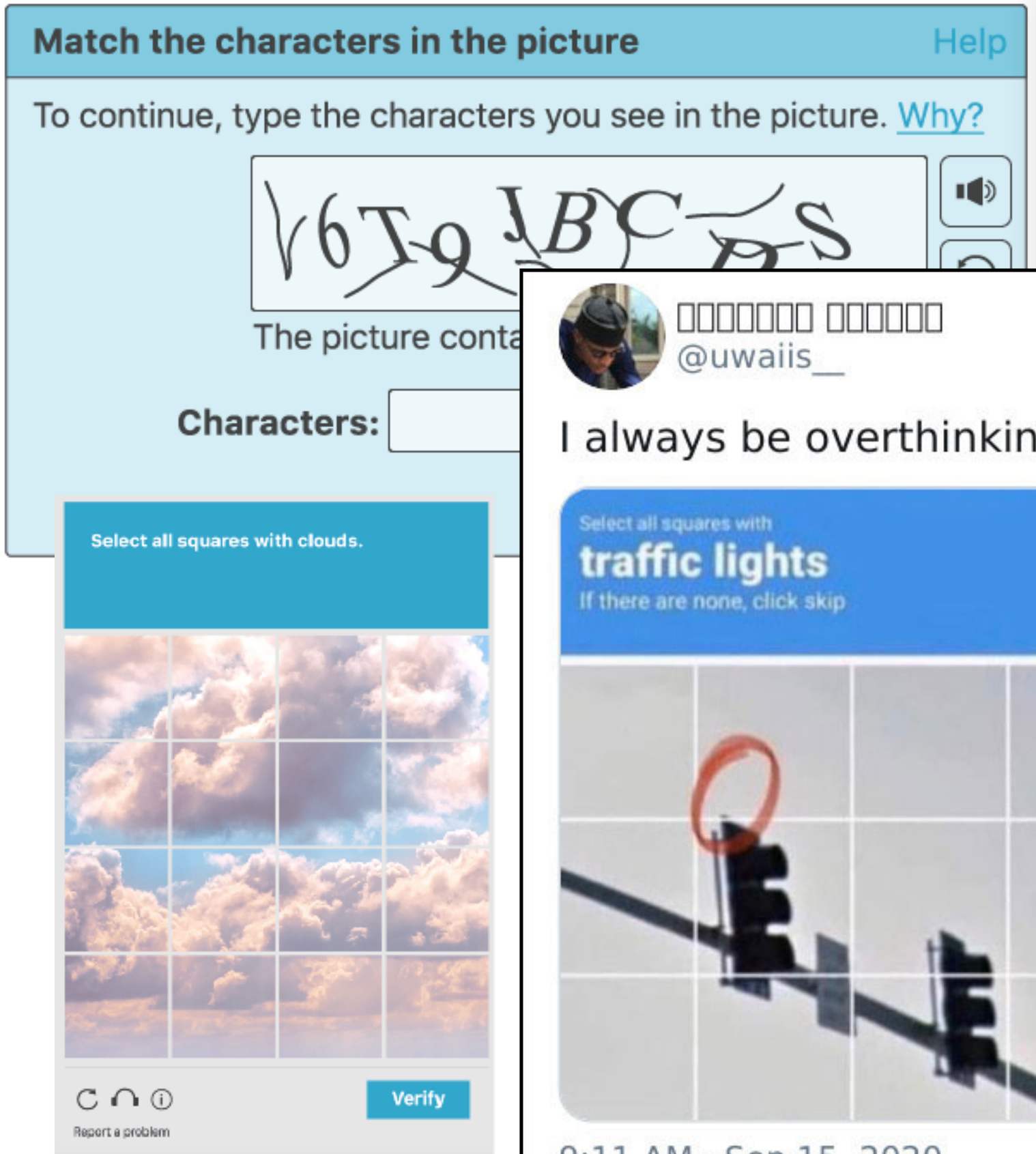
CLOUDFLARE

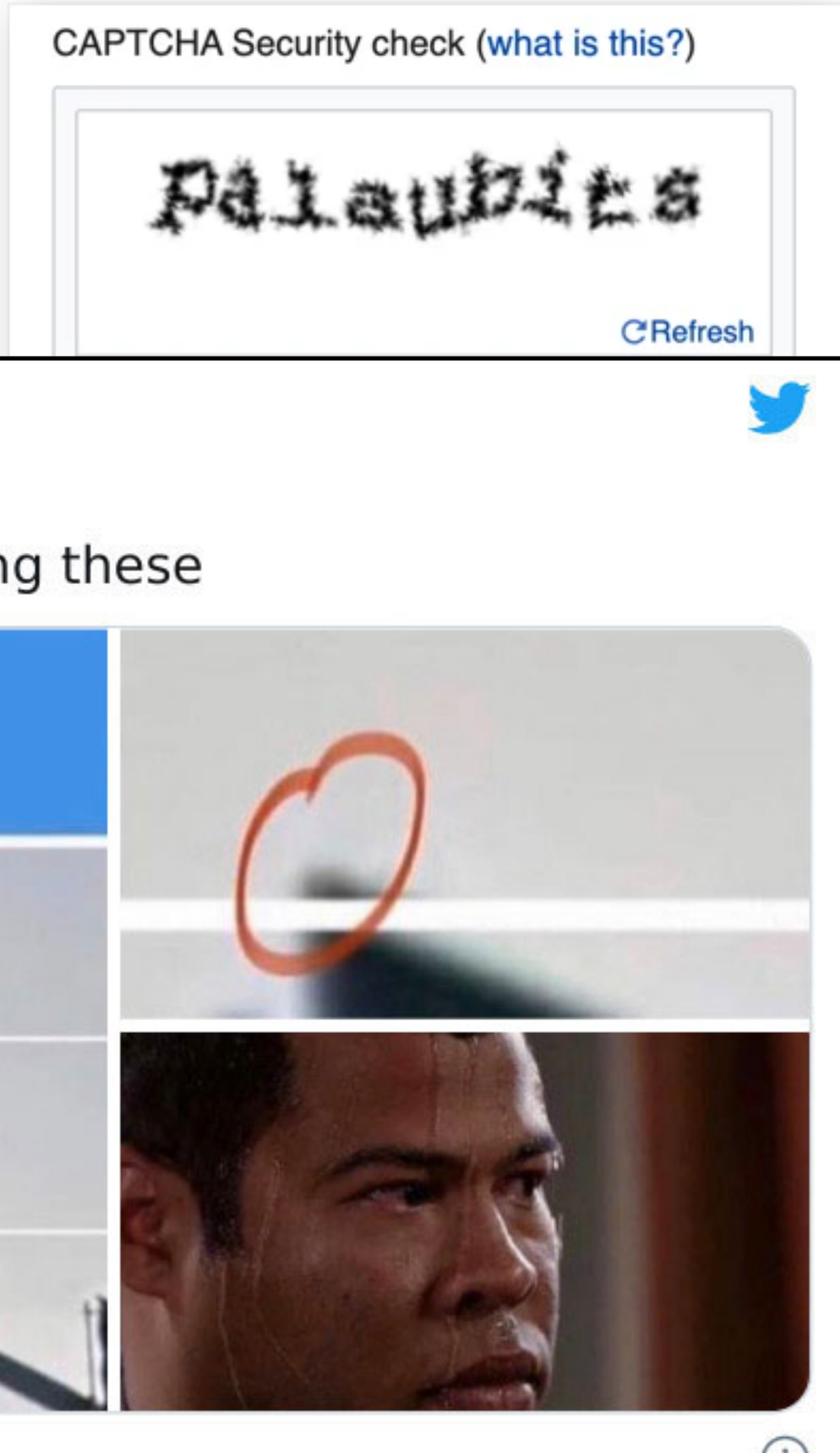


Motivation

- CDN serves 71% of web traffic
- It also protects websites from DDoS attacks by IP-based *reputation* check
- IP with poor reputation may get blocked, …
- … or get asked to perform proof-of-human checks (captcha)

CAPTCHA: Completely Automated Public Turing test to tell Computers and Humans Apart."

