# PubHubs Identity Management*

Bart Jacobs, Bram Westerbaan,
Omar Javed, Harm van Stekelenburg,
Lian Vervoort, Jan den Besten

iHub, Radboud University, Nijmegen, The Netherlands

**Abstract**

Finding a combination between privacy and accountability in the online world is a challenge. Too little accountability supports problematic behaviour. Too little privacy undermines individual freedom and has a chilling effect. This paper describes the identity infrastructure of a new open source community platform called PubHubs. It combines local group conversations, via its own adaptation of Matrix, with proportional authentication of users, in local identity spaces. PubHubs thus achieves a combination of privacy and accountability. The technical core of the paper describes the cryptographic protocols for managing digital identities, via both personal attributes and local pseudonyms. Around this core, the roles of digital identities within the PubHubs platform are explained in functional terms, giving participants for instance more certainty about others in a conversation, or giving moderators new tools.

## 1 Introduction

There is widespread discomfort about the influence of the current big, global social networks, like Facebook, Twitter and TikTok. They have commercial goals and employ various techniques to keep users on their platform — via algorithmic engagement maximisation, leading to extremism and fake news — and to profile users so that they can be manipulated for commercial or (geo)political aims, see *e.g.* [9, 21, 22]. Many people have become dependent on such networks for their social interaction and for news and entertainment, but the networks offer little certainty (about other users) or reliability (*e.g.* about authenticity of content). Although these networks thus perform a role as utility, their regulation is still in its infancy. In Europe such regulation is just starting, notably via the Digital Services Act. This leaves room for alternative platforms that

*To appear in: Journal of Logic and Computation 2023

put public values at the center stage, like The Good Web [8] or New_ Public[1]. PubHubs[2], short for Public Hubs, is one such alternative that started in 2022 in The Netherlands. Most of the design & development of PubHubs happens within an academic setting, but connections and collaborations with other (non-profit) parties are being set-up. The PubHubs initiative has quickly attracted attention, at least in the Netherlands. A broadly adopted motion in national parliament urged the government to support the initiative and communicate with citizens via PubHubs, in the future.

PubHubs uses groups of users (communities) as its main units, and not individuals with personal profiles. It therefore likes to see itself as a 'community' platform instead of a 'social' network — the term that is commonly used today. The underlying idea is that people usually communicate in small groups, at work, among friends, within the neighbourhood, within a school *etc.* Thus, participants in PubHubs cannot talk (or shout) to the whole world, and vice-versa, are not exposed to the whole world — with all associated risks. The idea behind PubHubs is that people are more likely to behave a bit more civilised in smaller (online) communities. In addition, smaller groups are easier to moderate, via community stewardship stemming form the communities themselves.

PubHubs is organised as a network of independent hubs, with a shared single-sign-on. Conversations take place in local hubs (and not globally) and the associated (conversation) data is managed decentrally, within each hub. Each participant has, per hub, a separate, persistent pseudonym, so that conversations in different hubs take place in separate (name) spaces. Hubs, or actually rooms[3] within hubs, may ask (optionally) for additional personal information, in the form of 'attributes', such as a postal code for a neighbourhood community. The identity infrastructure aims to combine privacy protection and accountability. Even if people are known locally, in a hub, via local pseudonyms, they can still be held accountable if they behave against the rules of the hub. Ultimately, they can be banned from the hub — by blocking their local pseudonym — but less severe sanctions are also possible, like: delayed posting, exposure of identity details — to the moderator or to the whole room — or temporary bans.

PubHubs builds on the Matrix protocol [7], an open standard for decentralized real-time communication, and uses a version of the open source Synapse matrix homeserver[4]. The software of these homeservers is modified via custom modules, to form 'hubs' within PubHubs. The homeservers each serve their own client to the user, see Figure 2. Crucially, PubHubs does not use the federation offered by the Matrix protocol but adds its own identity infrastructure. This identity layer is incompatible with the Matrix federation and does not accept such commonly used Matrix-logins. PubHubs requires a more elaborate identity infrastructure than offered by existing social platforms [19], that are in essence username-password based and rely on self-declaration when it comes to personal information (and derived, 'implicit' identity information, see *e.g.* [4]). They may

[1] See newpublic.org

[2] See pubhubs.net and the PubHubs position paper for more information.

[3] A room, in Matrix terminology, corresponds to a channel in Slack.

[4] Synapse is the name of the most common and popular Matrix homeserver.

verify an email address or phone number of ordinary users, and perform a more serious identity check for famous people, for people who choose to pay for such checks, or for those who are suspected of identity fraud [15].

Since moderation, including possible bans, is an essential part of PubHubs, it must be prevented that banned individuals re-appear easily under a newly (self-created) identity. For this reason, at registration, new participants need to disclose persistent personal attributes that are (relatively) hard to re-create.

Personal attributes are used not only at registration, but optionally also within hubs. The purpose is not: providing (vertical) top-down control, but: providing participants in a conversation reliable relevant information and appropriate levels of (horizontal) certainty about the identities of other participants. In addition, such attributes may be used in attribute-based signatures [16, 1], to ensure authenticity of certain posts (*e.g.* from a medical doctor) in conversations.

Given that hubs run their own homeserver, handling of conversation data happens locally, within these hubs. This data is thus invisible to other hubs, or to what we call PubHubs Central: the organisation running the central login. A point of continuous care in the design and implementation of PubHubs is to make sure that local data remains local, while at the same time users have an overview over what happens locally. For instance, once logged in, users will be given an overview of their favourite hubs and of the unread messages that sit waiting for them there.

In addition, PubHubs Central works together with a separate centrally hosted entity, called the Transcriptor. The set-up is such that PubHubs Central knows the identity of PubHubs users, but not where they go (that is, to which hubs), while the Transcriptor knows where people go, but not who they are. Details appear in Section 3. PubHubs Central and the Transcriptor are honest-but-curious. Individually, even when they deviate from their tasks, their access to privacy-senstive information remains limited. Only when both components become attackers and start cooperating, user privacy will be compromised. During normal operation, PubHubs Central and the Transcriptor work completely independently, with separate tasks, within separate organisations (legal entities). Under exceptional circumstances, for instance when ordered by a judge, they can cooperate and reveal identifying information about a particular participant, see Subsection 5.1 for details.

The emphasis in this paper is not on PubHubs itself, as an emerging community platform, but on its identity infrastructure and on the role of this infrastructure in a community platform, for instance to support moderation. The infrastructure is generic and of wider interest, since it may be used for other distributed platforms as well. However, here we will describe it in the context of PubHubs. The essential parts of this infrastructure have been implemented and are now being tested and refined in small pilots, see Section 6 for details.

This paper describes the main ideas behind this identity layer and its design, including some of the cryptographic protocols involved, especially for pseudonyms (building on [18]). Personal attributes are handled via the open source

privacy-friendly authentication app Yivi[5]. Within Yivi, users can collect personal data about themselves, from trusted sources, in the form of attributes, such as name, address, phone number, email addres, date of birth, citizen number *etc.* These attributes can be disclosed selectively, via zero-knowledge proofs. The details of how Yivi works are beyond the scope of the paper. Yivi may be seen as a precursor identity wallet, as currently under development in the European Union[6]. Later on, PubHubs may use such other wallets too. In the descriptions below, Yivi is simply used as an existing, independently operated component in the PubHubs infrastructure.

The identity layer of PubHubs serves to combine privacy protection and accountability.

1. It gives participants in group conversations an appropriate level of certainty about other participants in a conversation. For instance, in patient self-help groups, typically centered around a particular disease, the patients themselves often like to remain unknown (pseudonymous), but they do wish to recognise healthcare professionals as such, specifically their medical expertise and/or medical registration number.

2. It gives the hub administrators and moderators effective means for moderation — a key element of PubHubs — and for conflict resolution. Even if participants are known only via (persistent) local Hub-pseudonyms, they can be addressed and even sanctioned, if needed, via these pseudonyms.