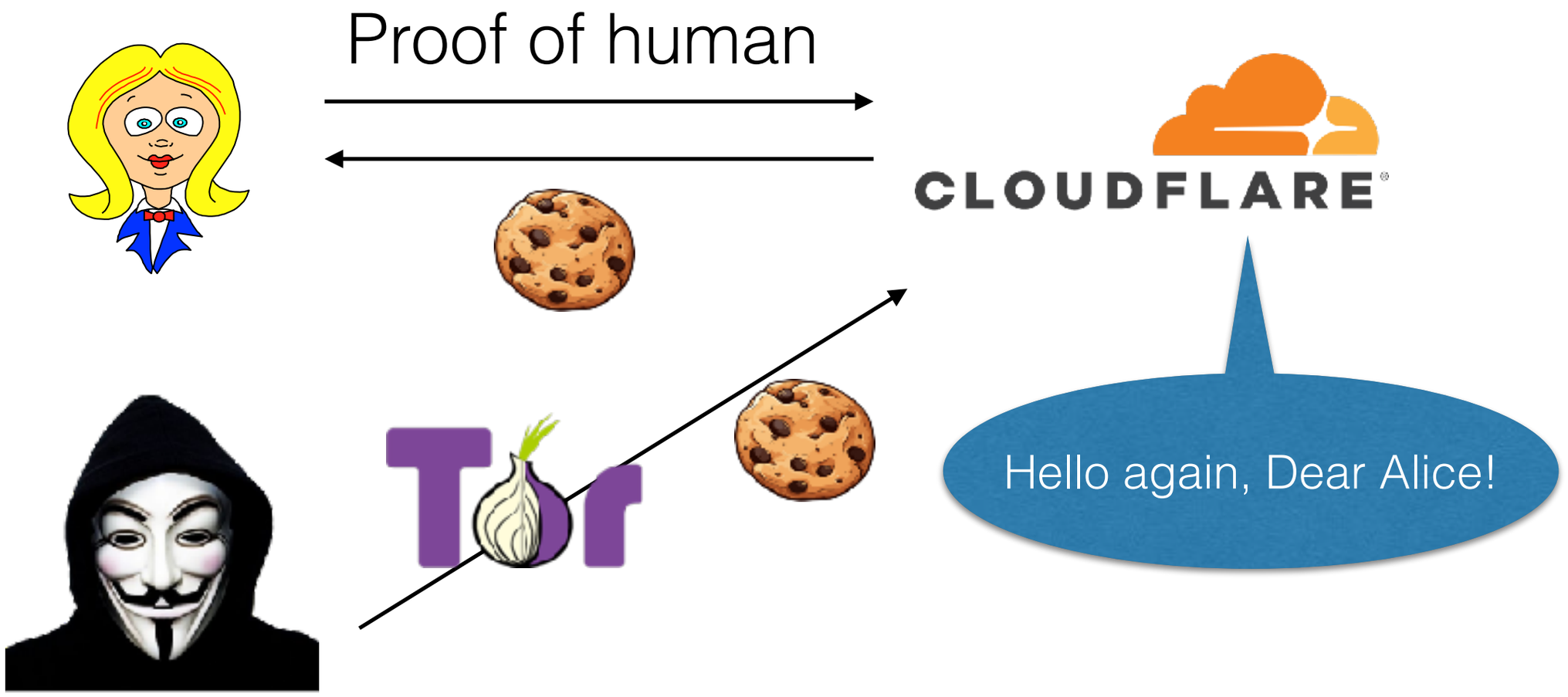




issues with captcha

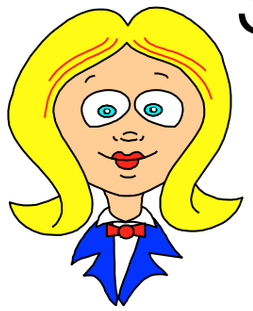
- It’s not clear that they block bots/AI, but they annoys human users for sure.
- It’s an **accessibility barrier** to people with disabilities.
- Particularly affects users of VPN, Tor, proxy, etc. as they tend to use short-lived IP addresses
  - They already have a hard time using the Internet

A anonymous credential problem



- Grant user “access tokens” for future uses
- Problem: token issuer & consumer are the same party! They can track users.

Private Pass



Some proof-of-human

Tokens

Issuer

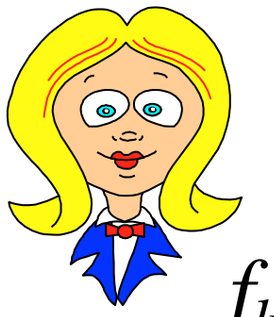
Verifier

“I have a valid token”

- 1) users are **issued** tokens after verification;
- 2) user can later **redeem** tokens;
- 3) No **linkable** between issued and redeemed tokens

Blind Sig vs VORPF

[x's are random values]



$x_1, \cdots, x_N$

k

VORPF

$f_k(x_1), \cdots, f_k(x_N), \pi$

[π: proof that k is used]

$f_k(x_i)$

[redeem]

$Y = g^k$

Issuer

Verifier

- Blind Sigs can be directly used (can be slow)
- They instead leverages the fact that issuer and verifier **have access to the same key k**

Verifiable OPRF

- ORPF + Verifiability
- Sever holds a “private” key k, publishes a “public key” $Y = g^y$
- User gets $f := H(x)^k$ and a proof π // other OPRF possible
- What should π prove to convince user of the *correctness*?