One may think of this set-up as securitised openness[7]: the PubHubs platform is open for everyone to join and to remain unknown (pseudonymous), up to a point, where personal details become relevant — like a postal code, in order to participate in a neighbourhood discussion room. There is a security infrastructure in place that can be activated (by hub administrators and moderators) for providing horizontal certainty in rooms and for holding users accountable. How moderation is organised in PubHubs is still very much under development. It is briefly discussed in Section 5, in relation to banning, but moderation itself [12] is out of scope.

The paper starts with an overview of the identity infrastructure of PubHubs. Subsequently, Section 3 discusses how 'polymorphic' pseudonyms are managed, not by users, but by 'the system' (and also by its administrators). Section 4 explains the role of attribute-based authentication in PubHubs, both for central login and for 'secure' rooms in hubs. Section 5 describes how the identity infrastructure can be used to support moderation and ultimately banning. Finally, Section 6 and 7 describe the current status of PubHubs and draw conclusions.

[5]see yivi.app; the Yivi app, formerly known as IRMA, see [2], has been renamed for legal reasons.

[6]For an early analysis, see https://blog.avast.com/analysis-of-eu-digital-identity-architecture-and-reference-framework-avast.

[7]The term 'securitised openness' is borred from the Good Web project [8], an initiative with aims that are similar to PubHubs.

## 2 An architecture overview of PubHub's

This section introduces the essential aspects of the identity infrastructure of PubHubs, starting from the picture below. Subsequently, the PubHubs client and the architecture will be discussed. Later sections will provide the relevant details.

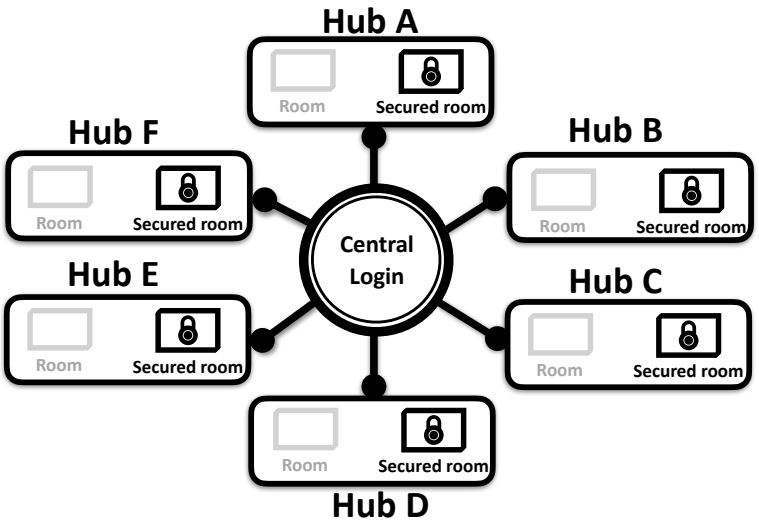


Figure 1: Schematic description of PubHubs, with its central login, providing access to separately operated hubs, each offering (secure and non-secure) rooms where the actual conversations take place.

After the central login, all participating hubs are accessible for users, without further authentication, see Figure 1. Automatically, "under the hood", users get their own (local, persistent) pseudonym when they choose to enter a hub, see Subsection 3.4 for details. Within hubs additional authentication requirements may exist for 'secure' rooms, as indicated by the locks in the picture. When users choose to enter such a room, they are asked to disclose the required information about themselves (in the form of attributes), see Subsection 4.2 for details.

The entity for central login, see Figure 1, will be called PubHubs Central, often abbreviated as *PHC*. It consists of a combination of services that will be organised by the legal entity that runs PubHubs.

The end-user interacts with PubHubs via a web-based *PubHubs client*. This is a custom Matrix client that supports the identity infrastructure that PubHubs adds to Matrix. The PubHubs client consists of several layered components, see Figure 2. The overal goal is to guarantee that the central level does not learn who does what in which hub. Also, activities in different hubs are separated.

1. The PubHubs client forms a container for multiple separate *hub clients*, through which the user interact with a hub (reading and sending messages,

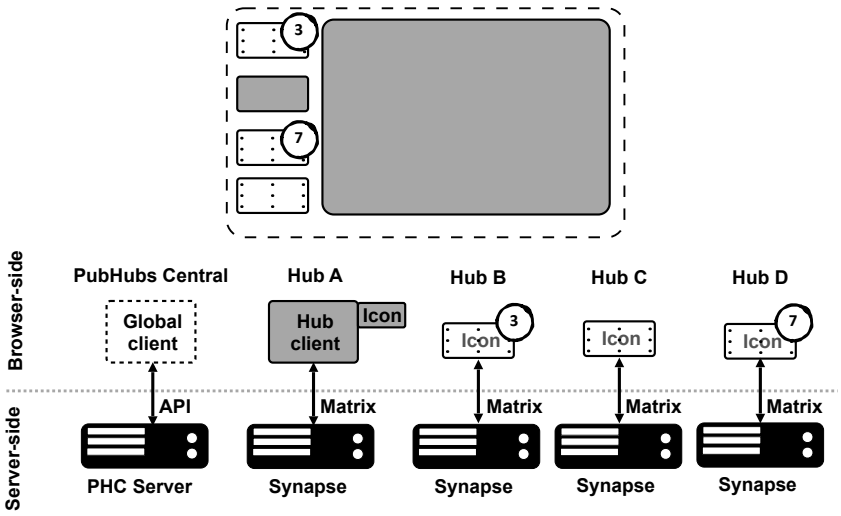


Figure 2: A schematic overview of the PubHubs client. The user's view is on top, with a sidebar of hub icons. The hub in which the user is currently active is in gray. The outer dotted part is served by PubHubs Central (*PHC*). The inner parts stem from separate hubs, here called A, B, C, D. The user can seamlessly switch to other hubs via icon-clicks. At the bottom the communication of the different parts of the PubHubs client with their servers is represented. The numbers 3 and 7 indicate the totals of unread messages in (rooms within) a hub. This is local information that is processed within the PubHubs client in separate silos.

joining rooms). These hub clients are served in an iframe from a domain controlled by the hub, for example, a domain `https://hub-a.com` of a hub called 'a'. This domain separation not only allows hubs the freedom to customise their own hub clients, but also prevents hubs from (accidentally) accessing user information from other hubs, or from the central level.

2. The Hub clients are in an iframe within the global client which is served from the domain of PubHubs Central (*PHC*). These iframes provide separate data spaces, for the different hubs and for *PHC*. Users can quickly switch between hubs, via a sidebar of *icons* showing the different accessible hubs. This hub icons are listed on the top left in Figure 2.

3. The *hub icons* in the global client's sidebar are served by hubs themselves, in iframes too, in order to separate data flows for privacy protection. In this way the hubs do not leak, for instance, the number of unread messages (in hubs) to the global client. This list of relevant hub-icons must be synchronized across multiple devices of the same user. Hence it is stored at PubHubs Central, but encrypted, with a key stored in the global client, see Remark 4.1 on page 18 for more details.

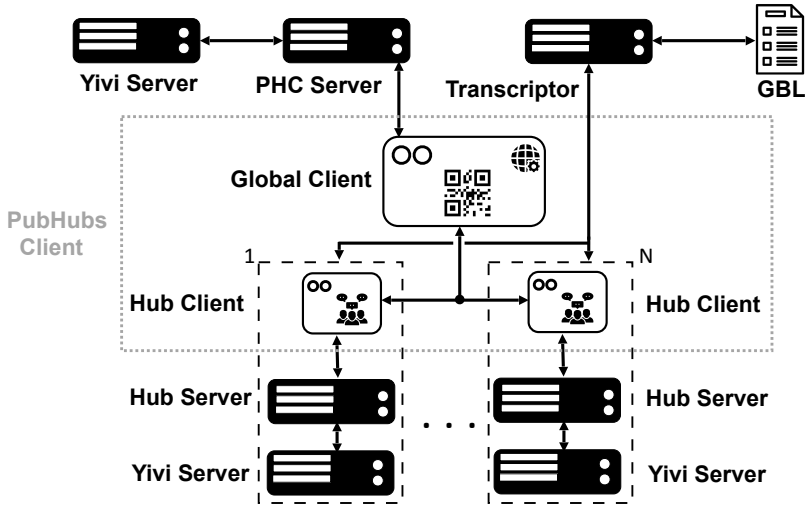


Figure 3: The main components of PubHubs. The Yivi server is an Authentication Server that communicates with the PHC server for central login. The bottom Yivi servers communicating with hub servers are used for authentication for secure rooms (if any) within hubs. The Transcriptor is explained in Section 3 and the Global Ban List (*GBL*) in Section 5.