Observation: π just need to prove that $\mathsf{DL}(f) = \mathsf{DL}(Y)$

DLEQ (Discrete Log Equivalence) Proofs

- Let $\mathbb{G}$ be a group of prime order q, and g, h be two generators.
- Denote DLEQ(g, P, h, Q): I know x such that $P = g^x, Q = h^x$
- Equivalently, (h, P, Q) is a DH-triple (of form g^a, g^b, g^{ab})
- First protocol given by Chaum-Pedersen (’92)
  - Two Schnorr’s identification protocol combined

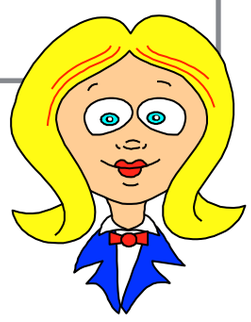
Recall: Schnorr's identification

I

Private / Public Keys:
$(x, y = g^x)$

r

Alice's public key: *y*



One-time private key: k

$s = rx + k$

Blinding factor: $I = g^k$

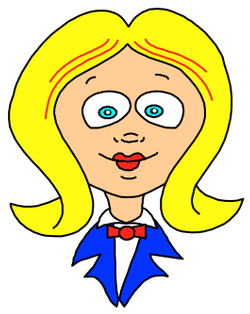
$$g^s \stackrel{?}{=} y^c \circ I$$

DLEG (Discrete Log Equivalence) Proofs

Public knowledge: g, P, h, G

A, B

$x \in \mathbb{Z}_q$ s.t. $P = g^x, Q = h^x$



c

One-time private key: k

$s = xc + k$

$g^s \stackrel{?}{=} P^c \circ A$ and
$h^s \stackrel{?}{=} Q^c \circ B$

Blinding factors: $A = g^k, B = h^k$

Special zero-knowledge

- Given (g, P, h, Q), anyone can simulate *transcripts* without knowing x
  - Choose uniform $c, s \leftarrow^{\$} \mathbb{Z}_q$
  - Compute $A = g^s / P^c, B = g^s / Q^c$
  - (A, B, c, s) is distributed identically as a "real" transcript

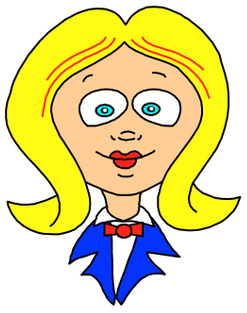
Special soundness

- Given (g, P, h, Q), if an adversary can answer two challenges, then she can recover the secret.
  - Suppose (A, B, c, s) and (A, B, c', s') are both valid
  - I.e., $g^s = P^c \circ A$ and $h^s = Q^c \circ B$
  - I.e., $g^{s'} = P^{c'} \circ A$ and $h^{s'} = Q^{'c} \circ B$
  - => $\log_g P = log_h Q = \frac{s - s'}{c - c'}$

Non-interactive version

Public knowledge: g, P, h, G

$x \in \mathbb{Z}_q$ s.t. $P = g^x, Q = h^x$



- Sample one-time key: $k \in \mathbb{Z}_q$
- Compute $A = g^k, B = h^k$
- $c = H(g, h, P, G, A, B)$
- $s = xc + k$
- Output $\pi_k = (c, s)$

Verification of (c, s)

- $A = g^s / P^c$
- $B = h^s / Q^c$
- Output $c \stackrel{?}{=} H(g, h, P, Q, A, B)$

Sigma protocols (a slight detour)

- Σ protocols are an *incredibly* useful class of protocols for non-interactive zero-knowledge proof of knowledge.

- Countless applications
  - Schnorr's identification protocol
  - DLEG
  - two El Gamal ciphertexts encrypt the same value
  - an El Gamal ciphertext and a Pedersen commitment hide the same value
  - Arbitrary linear relationship $g_1^{x_1} g_2^{x_2} = u_1 \wedge g_3^{x_1} g_4^{x_2} = u_2$
- Efficient!

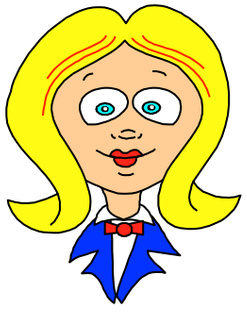
Batched DLEQ Proofs

- How to prove $P = g^x, Q_i = h_i^x$ for $1 \leq i \leq m$?
- Idea: random combination
- for random $c_1, c_2, \cdots, c_m$ in $\mathbb{Z}_q$ …
- Let $h = \prod h_i^{c_i}$ and $Q = \prod Q_i^{c_i}$
- if $h^x = Q$, then $h_i^x = Q_i$ for all $i \in [1,m]$ except for negligible probability
  - Suppose $h_m^{k'} = Q_m$ and $h_n^{k''} = Q_n$ for some $k', k'' \neq k$.
  - To pass verification, $(c_m + c_n)k = c_m k' + c_n k''$
  - For any c_m, only one c_n works, chosen w.p. $1/q$ (negligible)

Batched DLEQ Proofs

- To prove DLEQ(g, P, h_i, Q_i) for many i
- Prover
  - Derives $c_1, \cdots, c_m$ from $H(g, P, \{h_i\}, \{Q_i\})$
    - computes $\hat{h} = h_1^{c_1} \cdots h_m^{c_m}$ and $\hat{Q} = Q_1^{c_1} \cdots Q_m^{c_m}$
    - Computes $\hat{\pi} =$ DLEQ$(g, P, \hat{h}, \hat{Q})$
- Verifier
  - Redo first two steps to compute $\hat{h}, \hat{Q}$; verify $\hat{\pi}$
- Denoted mDLEQ$(g, P, \{h_i, Q_i\}_{i=1}^m)$

VOPRF (HashDH)



User

Public Key $Y = g^x$

Server

Inputs $x_1, x_2, \cdots, x_N$

$X_i = H(x_i)^{r_i}$

$\xrightarrow{X_1, X_2, \cdots, X_N}$

$f_i = X_i^k$

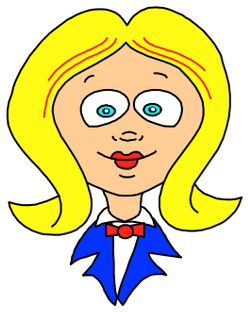
$\pi = \text{mDLEQ}(g, Y, \{X_i, f_i\})$

$\xleftarrow{f_1, f_2, \cdots, f_N, \pi}$

$t_i = f_i^{1/r_i}$

$t_i = H(x_i^k)$

Privacy Pass



$x_1, \cdots, x_N$

$t_1, \cdots, t_N, \pi$

VORPF

k

$Y = g^k$

Server

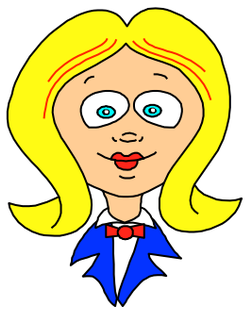
Verifier

- Let R be request
- $K_m = H(x_i, t_i)$
- $T = \mathsf{Mac}(K_m, R)$

(x_i, R, T)

- Verifier checks the MAC (after deriving keys from k)
- Stores x_i to prevent “double-redeeming”

Security Properties



$x_1, \cdots, x_N$

$t_1, \cdots, t_N, \pi$

VORPF

k

$Y = g^k$

Server

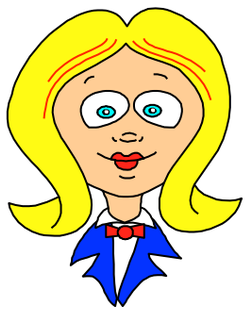
Verifier

- Let R be request
- $K_m = H(x_i, t_i)$
- $T = \text{Mac}(K_m, R)$

(x_i, R, T)