Figure 3 puts the client of Figure 2 in a wider perspective. At the top level we see the central components of PubHubs: a Yivi server for attribute-based

authentication, a PubHubs Central (*PHC*) server, a Transcriptor (*TS*), and a Global Ban List (*GBL*). The Yivi server and *GBL* play auxiliary roles; the real work is done by *PHC* and *TS*. These two components must be run separately, by different organisations. Roughly, the idea is that *PHC* knows the identities of PubHubs users, but it does not learn to which hubs these users go. The Transcriptor does know about the hubs that are visited by 'entities', but it does not know the identities of these entities, not even their pseudonyms. This requires some careful separation of tasks and of cryptographic secrets, as will be explained in the next section.

# 3 Polymorphic pseudonyms in hubs

The Elgamal cipher [10, 17] provides the cryptographic basis for the pseudonym management in PubHubs, following [18]. It uses Elgamal Encryption $\mathcal{EG}$ with associated operations $\mathcal{RK}$, $\mathcal{RR}$ and $\mathcal{RS}$ for respectively re-keying, re-randomisation and re-shuffling. This approach is called PEP, for Polymorphic Encryption and Psuedonymisation. It exploits the malleability of Elgamal encryption. Its essentials are described below. PEP is actively being used in the management of large collections of medical data for research purposes [14]; it also features in a design for privacy-friendly analysis of network traffic [20]. The Dutch government also uses PEP for pseudonymous identity management[8]. Below we show how these cryptographic primitives are used in the PubHubs identity infrastructure. A cryptographic (security) analysis of these primitives is out of scope.

## 3.1 A recap on Elgamal encryption over a curve

Pseudonyms in PubHubs are sequences of 32 bytes. They are too large to be managed by humans, but they do play a (partly) visible role and can be replaced by user-chosen nicknames, see Remark 3.1 on page 15. Mathematically, the PubHubs pseudonyms encode points on an elliptic curve, written abstractly as $E$, with addition $+$. The particular curve $E$ that is used is the 'ristretto' variant[9] of curve25519 [3] used also by *e.g.* Signal [6]. Vanilla curve25519 was designed specifically for Diffie—Hellman key exchange and contains a 'small subgroup' that can easily cause problems (known as small-subgroup attacks) for more general applications[10]. Ristretto removes the small subgroup of curve25519 resulting in a group $E$ that has the cryptographically desirable property of being a cyclic group of prime order $\ell$, a very large number. This means that for every pair of non-zero points $P, Q \in E$ there is a number $n \in \mathbb{Z}/\ell\mathbb{Z}$, that is $n \in \mathbb{Z}$ modulo $\ell$, such that $Q = n \cdot P = P + \cdots + P$ ($n$ times); moreover, the map $P \mapsto n \cdot P \colon E \to E$ is a group isomorphism, for any non-zero $n \in \mathbb{Z}/\ell\mathbb{Z}$. In other words, all non-zero points of $E$ look the same. The numbers $n \in \mathbb{Z}/\ell\mathbb{Z}$ are

[8] See their BSNk PP service; page in Dutch.

[9] see ristretto.group and [13].

[10] See *e.g.* https://moderncrypto.org/mail-archive/curves/2017/000898.html.

called *scalars*. A special *base point* $B \in E$ is fixed. The scalar multiples $n \cdot B$, for $n \in \mathbb{Z}/\ell\mathbb{Z}$, then generate all points of $E$.

The cryptographic use of $E$ derives from the observation that only the basic group operations, such as addition $P + Q$ for $P, Q \in E$ are easy to compute — and anything directly derived, like scalar multiplication $n \cdot P$ for $n \in \mathbb{Z}/n\mathbb{Z}$ and $P \in E$. For example, while we know that for all non-zero points $P, Q$ there is a unique $n \in \mathbb{Z}/\ell\mathbb{Z}$ with $Q = n \cdot P$ (called the base-$P$ *discrete log* of $Q$) no currently[11] viable method to compute such $n$ is known. Finding such $n$ is called the Elliptic Curve Discrete Logarithm Problem (ECDLP). The Elliptic Curve Diffie-Hellman Problem (ECDHP), finding $n \cdot m \cdot P$ given $n \cdot P$ and $m \cdot P$, is another task considered intractable.

For encryption, a *master private key* $x \in \mathbb{Z}/\ell\mathbb{Z}$ is used. Within PubHubs this key $x$ is actually a product $x = x_{PHC} \cdot x_{TS}$ of two scalars. For security reasons, it is divided over two separate parties: one part of $x$ is held by PubHubs Central (namely $x_{PHC}$) and one part by the Transcriptor (namely $x_{TS}$). Both these parties keep their parts of the master private key $x$ secret and in particular, they do not share there parts between them. Both shares $x_{PHC}$ and $x_{TS}$ are used separately below, in the protocol (4), to derive a private key $x_H$ for a participating hub $H$, in such a way that only $H$ knows $x_H$.

The *master public key* is the point $Y$, defined as $Y \coloneqq x \cdot B \in E$, where $B$ is the fixed base point of curve25519. This point $Y$ is known to everyone involved in PubHubs. The public key for hub $H$ is the point $Y_H \coloneqq x_H \cdot B$.

The *Elgamal* encryption scheme will be described by a function written as $\mathcal{EG}$. It takes three arguments, namely a random scalar $r$, a message $M \in E$ to be encrypted, and a public key, say $Z \in E$. The function $\mathcal{EG}$ produces a 3-tuple, of the form:

$$\mathcal{EG}(r, M, Z) \coloneqq \langle\, r \cdot B,\ r \cdot Z + M,\ Z \,\rangle. \tag{1}$$

The first input $r$ is a arbitrarily chosen scalar that is used to make the encryption random, so that multiple encryptions of the same message look different[12]. The second input $M \in E$ of $\mathcal{EG}$ is the message that is being encrypted. The third input $Z \in E$ is a public key. This public key $Z$ is repeated as third component of the ciphertext output for notational purposes and is in practice often omitted.

If a number $z$ is the private key associated with the above public key $Z$ (so that $Z = z \cdot B$) then the owner of $z$ can decrypt a ciphertext $c = \langle c_1, c_2, c_3 \rangle$ produced via (1), in the following way.

$$\mathcal{EG}^{-1}\Big(\langle c_1, c_2, c_3 \rangle,\ z\Big) \coloneqq c_2 - z \cdot c_1. \tag{2}$$

The crucial property, $\mathcal{EG}^{-1}\Big(\mathcal{EG}(r, M, z \cdot B), z\Big) = M$, is not hard to prove.

[11] An algorithm exists for a quantum computer, but such a machine has yet to materialise.

[12] One has to be careful: reusing both the random $r$ and the public key $Z$ is problematic since subtraction of ciphertexts $\mathcal{EG}(r, M_1, Z) - \mathcal{EG}(r, M_2, Z) = (0, M_1 - M_2, 0)$ leaks information $(M_1 - M_2)$ about the messages.

### 3.2 Deriving private keys for hubs

As mentioned, PubHubs is a network of collaborating hubs under a common umbrella, providing single-sign-on including management of 'local' keys and pseudonyms, see Figure 1. When we say 'local' we mean 'in a hub', opposed to 'at the central level'. (Later in Section 5, we will also use local pseudonyms for the ban list.)

Hubs are run by separate partner organisations, such as schools, libraries, municipalities, patient federations, *etc.*, typically with a public task/role. The PubHubs software is open source, but its network is cryptographically closed, in the sense that only accepted organisations that have obtained a suitable private key can participate. The organisational and legal aspects of this acceptance are beyond the scope of this paper, but in broad terms it involves a hub's support of PubHubs' goals and values and a hub's commitment to properly run and moderate the conversations within their own hub. This is layed down in a contract, between PubHubs Central ($PHC$) and a (new) hub.

Once such a contract is signed, the new hub $H$ installs the PubHubs software — an adapted version of the Matrix homeserver Synapse on its own machine[13] — and registers its own domain name at $PHC$. Subsequently, it receives its own private key, of the form $f_H \cdot x$, where $f_H \in \mathbb{Z}/\ell\mathbb{Z}$ is called a factor, or more specifically, an 'encryption' factor. Another 'pseudonymisation' factor $g_H \in \mathbb{Z}/\ell\mathbb{Z}$ will be used for the hub, see Subsection 3.4 below. Only the Transcriptor knows these factors $f_H$ and $g_H$.

In our implementation these scalar factors $f_H$ and $g_H$ for hub $H$ are not stored by the Transcriptor — *e.g.* in a large table — but they are generated each time that they are needed, via an HMAC from the hub identifier $H$ and a secret key.

The key distribution protocol is described in Figure 4, in the form of a message sequence chart (msc). As usual, we assume that protocol messages are authenticated and encrypted — in practice through TLS and the use of JWTs — and that protocol implementation details are omitted. PubHubs Central ($PHC$) and the Transcriptor ($TS$) have their own, respective shares $x_{PHC}$ and $x_{TS}$ of the master private key $x = x_{PHC} \cdot x_{TS}$. The main reason to split $x$ is to cryptographically enforce the separation of the domains of concern of the transcriptor and PHC. Not only does PHC not record to which hub the users goes, PHC does not even know. Not only does the transcriptor not keep records of the entities making requests to it, it cannot tell who they are. We write $K$ for a secret scalar[14] that is shared (exclusively) between $PHC$ and $TS$.

The protocol in Figure 4 achieves that (the new) hub $H$ receives a private key $x_H = f_H \cdot x$, with corresponding public key $Y_H = x_H \cdot B$. The hub's private key $x_H$ is known neither to PubHubs Central nor to the Transcriptor. The Transcriptor does know the factor $f_H$. This is crucial. With this factor $TS$ can

[13] We foresee that hosting hubs will be offered as a commercial service; this does not alter the story, since from a legal perspective the organisation that signs the PubHubs contract as hub remains responsible, via a processing agreement with the hoster (under the GDPR).

[14] This $K$ depends on the hub $H$ too, but this dependence is omitted for simplicity.

**msc** "Private key distribution to hub $H$"

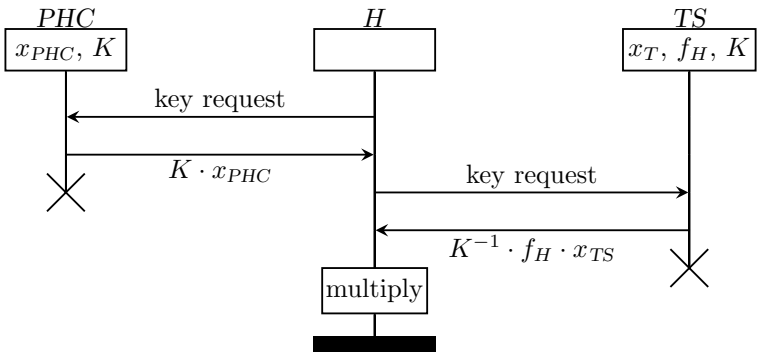


Figure 4: Protocol for distributing a private key $x_H = f_H \cdot x$ to Hub $H$ via multiplication of the (content of the) two messages sent to $H$, namely: $\big(K \cdot x_{PHC}\big) \cdot \big(K^{-1} \cdot f_H \cdot x_{TS}\big) = f_H \cdot x_{PHC} \cdot x_{TS} = f_H \cdot x = x_H$.

blindly 'rekey' messages encrypted with the master public key $Y = x \cdot B$ so that they become encrypted with the hub's public key $Y_H = f_H \cdot Y$. This works via a 'rekey' function $\mathcal{RK}$ acting as follows on a ciphertext $c = \langle c_1, c_2, c_3 \rangle$.

$$\mathcal{RK}\Big(\langle c_1, c_2, c_3 \rangle, f\Big) \coloneqq \langle\, f^{-1} \cdot c_1,\, c_2,\, f \cdot c_3 \,\rangle. \tag{3}$$

If $Z = z \cdot B$, so that $z$ is the private key associated with $Z$, then $f \cdot z$ is the private key for the new public key $f \cdot Z$. The scalar $f^{-1}$ is the multiplicative inverse[15] in $\mathbb{Z}/\ell\mathbb{Z}$, so that $f \cdot f^{-1} = 1$. We now have:

$$\begin{aligned} \mathcal{RK}\Big(\mathcal{EG}(r, M, Z),\ f\Big) &\overset{(1)}{=} \mathcal{RK}\Big(\langle\, r \cdot B,\, r \cdot Z + M,\, Z \,\rangle,\ f\Big) \\ &\overset{(3)}{=} \langle\, f^{-1} \cdot r \cdot B,\, r \cdot Z + M,\, f \cdot Z \,\rangle \\ &\overset{(1)}{=} \mathcal{EG}\big(r \cdot f^{-1},\, M,\, f \cdot Z\big). \end{aligned}$$

This new chiphertext can be decrypted with private key $f \cdot z$.

### 3.3 Polymorphic pseudonym, upon registration

When a new user $U$ registers at PubHubs Center ($PHC$) — see Subsection 4.1 for details — a (random) user identity $\mathrm{ID}_U$ is generated and stored in a (new) user record. This $\mathrm{ID}_U$ is a point on the curve $E$. In each participating hub the user will have a pseudonym that is derived from $\mathrm{ID}_U$, namely $\mathrm{ID}_{H,U} \coloneqq g_H \cdot \mathrm{ID}_U$, where $g_H$ is a hub-specific 'pseudonymisation' factor known only to the Transcriptor ($TS$).

[15]The inverse of $f$ can be obtained as $f^{\ell-2}$, since $f \cdot f^{\ell-2} = f^{\ell-1} = 1$ by Fermat's little theorem.

As part of the registration process, PubHubs Central (*PHC*) forms what is called the *polymorphic pseudonym* $PP_U$ of user $U$. It is an (Elgamal) encryption of the user identity $\mathrm{ID}_U$ with the master public key. Thus:

$$PP_U \coloneqq \mathcal{EG}\big(r, \mathrm{ID}_U, Y\big), \tag{4}$$

where, we recall, $r$ is random scalar and the point $Y$ is the master public key. No single entity within PubHubs can decrypt this, but jointly, *PHC* and *TS* can.

When user $U$ chooses to visit hub $H$, *PHC* sends the polymorpic pseudonym $PP_U$, via the user, to *TS*, so that *TS* can transform $PP_U$ into an encryption (for $H$) of the 'local' pseudonym of $U$ at $H$. This is done in such a way that *PHC* does not learn to which hub user $U$ is going, while *TS* does not learn who is visiting the hub. For details, see Figure 5.