- **Unlinkability**: $H(x_i)^{r_i}$ is a random group element, containing no information about $H(x_i)$ or x_i

Security Properties



$x_1, \cdots, x_N$

$t_1, \cdots, t_N, \pi$

VORPF

k

$Y = g^k$

Server

Verifier

- Let R be request
- $K_m = H(x_i, t_i)$
- $T = \text{Mac}(K_m, R)$

(x_i, R, T)

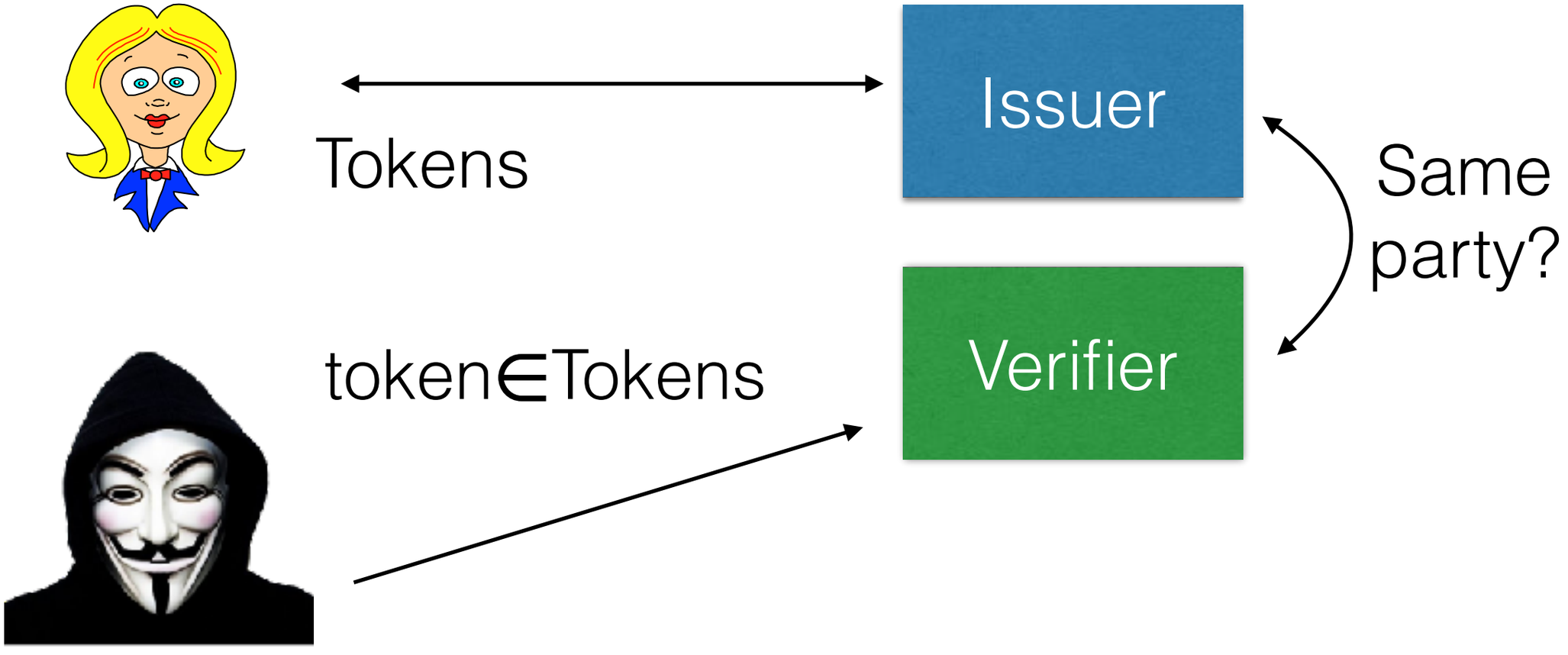
- **One-more-token unforgeability**: user can't get one more token given $t_1, \cdots, t_N$ (reduction to El Gamal)

Applications of Privacy Pass

- CAPTCHA Reduction (the original motivation)
- Anonymous telemetry (DIT proposal by Facebook)
- Anonymous Ticketing
- Rate Limiting Without Tracking
  - E.g., a user can only speak 10 times a day in an anonymous forum
- Paywalls and Micropayments with Anonymity
  - E.g., each token grant access to an article read anonymously
- Privacy Preserving Single-Sign-on (SSO)
  - If a user “passes” a challenge on Service A, they can redeem tokens on Service B—without the services being able to track.
- …

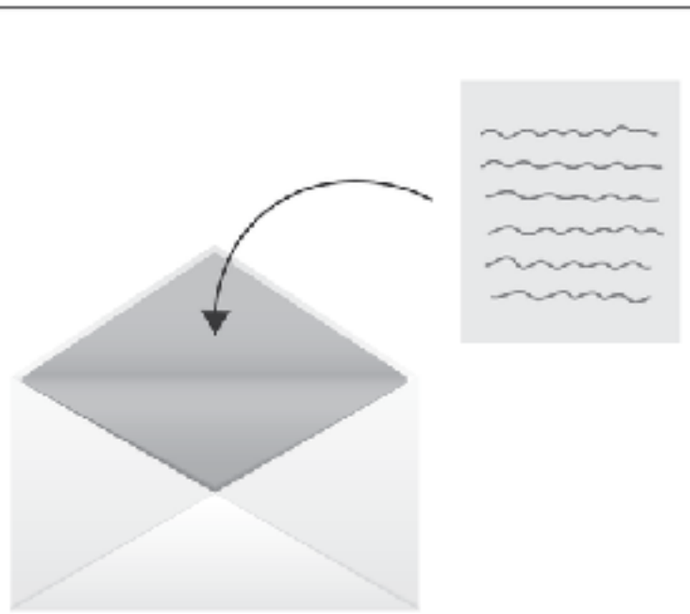
Note: Public verifiability v.s. designated-verifier

- “Privately verifiable” is I and V are the same party
- Privacy Pass we saw is **privately verifiable**
- Publicly verifiable is desirable in many applications



Blind signatures is publicly verifiable!