If *PHC* sends the same data as polymorphic pseudonym to *TS* about $U$, this *TS* may learn patterns (of pseudonyms going to hubs). To avoid this, *PHC* first performs a re-randomisation via a function called $\mathcal{RR}$. For a fresh random number $s$, re-randomisation changes a chiphertext $c = \langle c_1, c_2, c_3\rangle$ in the following manner.

$$\mathcal{RR}\Big(\langle c_1, c_2, c_3\rangle, s\Big) \coloneqq \langle\, s \cdot B + c_1,\ s \cdot c_3 + c_2,\ c_3 \,\rangle. \tag{5}$$

Notice that the public key part $c_3$ remains the same. The crucial property of re-randomisation appears when we apply it to an encryption: it yields an encryption with the same key, but with new a random, obtained via addition:

$$\begin{aligned}
\mathcal{RR}\Big(\mathcal{EG}(r, M, Z),\ s\Big) &\overset{(1)}{=} \mathcal{RR}\Big(\langle\, r \cdot B,\ r \cdot Z + M,\ Z \,\rangle,\ s\Big) \\
&\overset{(5)}{=} \langle\, s \cdot B + r \cdot B,\ s \cdot Z + r \cdot Z + C,\ Z \,\rangle \\
&= \langle\, (s + r) \cdot B,\ (s + r) \cdot Z + C,\ Z \,\rangle \\
&\overset{(1)}{=} \mathcal{EG}\big(s + r,\ M,\ Z\big).
\end{aligned}$$

### 3.4 Hub login and pseudonymisation

We now consider what happens when a user $U$ logs into a hub $H$. We assume that the user $U$ is already (registered and) logged in to PubHubs at the central level, and that PubHubs Central (*PHC*) has computed a polymorphic pseudonym $PP_U$ — as described in Subsection 3.3. Logging into a hub does not require any further authentication steps of the user. Simply by clicking on the hub's icon in the icon list — on the left in the client, at the top in Figure 2 — the user's (hub) client starts interacting with the server of hub $H$, whereby the hub-login protocol is executed, see Figure 5. The goal of the protocol is to provide the hub $H$ with the local pseudonym of $U$ at $H$. This local pseudonym is sent in encrypted form, in a message called $PP_{H,U}$ below.

Each time that user $U$ visits hub $H$, this local pseundonym $PP_{H,U}$ is provided (exclusively) to hub $H$, in encrypted form. Thus, the pseudonym is persistent. It can be used by the hub to connect all activities of user $U$ within

that hub, across different rooms in the hub, and across time. Since different hubs get different pseudonyms, tracing users across hubs is not possible — at least not via these pseudonyms. Deliberately, hubs operate as separate islands, technically with their own name and data space, and operationally with their own culture and norms, as expressed *e.g.* via moderation, see Section 5.

This hub-login involves a third operation on Elgamal ciphertexts, in addition to re-randomisation (5) and re-keying (3), which is called re-shuffling. It uses a scalar factor $g$ to transform the message. It works on a ciphertext $c = \langle c_1, c_2, c_3 \rangle$ in the following manner.

$$\mathcal{RS}\Big(\langle c_1, c_2, c_3 \rangle, g\Big) \coloneqq \langle\, g \cdot c_1,\, g \cdot c_2,\, c_3 \,\rangle. \tag{6}$$

The effect is that an encrypted message $M$ is turned into $g \cdot M$, without affecting the key with which the message is encrypted — and thus without affecting who can decrypt.

$$\begin{aligned} \mathcal{RS}\Big(\mathcal{EG}(r, M, Z),\, g\Big) &\overset{(1)}{=} \mathcal{RS}\Big(\langle\, r \cdot B,\, r \cdot Z + M,\, Z \,\rangle,\, g\Big) \\ &\overset{(6)}{=} \langle\, g \cdot r \cdot B,\, g \cdot \big(r \cdot Z + M\big),\, Z \,\rangle \\ &= \langle\, g \cdot r \cdot B,\, g \cdot r \cdot Z + g \cdot M,\, Z \,\rangle \\ &= \mathcal{EG}\big(g \cdot r,\, g \cdot M,\, Z\big). \end{aligned}$$

In the current scenario — of user a $U$ logging into hub $H$ — we wish to ensure the global invariant that *PHC* knows identities, but not where they go, and *TS* does not now identities, but does know the traffic (of unknown entities going to hubs). More specifically, *PHC* knows who logs into some (unkown) hub, at what times; *TS* knows which hubs get logins, at what times. *PHC* and *TS* are assumed to be honest-but-curious: they follow the protocol rules but may collect all the information that is available to them.

The protocol in Figure 5 describes how this works, while abstracting away from many implementation details. The user first goes to *PHC* with an (authenticated) request for its encrypted polymorphic pseudonym $PP_U = \mathcal{EG}(r, \mathrm{ID}_U, Y)$, see 4. The 'user' here means the user's hub client, forming part of the PubHubs client, see Figure 2. Once in possession of $PP_U$, the user proceeds to *TS* and only at this stage makes know that it wishes to connect to hub $H$. Then, *TS* forms the encrypted local pseudonym $PP_{H,U}$ via re-keying and re-shuffling, using the factors $f_H$ and $g_H$ that it knows (or can generate) for hub $H$. This yields:

$$PP_{H,U} \coloneqq \mathcal{RS}\Big(\mathcal{RK}\big(PP_U,\, f_H\big),\, g_H\Big). \tag{7}$$

Notice that in this way *TS* 'blindly' turns a polymorphic pseudonym $PP_U$ into an encrypted local pseudonym, without knowing the identity $\mathrm{ID}_U$ involved. The effect of this operation (7) can be made explicit by unravelling all the definitions

**msc** "Hub login with local pseudonym"

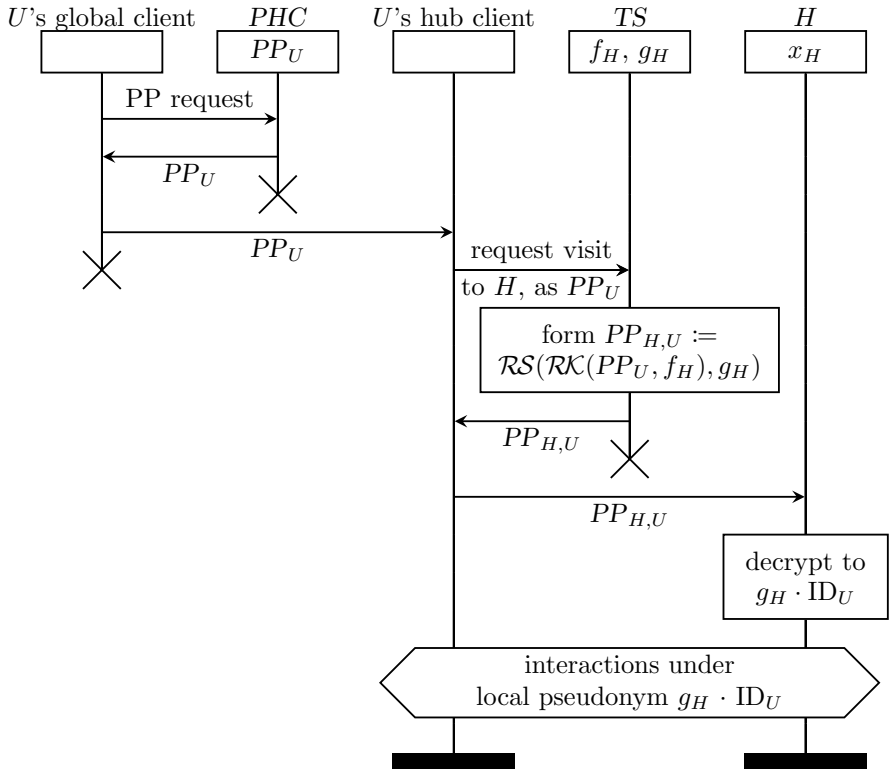


Figure 5: Protocol for providing hub $H$ with an encrypted version of the local pseudonym $g_H \cdot \mathrm{ID}_U$ of user $U$, as start of interactions of $U$ in hub $H$ under this local pseudonym. Recall from Figure 2 that the PubHubs client runs several separate clients, in iframes, on the user's side. The global client has done the central login at PubHubs Central (*PHC*), and can re-authenticate to *PHC* (via a JWT). This happens in the first message, for obtaining an encrypted polymorphic pseudonym $PP_U$ of the user $U$, as in (4), in re-randomised form. The global client passes this $PP_U$ on to the hub client for $H$, via a message exchange between iframes, within the PubHubs client of the user. This hub client then contacts the Transcriptor *TS* and discloses that it wishes to visit hub $H$, using $PP_U$. Subsequently, the Transcriptor produces the message $PP_{H,U} \coloneqq \mathcal{RS}(\mathcal{RK}(PP_U, f_H), g_H)$ via re-keying and re-shuffling, see (7), using its encryption and pseudonymisation factors $f_H, g_H$ for hub $H$. This $PP_{H,U}$ forms an encryption of the local pseudonym $g_H \cdot \mathrm{ID}_U$ that the hub $H$ can decrypt, see (8), with its private key $x_H$.

involved.

$$\begin{aligned} PP_{H,U} &\overset{(7)}{=} \mathcal{RS}\Big(\mathcal{RK}\big(PP_U,\, f_H\big),\, g_H\Big) \\ &\overset{(4)}{=} \mathcal{RS}\Big(\mathcal{RK}\big(\mathcal{EG}\big(r, \mathrm{ID}_U, Y\big),\, f_H\big),\, g_H\Big) \\ &\overset{(3)}{=} \mathcal{RS}\Big(\mathcal{EG}\big(f_H^{-1} \cdot r, \mathrm{ID}_U, f_H \cdot Y\big),\, g_H\Big) \\ &\overset{(6)}{=} \mathcal{EG}\big(g_H \cdot f_H^{-1} \cdot r,\, g_H \cdot \mathrm{ID}_U,\, Y_H\big) \end{aligned} \tag{8}$$

This result can be decrypted by the hub $H$, with its own private key $x_H = f_H \cdot x$, corresponding to public key $Y_H \coloneqq x_H \cdot B = f_H \cdot x \cdot B = f_H \cdot Y$.

**Remark 3.1.** *At the end of the protocol in Figure 5 user $U$ can start its interactions at hub $H$, in the different available rooms. In principle, the user's identity is displayed as the 32 byte local pseudonym $g_H \cdot \mathrm{ID}_U$. This is not very human-friendly. Therefore, PubHubs offers users the option to choose a local nickname, in each hub. Automatically, a few characters from the local pseudonym are appended to the nickname, for persistency and recognisability. The displayed nicknames in PubHubs are thus of the form 'john123'. It may be changed to 'bob123', but the '123' part remains stable.*

# 4 Attribute-based authentication

PubHubs combines two identity management mechanisms, one based on pseudonyms and one on personal attributes. The previous section described how these local, hub-specific pseudonyms work. The current section will focus on how and where attribute-based authentication is used, namely for registration and central login, and for accessing secure rooms within hubs.

In general, attribute-based authentication is a privacy-friendly mechanism that allows individuals to reveal very specific information about themselves in order to get access. An example is the attribute "older than 18" which may be required to play certain games online, or to order alcoholic drinks. The European Union has adopted attribute-based authentication in its wallet-ID plans, launched in June 2021. Yivi (formerly known as IRMA) is a precursor to this EU-wallet that is already up and running in The Netherlands[16].

Yivi is a non-profit, open source mobile phone app, with decentralised (in-app) storage of attributes, and selective disclosure of attributes via zero-knowledge proofs based on Idemix, see *e.g.* [5, 2] for details. The essence is that users can collect in their Yivi app digitally signed attributes from trusted issuers, such a public or private registers. Technically, several related attributes are issued simultaneously, in a single credential. For instance, a 'citizen' credential that is issued in The Netherlands contains, among other things, family name, given names, date of birth, age limits (like: older than 16, or 18), and citizen number. Subsequently, users can selectively disclose such attributes to 'verifiers' that they trust, like health service providers or webshops. This disclosure happens

[16] With, at the time of writing, more than 100K users, especially in the health care sector.

via a direct connection between the user's Yivi app and the verifier, without any intermediate third parties that form a privacy hotspot. This connection between the Yivi app and the verifier is established via a scan of a QR-code, see Figure 6. At the moment, PubHubs uses Yivi, but it might include other EU-wallets in the future — as long as they are free, open source, privacy-friendly and secure.

Below we briefly discuss the two places where attribute-based authentication (via Yivi) is used in PubHubs, namely at the central login and at secure rooms, see Figure 1.

## 4.1 Registration of new user

When a new user chooses to join a specific hub, the user is redirected to PubHubs Central (*PHC*) where the user needs to register to PubHubs. Upon registration, PubHubs' general (privacy) policy should be accepted. It holds for the whole system and includes general rules about processing of personal data and about respectful behaviour. In addition, hubs may have their own policies.

This central registration involves the disclosure of identifying personal attributes by the new user, via their Yivi app, see figure 6. These attributes are stored in the user record that is created, together with the user identity $\mathrm{ID}_U$, see Subsection 3.3. These attributes are not disclosed to hubs. All they get to see (via *PHC*) about users are their hub-specific pseudonyms.

The identifying information of users is required at the central level so that user have a persistent identity. The goal is to prevent users from easily setting up a new identity, *e.g.* after being banned, see Section 5.

In the current set-up of PubHubs, a new user is asked to reveal both an email address and a mobile phone number. This is a pragmatic choice, to be explained below. It does not fully preclude that users reregister with a different email and phone number, but it makes it harder. When needed, additional attributes can be required upon registration, like a (real) name or date of birth from a citizen credential in Yivi.

So why disclosure of email and phone? There are several reasons.

- PubHubs takes data minimisation, as required by the GDPR, seriously.
- Asking much personal information scares off potential new users. For that reason a real name is not required for registration.
- Yivi is not really used outside The Netherlands. Such usage in other countries requires connections to national (public and private) registers, so that people can collect sufficiently many useful attributes. Email and phone attributes are different in Yivi, since they are not issued from a register. They are issued after an ownership check, via a one-time message. Anyone on earth can thus install the Yivi app and get an email credential in their app, simply by replying to a confirmation email message. Similarly, individuals can add their mobile phone number to Yivi

via a SMS-confirmation[17].

- In addition, in exceptional operational cases, users may be contacted via email and/or phone[18].

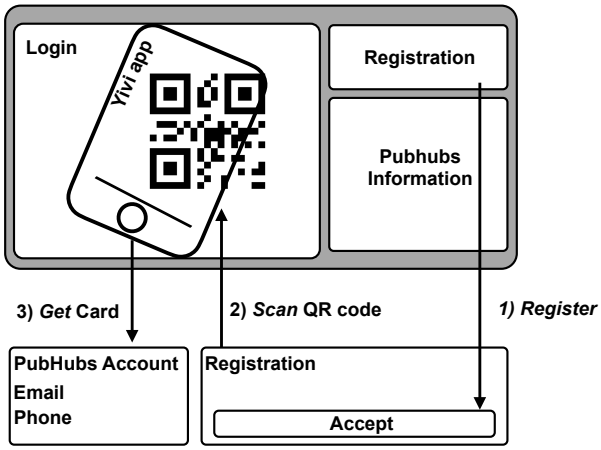


Figure 6: The central registration screen where personal attributes are disclosed via a QR-scan with the Yivi app.

Figure 6 gives an impression of how registration proceeds. Assuming that the new user already has the Yivi app installed and has loaded the relevant credentials, the QR-code can be scanned. Within the Yivi app the user is asked to agree to disclosure of an email address and mobile phone number to PubHubs. If so, the user receives within the same Yivi session a new 'PubHubs' credential in the app, including a PubHubs registration number. It can be used for subsequent logins.

Thus, in Yivi terms, PubHubs is not only a verifier, but also an issuer of its own credentials. This PubHubs credential has a validity of one year. Renewal requires redisclosure of the earlier-used email address and phone numbers. It will be possible to change the email address used at registration, while keeping the registration number stable. At registration a check is performed if the newly disclosed email address and/or mobile phone number are not already used for another registration[19].

[17] However, again for pragmatic (cost) reasons, this sending of SMS confirmations is limited to Europe, by the organisation (SIDN) that operates Yivi.