- Though slower



Seal a message with carbon copy paper

Sign the envelope pressing into the carbon paper

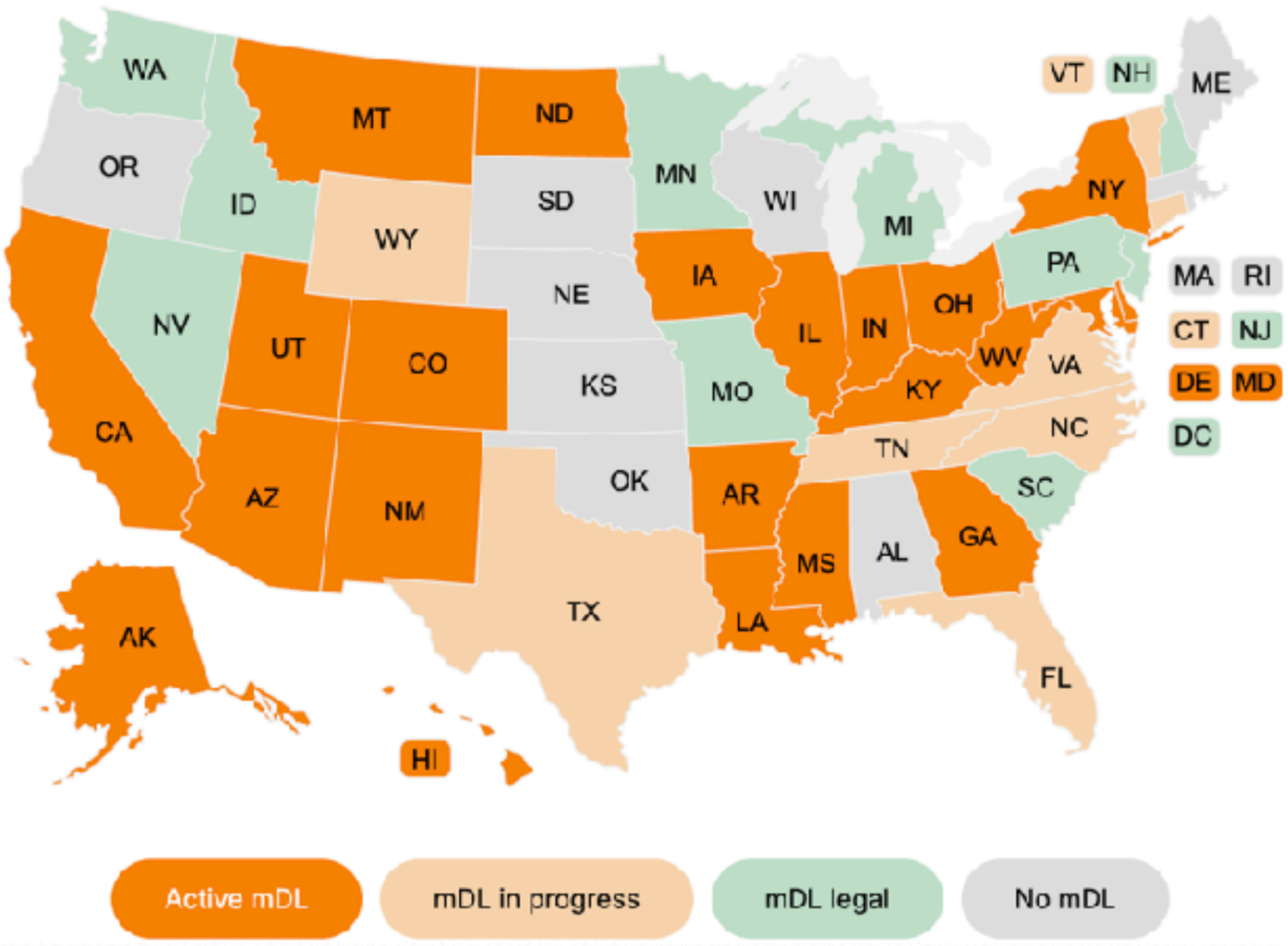
Signs the message without revealing content

Limitations of Chaumian tokens

- They don’t carry much information!
  - They carry exactly one bit information: the endorsement of issuer
  - E.g., age over 21 v.s. not
  - E.g., this age credential is useless is the user is asked to prove age over 25.

A very expressive approach

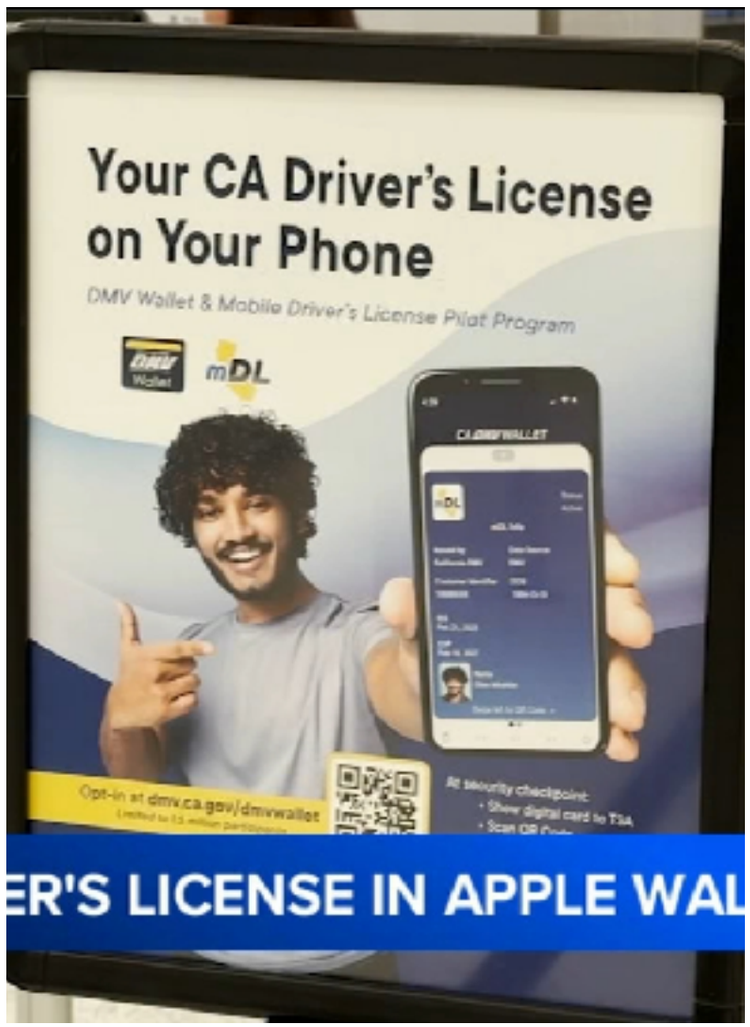
- Ingredient 1: signed identity documents (e.g., mobile Driver's License, mDL)



https://idscan.net/mobile-drivers-licenses-mdl-state-adoption/

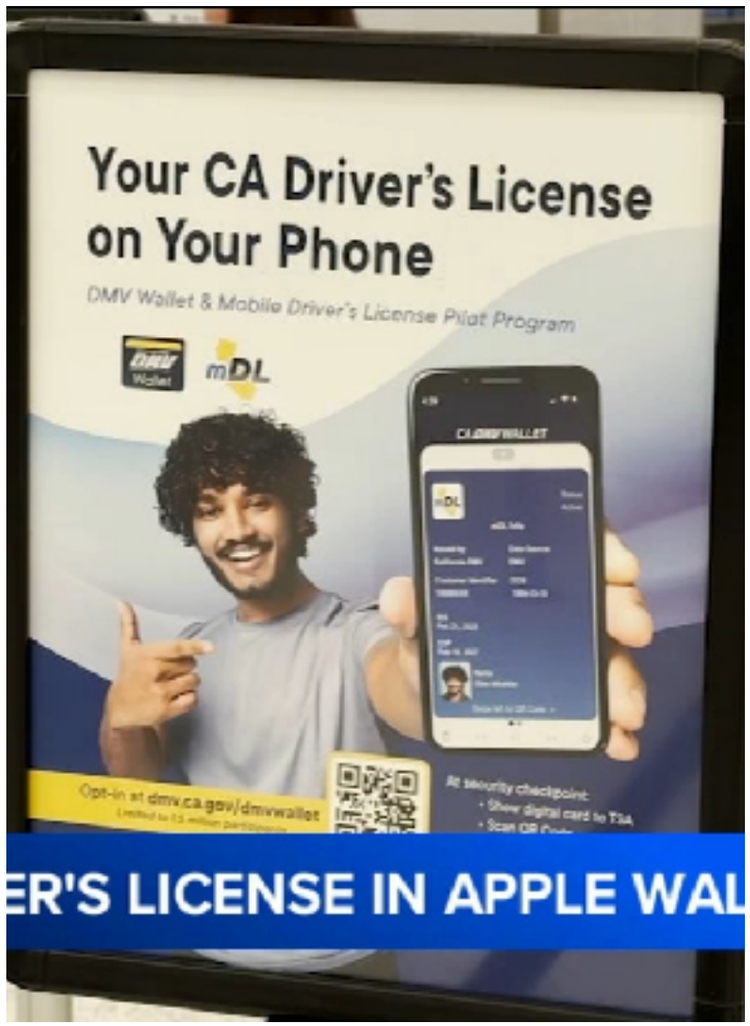
A very expressive approach

- Ingredient 1: signed identity documents (e.g., mobile Driver’s License, mDL)




```
{
   "issuer": "US-CT-DMV",
   "subject": {
      "given_name": "Adam",
      "family_name": "Smith",
      "birth_date": "1990-04-12",
      “home_addres": “123 Apple St",
      "issuing_state": "CT",
      "issuing_country": “US"
   },
   “signature”: “0xdeadbeef”
}
```

A very expressive approach



- Ingredient 2: general purpose zero-knowledge proofs (e.g., zkSNARKs)
- E.g., proof of knowledge of an ID s.t.
  - ID.BIRTHDATE <= (TODAY – 25 years)
  - ID.ISSUE_STATE != NY
  - ID.ISSUE_COUNTRY = US
  - Verify(ID.SIGNATURE, some known state DMV public key) = True
- This proves I’m over 25, and not a resident in NY.
- Selective disclosure

A very expressive approach

Longfellow ZK

License Apache 2.0 | eprint 2024/2010 | IETF Draft draft-google-cfrg-libzk

Overview

The Longfellow library enables the construction of zero-knowledge protocols concerning legacy identity verification standards such as the ISO MDOC standard, the JWT standard, and W3 Verifiable Credentials. This implementation is described in:

- Anonymous credentials from ECDSA
- libzk: A C++ Library for Zero-Knowledge Proofs
- Project documentation

It is named after the bridge outside the Google Cambridge office.

- https://github.com/google/longfellow-zk

Identity theft becomes catastrophic

- What happens if a hacker steals one of these re-usable credentials?
- This is catastrophic: a single stolen credential anywhere means that the guarantees of the global system become useless.
  - E.g., if my ID is stolen, then anyone can pretend to be “over 25, and not a resident in NY.”
  - What if someone sells their IDs on the dark web?
  - Even better, one can start a website to sell *proofs*, instead of identities.

Preventing identity theft

- **Google's hardware-based solution**: put secrets in a "Secure Element" on Android phones
  - owners of identity doc cannot "see" the identity in plaintext; they can only use it through pre-defined APIs
- Catch: There are hundreds of millions of phones in the world, and the Secure Element technology in them runs the gamut from "very good" (for high-end, flagship phones) to "modestly garbage" (for the cheap burner Android phone you can buy at Target.)

https://blog.cryptographyengineering.com/2026/03/02/anonymous-credentials-an-illustrated-primer/

Idea: rate limit

```
{
  "issuer": "US-CT-DMV",
  "subject": {
    "given_name": "Adam",
    "family_name": “Smith",
    ...
  },
  “user_key”: PK_Adam,
  “signature”: “0xdeadbeef”
}
```

- Idea: user generates serial numbers from sk. Websites store all SNs.
- E.g., N-show tokens
  - $sn_i = \text{PRF}(sk, i)$ for some $i \leq N$.
- Can be modified to support daily limit, say.

Idea: expiration

```
{
  "issuer": "US-CT-DMV",
  "subject": {
    "given_name": "Adam",
    "family_name": “Smith",
    ...
  },
  “valid_until”: T,
  “signature”: “0xdeadbeef”
}
```

Idea: ban

- Already happens all the time on real sites: for example, when a user posts spam on a website, or abuses the site’s terms of service.
- Difficult when users are anonymous: ban whom??!
- In a non-anonymous credential system we simply identify the user and add them to a banlist. But anonymous credential users are anonymous! How do you ban a user who doesn’t have to identify themselves?

Idea: ban

- Login:
  - Website generates a random s
  - Ask for $SN' = \text{PRF}(sk, s)$ in addition to credentials and proofs
  - If this user is to be banned, add (s, SN') to S
- Going forward, ask every user to show $\text{PRF}(sk, s_i) \neq SN_i$, for all i, using this user's key.

Summary of big ideas

- **Main idea**: Separating issuance & usage
- **Always ask**: how is duplication prevented?
  - duplication by thefts, or by legit owners
- **Typical solutions**: 1) One-show (or N-show), 2) Revoke / ban, and 3) Hardware.
- **Privacy Pass** is one of the most widely used anonymous credentials in practice.
- **Google's Longfellow** is a ZK-based multi-show scheme from 2025, so the adoption is growing!