[18] PubHubs is not for profit and reaching out to users for advertisement or manipulation is not foreseen; it is excluded in the goal binding provisions in PubHubs' privacy policy.

[19] Some lenience may be appropriate here, because it does happen that *e.g.* elderly couples share a mobile phone.

After successful registration, the user is automatically logged in centrally. This means that the user receives a JWT and can proceed to the hub of interest. The hub then gets a local pseudonym for this user, as described in Subsection 3.4.

An already registered user can log into PubHubs via essentially the same flow. The user then discloses the earlier-received PubHubs credential to PubHubs Central, and can proceed to any of the participating hubs, without any further user action — except selecting the relevant hub.

**Remark 4.1.** *Upon registration to PubHubs another thing happens, invisible to the user. Space is reserved at PubHubs Central (PHC) for encrypted storage of user secrets, via an encryption key that is available only to the user. These user secrets contain information about the user that PHC should not learn, such as the history of hubs in which the user is active. This information is displayed to the user by the global client, see Figure 2.*

*The encryption key for these secrets is obtained by the user with the help of the central Yivi authentication server. This happens when a user logs in at PubHubs Central, with an existing or new device. Details of how this works precisely are not so relevant for this paper.*

## 4.2 Secure rooms within hubs

When entering a hub, the user is confronted with a 'hubpage' that presents some general information about the hub and about what happens there. The user can participate in the conversations in the hub by proceeding to a room in the hub. Some of the rooms can be entered immediately. But some of the rooms are 'secured'. This means that users who choose to enter these secure rooms have to disclose certain attributes. Setting up such a secure room, and choosing the required attributes, is done by the administrator of the hub. How this works is out of scope here. There can be two sorts of attributes for entering a room: *check attributes* and *profile attributes*.

Check attributes are typically boolean in nature. This means that the user either meets the criterion or does not meet the criterion for accessing the secure room. For example, a neighborhood discussion room will require a *check attribute* to allow only users residing with a specific postal code. Another, *check attribute* can be an age limit, for example older (or younger) than 16. Check attributes can also come in the form of an allowed list. Certain attributes of the user have to be on this list in order to access the room. For example, an allowed list can contain the email addresses of the registered library members. In that case, the user will only gain access to the secure room if the user's email address matches one of the addresses on the allowed list. Such checks ensure that only registered members gain access to the room.

A characteristic property of check attributes is that they are not visible to (other) participants in the room. This is not needed, since all admitted participants share these attributes as access requirements (like a postal code). In contrast, profile attributes are used to reveal relevant information to (other)

participants in a room. For example, a study room comprising of a teacher and students can have identifying attributes, such as a (first and/or last) name of teacher and students, as profile attributes. They are then visible to all participants in the room.

Secure rooms in PubHubs are implemented via an extension of the functionality of the Synapse Matrix server[20]. Users can navigate to the desired hub where the user can select a room of interest, possibly secure. Separately, a Yivi server needs to be set up, enabling a Yivi disclosure flow, via a QR code, to enter a secure room. To avoid repeated disclosure of attributes each time the user enters the same secure room, the room keeps track of whether the user has already disclosed their attributes to the room.

# 5 Moderation and banning

As described, PubHubs offers an (identity) umbrella for independent hubs that run their own homeserver and are responsible for their own data management associated with local conversations in local identity spaces. An integral part of this responsibility includes moderation of these conversations, within their own hub. On the big social media platforms moderation by humans has been set-up after external, societal pressure and is often outsourced to low-income countries (see [12] for a discussion). In contrast, within PubHubs, moderation is seen as an essential part of, and contribution to, one's community, as a respectable civic responsibility [23] and as a form of community stewardship. Hubs will have the freedom to apply their own community norms, within a general normative setting, layed down in rules about respectful behaviour that apply system-wide, across all hubs.

It is too soon to say how this moderation will actually work in PubHubs. Instead, this section will sketch how PubHubs' identity infrastructure offers support for accountability in moderation, and more generally, for handling conflicts, ultimately possibly leading to the banning of users that repeatedly break the rules.

To start, there is the choice if a room within a hub, dedicated to a potentially sensitive topic, should be secure or not. If it is set up as secure, visitors of the room will have to disclose certain attributes about themselves, like their real name or contact details, before they can participate. Alternatively, participants may be asked to digitally sign their posts with a selection of their attributes. In the latter case, only active participants reveal their identity and commit to their contributions, whereas passive participants may remain pseudonymous. All this may have a dampening effect on potential misbehaviour[21]. But even if participants in a room are known only via their (persistent, hub-specific) pseudonyms, moderators can still use a variety of warning and sanction mechanisms, such as

[20] `https://matrix-org.github.io/synapse/develop/usage/configuration/config_documentation.html?highlight=modules#modules` consulted April 17, 2023.

[21] To what extend disclosure of identity information makes a discussion more civilised is an open question. The PubHubs platform may provide useful empirical data.

yellow cards, delayed posting for certain pseudonyms, content scanning before posting, identity disclosure obligation (to the moderator, or to the room) for any further posts, a temporary ban, or ultimately a total ban from the room.

When an individual is banned from a certain room in a hub, the individual's local pseudonym may be put on a special list within the hub. When this individual is banned from several rooms, say from five or ten, the hub may decide to ban this individual from the entire hub, either temporarily or permanently. This banning from a hub can be done without knowing the identity of the banned person; the local pseudonym for that hub suffices. The offending user may then still visit other hubs.

When an individual is banned from a hub, this is registered centrally by an entity called the Global Ban List (GBL), see Figure 3. The hub sends the encrypted pseudonym of the banned individual to the transcriptor and asks it to rekey and reshuffle it to the GBL. The indivual will thus be listed under the local pseudonym for the GBL. When the same individual is banned from another hub, a similar translation via the Transcriptor happens, producing the same local pseudonym in the GBL. As a result, the GBL can detect if certain individuals are banned from multiple hubs — without knowing their identity. At some point, say after 10 hub bans, such an individual may be banned from PubHubs altogether.

There may be several ways to do this. PubHubs Central and the Transcriptor may cooperate for this specific goal and decrypt $PP_U$ in (4) to the user identity $\mathrm{ID}_U$, via their own shares $x_{PHC}$ and $x_{TS}$ of the global private key $x = x_{PHC} \cdot x_{TS}$. Alternatively, the Transcriptor may rekey the encrypted pseudonym to *PHC*, without reshuffling the message, so that *PHC* may decrypt on its own and learn the identity. Such a global ban happens via blocking any further logins with identity $\mathrm{ID}_U$ disclosed at registration (via attributes).

When an individual is banned centrally, via the registered combination of email address and mobile phone number, there is the possibility that the same individual re-registers with a different email address and phone number. Getting a different email address is easy, but getting a different mobile phone number (that is not already in use) is a slightly bigger hurdle. When it becomes clear, in practice, that this hurdle is too low, then PubHubs can always require more identifying attributes at registration, such as a participant's real name. As long as re-registrations of the same (offending) individuals is not a problem, PubHubs likes to keep the attribute disclosure requirements at registration as light as possible.

## 5.1 PubHubs and law enforcement

Finally, we briefly look at what law enforcement authorities can do in the context of PubHubs. Here we assume that there is a disclosure order, based on due legal process, involving an independent judge. We briefly consider multiple scenarios.

- An individual who is active within a hub may become a suspect. In that case at least the individual's pseudonym is visible, within that hub.

The authorities may then request the hub administrator to produce all behavioural data associated with that pseudonym. This could include attributes that the individual has revealed within the hub, at some earlier stage. Possibly, this enables de-pseudonimisation.

- The authorities may also take this pseudonym "upwards" to PubHubs Central and request de-pseudonimisation. This is possible, with the help of the Transcriptor, as sketched above. We recall that PubHubs will be set-up in such a way that *PHC* and *TS* are run by different legal entities, so that both must (be tasked to) cooperate with the request. In this case the registration data of the individual becomes available to law enforcement. In addition, the authorities may ask *TS* to provide pseudonyms for other hubs, so that the local activities of the individual at hand elsewhere, in other hubs, can also be investigated.