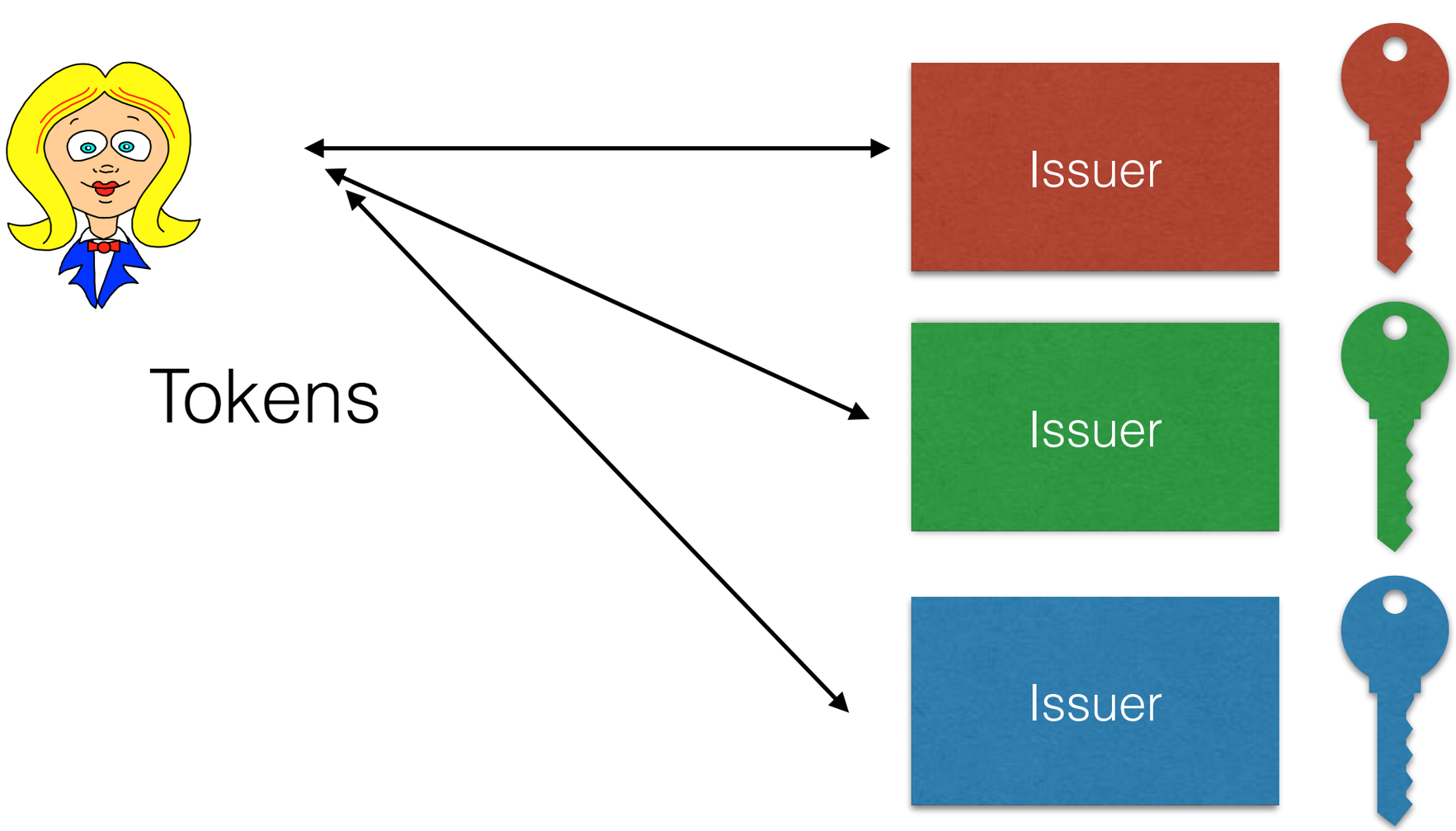
Recent academic papers (skipped)

Decentralized issuance

- Single point failure if Issuer’s key is compromised.

Decentralized issuance

- Token issuance requires t of n issuers (E.g., 2 out of 3)
- Dynamic issuers: issuers may leave and join (threshold OVRF doesn’t fit the order)



Blind Multisignatures for Anonymous Tokens with Decentralized Issuance

Ioanna Karantaidou*
George Mason University
Fairfax, VA, USA
New York University
New York, NY, USA
ikaranta@gmu.edu

Omar Renawi*
CISPA Helmholtz Center for Information Security and Saarland University
Saarbrucken, Germany
omar.renawi@cispa.de

Foteini Baldimtsi
George Mason University
Fairfax, VA, USA
Mysten Labs
New York, NY, USA
foteini@gmu.edu

Nikolaos Kamarinakis
University of Maryland
College Park, MD, USA
Common Prefix
Athens, Greece
k4m4@umd.edu

Jonathan Katz
Google
Washington DC, USA
University of Maryland
College Park, MD, USA
jkatz2@gmail.com

Julian Loss
CISPA Helmholtz Center for Information Security
Saarbrucken, Germany
loss@cispa.de

Abstract

We propose the first constructions of anonymous tokens with decentralized issuance. Namely, we consider a dynamic set of signers/issuers; a user can obtain a token from any subset of the signers, which is publicly verifiable and unlinkable to the issuance process. To realize this new primitive we formalize the notion of blind multi-signatures (BMS), which allow a user to interact with multiple signers to obtain a (compact) signature; even if all the signers collude they are unable to link a signature to an interaction with any of them. We then present two BMS constructions, one based on BLS signatures and a second based on discrete logarithms without pairings. We prove security of both our constructions in the Algebraic Group Model. We also provide a proof-of-concept implementation and show that it has low-cost verification, which is the most critical operation in blockchain applications.

1 Introduction

In the digital world, *authorization* plays a foundational role. From regulating access to online services to ensuring the integrity of voting systems, effective access-control mechanisms are crucial for maintaining security and trust. User authorization can be implemented via various methods depending on the application scenario. Common approaches include using credentials such as usernames and passwords or relying on third-party services, such as Auth0 or OpenID, to access user accounts. However, these authentication methods raise concerns regarding user privacy. Each time a user logs in, the service learns everything about the user's activities, enabling the creation of a full profile of their habits. While this level of information leakage may be necessary for certain applications, in many other cases it is desirable to avoid it. Consider, for instance, a subscription-based news portal. In a privacy-friendly world, the only thing that the service should learn is *whether* the user has a

Published in ACM CCS'24

Background on **BLS signature**

- KeyGen: $sk \leftarrow^{\$} \mathbb{Z}_q, pk = g^{sk}$
- Sign(sk, m): $H(m)^{sk}$
- Verify(pk, m, sig): Check if $(pk, H(m), \sigma)$ is a DH triple
  - How? DLEQ proof can be attached
- How to make signature short?

Background on **BLS signature**

- Boneh, Lynn, Shacham (’01) made use of a new tool: bilinear pairing
  - For certain groups $\mathbb{G}$, $\mathbb{G}_T$, there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$
    - For any a, b, we have $e(g^a, g^b) = e(g, g)^{ab}$
- Pairing can check if (g^a, g^b, g^c) is a DH triple
  - Simply: $e(g^a, g^b) \stackrel{?}{=} e(g^c, g)$

Background on **BLS signature**

- System parameter
  - $\mathbb{G}$ is a group of order q
  - $g \in \mathbb{G}$ a generator
  - a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$
    - $H(x)$: a hash mapping to group elements in $\mathbb{G}$
    - (In practice, Type-3 pairings are used $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$)
- KeyGen: $sk \leftarrow^{\$} \mathbb{Z}_q, pk = g^{sk}$
- Sign(sk, m): $H(m)^{sk}$
- Verify(pk, m, sig):$e(\sigma, g) \stackrel{?}{=} e(H(m), pk)$

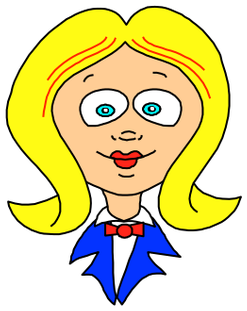
Background on **BLS signature**

- Many BLS sigs on the same message can be combined
- $\mathsf{Agg}(\{\sigma_i\}, \{pk_i\}, m) \rightarrow (\sigma_{\mathsf{agg}}, pk_{\mathsf{agg}})$
  - assert $\mathsf{Verify}(\sigma_i, pk_i, m) = 1$
  - Compute $c_1, \cdots, c_n \leftarrow H(\{pk_i\})$
  - $\sigma_{\mathsf{agg}} = \prod_i \sigma_i^{c_i}$
  - $pk_{\mathsf{agg}} = \prod_i pk_i^{c_i}$
- Verify($\sigma_{\mathsf{agg}}, pk_{\mathsf{agg}}$, m)=1 iff Verify($\sigma_i, pk_i, m$)=1 for all I
- Similar to how to batch DLEQ proofs work.

Blind BLS

$$sk \leftarrow^{\$} \mathbb{Z}_q$$
$$pk = g^{sk}$$

$$\bar{m} = H(m) \cdot g^r \longrightarrow$$



$$\longleftarrow \bar{\sigma} = \bar{m}^{sk}$$

User

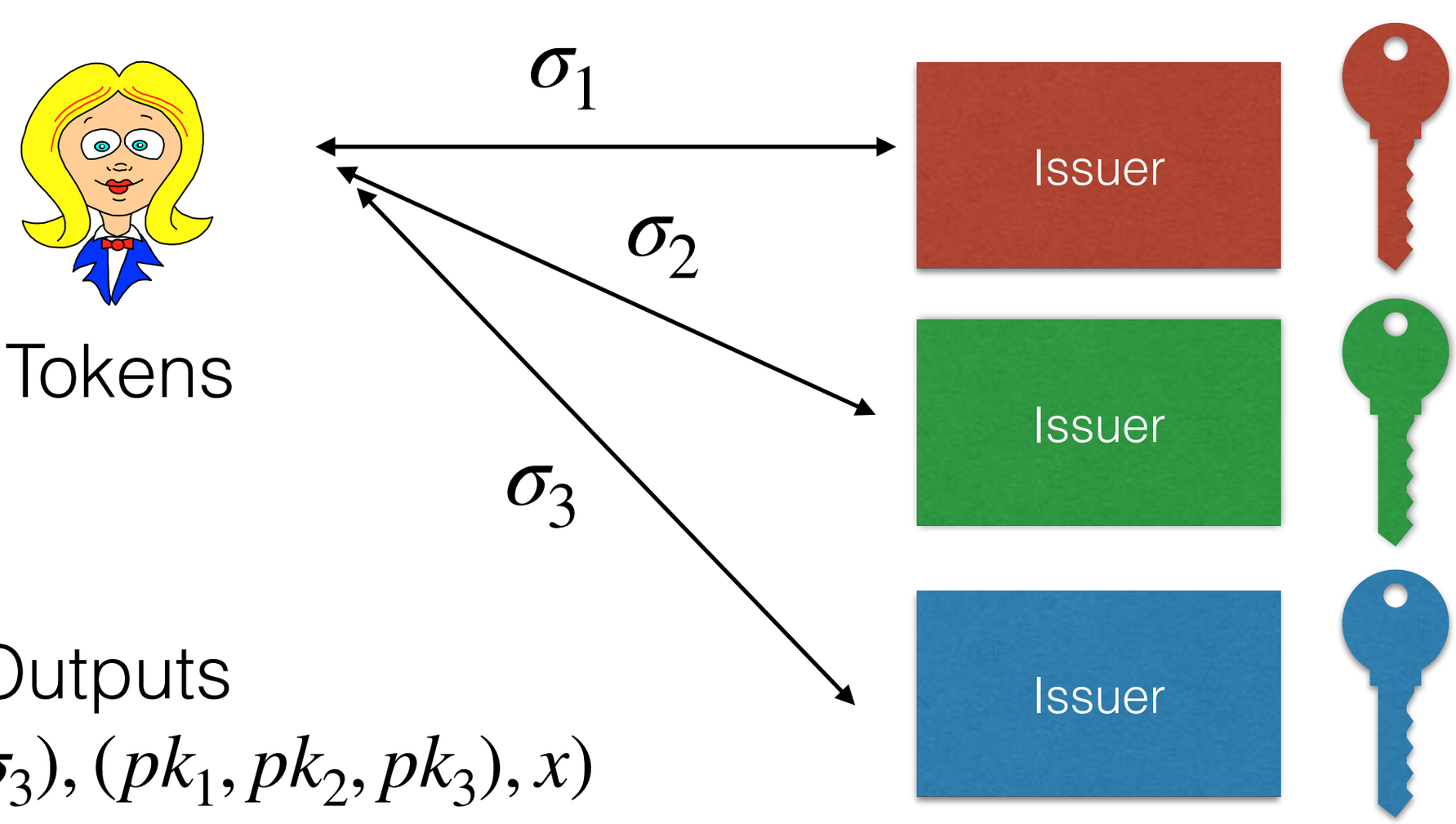
Server

$$\sigma = \bar{\sigma} \circ pk^{-r}$$

Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme. Alexandra Boldyreva. 2002.

Decentralized issuance

- Choose a token input x
- Gets blinded sig from issuers
- Aggregate them



Outputs

$\mathbf{Agg}((\sigma_1, \sigma_2, \sigma_3), (pk_1, pk_2, pk_3), x)$

Blind Multisignatures for Anonymous Tokens with Decentralized Issuance

Ioanna Karantaidou*
George Mason University
Fairfax, VA, USA
New York University
New York, NY, USA
ikaranta@gmu.edu

Omar Renawi*
CISPA Helmholtz Center for Information Security and Saarland University
Saarbrucken, Germany
omar.renawi@cispa.de

Foteini Baldimtsi
George Mason University
Fairfax, VA, USA
Mysten Labs
New York, NY, USA
foteini@gmu.edu

Nikolaos Kamarinakis
University of Maryland
College Park, MD, USA
Common Prefix
Athens, Greece
k4m4@umd.edu

Jonathan Katz
Google
Washington DC, USA
University of Maryland
College Park, MD, USA
jkatz2@gmail.com

Julian Loss
CISPA Helmholtz Center for Information Security
Saarbrucken, Germany
loss@cispa.de

Abstract

We propose the first constructions of anonymous tokens with decentralized issuance. Namely, we consider a dynamic set of signers/issuers; a user can obtain a token from any subset of the signers, which is publicly verifiable and unlinkable to the issuance process. To realize this new primitive we formalize the notion of blind multi-signatures (BMS), which allow a user to interact with multiple signers to obtain a (compact) signature; even if all the signers collude they are unable to link a signature to an interaction with any of them. We then present two BMS constructions, one based on BLS signatures and a second based on discrete logarithms without pairings. We prove security of both our constructions in the Algebraic Group Model. We also provide a proof-of-concept implementation and show that it has low-cost verification, which is the most critical operation in blockchain applications.

1 Introduction

In the digital world, *authorization* plays a foundational role. From regulating access to online services to ensuring the integrity of voting systems, effective access-control mechanisms are crucial for maintaining security and trust. User authorization can be implemented via various methods depending on the application scenario. Common approaches include using credentials such as usernames and passwords or relying on third-party services, such as Auth0 or OpenID, to access user accounts. However, these authentication methods raise concerns regarding user privacy. Each time a user logs in, the service learns everything about the user's activities, enabling the creation of a full profile of their habits. While this level of information leakage may be necessary for certain applications, in many other cases it is desirable to avoid it. Consider, for instance, a subscription-based news portal. In a privacy-friendly world, the only thing that the service should learn is *whether* the user has a

Also proposed a pairing-free version.