- Alternatively, the authorities may show up at PubHubs Central with an e-mail address and/or phone number and ask: are these attributes used for registration at PubHubs? If so, *PHC* can produce registration (meta)data, including the user identity $\mathrm{ID}_U$. The authorities may then proceed to *TS* and request that this identity be translated to hub-specific pseudonyms, so that the authorities can continue their investigation "downwards", within those hubs.

This last scenario seems less likely in a criminal investigation since user activity takes place in hubs, so any offensive behaviour will take place there, decentrally. The authorities will then seek an "upward" connection between a local pseudonym and registration data, as in the second bullet.

We conclude that the PubHubs' minimal, privacy-friendly identity infrastructure is powerful enough to support both internal and external sanctioning, by PubHubs (including its hubs) and by law enforcement authorities. It does offer a combination of privacy-protection and accountability. Whether this combination is sufficiently effective, remains to be seen when PubHubs is deployed at a larger scale.

# 6 Status and outlook

PubHubs is still in its implementation phase and has not been released as production code, for actual usage[22]. This paper focuses on the underlying identity infrastructure, which forms a stable basis. In this section we briefly sketch what is currently there and in what form. Since this information will be outdated fairly soon, it is meant to give a snapshot impression.

- The central login is there, including issuance of a Yivi credential with PubHubs attributes at registration, and with disclosure of these attributes

[22]The software is publicly available at https://github.com/PubHubs/PubHubs, even if it not yet at production stage.

for subsequent login. The possibility to change, once registered, the e-mail address or mobile phone number is not supported yet.

- Generation and translation of hub-specific pseudonyms, and login at Synapse homeservers of hubs via such pseudonyms, is implemented, however without separation of the tasks of PubHubs Central and of the Transcriptor.
- Secure rooms with login via specifically requested attributes work, in rudimentary form, still without properly explained user interfaces.
- The dedicated PubHubs client also exists in rudimentary form.
- PubHubs-specific support for moderation and banning does not exist yet. In particular, the Global Ban List is not implemented yet.

These bullet points address specific technical issues. The development team of PubHubs includes an interface designer. Systematic user tests have not been done yet, for *usable* privacy and security (see *e.g.* [11]).

# 7 Conclusions

PubHubs is a non-profit civil-society initiative to develop an open source community platform based on public values. PubHubs distinguishes itself, as a new community platform, through its digital identity infrastructure. The paper explains the design choices for combining privacy-protection and accountability: users are pseudonymous, in principle, but de-pseudonymisation possibilities exist, both for voluntarily providing (proportional) certainty to other participants in a conversation, and for sanctioning, by (internal) moderators and/or (external) law enforcement. The paper explains how this can be realised using existing cryptographic techniques: selective disclosure of personal attributes, via zero-knowledge proofs, and polymorphic pseudonyms, via Elgamal encryption. The set-up of PubHubs — through its technology and organisation — excludes systematic, global tracking of user activities, for whatever purposes.

Unlike common social networks, communication in PubHubs is compartimentalised by design, into local hubs with their own data and name spaces. This is appealing to the PubHubs community, but what also emerges is a desire for hubs to cooperate, for instance when a local museum wishes to organise an event together with the local library. How to precisely support such cooperation with the PubHubs architecture is one of the challenges for the future. But of course, organising and supporting an active user community is the main challenge, for the near future.

# References

[1] G. Alpár, F. van den Broek, B. Hampiholi, and B. Jacobs. Towards practical attribute-based signatures. In R.S. CHakraborty, P. Schwabe,

and J. Solworth, editors, *Proceedings of the Fifth Int. Conf. on Security, Privacy, and Applied Cryptography Engineering (SPACE 2015)*, number 9354 in Lect. Notes Comp. Sci., pages 310–328. Springer, Berlin, 2015. doi:10.1007/978-3-319-24126-5_18.

[2] G. Alpár, F. van den Broek, B. Hampiholi, B. Jacobs, W. Lueks, and S. Ringers. IRMA: practical, decentralized and privacy-friendly identity management using smartphones. In *10th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2017)*, 2017.

[3] D. Bernstein. Curve25519: new Diffie-Hellman speed records. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *Public Key Cryptography*, number 3958 in Lect. Notes Comp. Sci., pages 207–228. Springer, Berlin, 2006. doi:10.1007/11745853_14.

[4] D. Blanco, J. Sanz, and J. Pavón. Social identity management in social networks. In J. Corchado, S. Rodríguez, J. Llinas, and J. Molina, editors, *International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008)*, pages 62–70, Berlin, 2009. Springer. doi:10.1007/978-3-540-85863-8_9.

[5] J. Camenisch and E. van Herreweghen. Design and implementation of the Idemix anonymous credential system. In *CCS'02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 21–30. ACM, 2002.

[6] M. Chase, T. Perrin, and G. Zaverucha. The signal private group system and anonymous credentials supporting efficient verifiable encryption. Technical report, December 2019. https://signal.org/blog/pdfs/signal_private_group_system.pdf.

[7] The Matrix.org Foundation C.I.C. An open network for secure, decentralized communication. Technical report, 2019. https://matrix.org/.

[8] Demos. The good web project, 2022. https://demos.co.uk/research/good-web-project/.

[9] J. van Dijck, T. Poell, and M. de Waal. *The Platform Society*. Oxford Univ. Press, 2018. doi:10.1093/oso/9780190889760.001.0001.

[10] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. on Information Theory*, 31(4):469–472, 1985.

[11] N. Gerber, A. Stöver, and K. Marky, editors. *Human Factors in Privacy Research*. Springer Cham, 2023. doi:10.1007/978-3-031-28643-8.

[12] V. Gongane, M. Munot, and A. Anuse. Detection and moderation of detrimental content on social media platforms: current status and future directions. *Social Network Analysis and Mining*, 12(129), 2022. doi:10.1007/s13278-022-00951-3.

[13] M. Hamburg. Decaf: Eliminating cofactors through point compression. In R. Gennaro and M. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015*, number 9215 in Lect. Notes Comp. Sci., pages 705–723. Springer, Berlin, 2015.

[14] B. Jacobs and J. Popma. Medical research, big data and the need for privacy by design. *Big Data & Society*, pages 1–5, 2019. doi:10.1177/2053951718824352.

[15] G. Kolaczek. An approach to identity theft detection using social network analysis. In *2009 First Asian Conference on Intelligent Information and Database Systems*, pages 78–81, 2009. doi:10.1109/ACIIDS.2009.44.

[16] H. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In A. Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, pages 376–392, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi:10.1007/978-3-642-19074-2_24.

[17] H.C.A. van Tilborg. *Fundamentals of Cryptology: a professional reference and interactive tutorial*. Kluwer Academic Publishers, 2000.

[18] E. Verheul and B. Jacobs. Polymorphic encryption and pseudonymisation in identity management and medical research. *Nieuw Archief voor Wiskunde*, (5/18 nr. 3):168–172, 2017.

[19] R. Wang, S. Chen, and X. Wang. Signing me onto your accounts through Facebook and Google: A traffic-guided security study of commercially deployed single-sign-on web services. In *2012 IEEE Symposium on Security and Privacy*, pages 365–379, 2012. doi:10.1109/SP.2012.30.

[20] A. Westerbaan and L. Hendriks. Polymorphic encryption and pseudonymisation of IP network flows. In *2020 IFIP Networking Conference*, pages 494–498. IEEE, 2020.

[21] J. Williams. *Stand Out of Our Light*. Cambridge Univ. Press, 2018. doi:10.1017/9781108453004.

[22] S. Zuboff. *The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power*. Profile Books, 2019.

[23] E. Zuckerman. How social media could teach us to be better citizens. *Journ. of E-Learning and Knowledge Society*, 18(3):36–41, 2022. doi:10.20368/1971-8829/1135818.