Bubble Protocol: A Hybrid Web3 Framework

David Potter

Founder of OpenSig & Bubble Protocol
dave@bubbleprotocol.com

v0.0.3 August 15, 2022

Abstract

Web3 promises to reshape the architecture of the web, moving our reliance on central data monopolies towards decentralisation and self sovereignty. Numerous web3 projects are working on digital identity solutions, self-sovereign data and decentralised storage networks but progress towards an integrated user experience that can compete with existing web technology is slow. We believe that a pragmatic engineering approach is needed to bridge the gap between web2 and web3 architecture, one that lets developers, entrepreneurs and organisations embrace web3 technology today with minimal risk while giving everyday, non-technical users a quality user experience.

In this paper we introduce the Bubble Protocol - an on-chain, role-based, hierarchical digital identity solution integrated with a hybrid, private, decentralised storage layer. We will show how these technologies combine with optional third party services to create a practical development framework for building mass-market web3 privacy innovation.

1. Background

Autonomy is the freedom of individuals to lead the life that they wish without coercion from others. Although complete autonomy for all is an unattainable ideal - we share our world with others, after all, and our desired freedoms are not always the same - we can still strive to maximise it. Any loss of privacy reduces our autonomy by revealing information about ourselves that can potentially be used by others to manipulate us into doing what they want. This is especially concerning when we don't know who has our information, allowing others to subtly manipulate individuals or even entire populations without their knowledge[1][2].

In today's data-driven, online world our personal information is no longer our private information. The internet has enabled an exponential rise in data-dependent services and we now share our information more widely than ever before. This has made us vulnerable to identity theft and fraud, and has made it impractical to control our global information footprint.

We frequently hand over personal information to organisations where it is out of reach and out of our control, often blindly accepting terms and conditions hidden behind a checkbox and handing over more data than is needed for the service we are subscribing to. Most organisations, we hope, are keeping our data private and secure, using it only for the purpose for which we shared it. However, the last two decades have seen countless data breaches despite good intentions by organisations to keep our data secure [3]. Some organisations have become powerful data monoliths, selling access to giant sets of our personal data in a way that is unregulated and subject solely to the personal ethics of a few top executives[4][5].

Data protection regulations, like the GDPR in Europe, are genuine attempts to address these problems. However, due to the necessary flexibility and wide reach of these regulations, they have proven complex and confusing both for companies to implement and for regulators to enforce. Critics claim they create artificial barriers to cross-border data flows, amounting to a form of *data protectionism*[6].

Recent advances in cryptography, blockchain technology and the internet can now allow data privacy technology to catch up with our demand for data-driven services. Dubbed *Web 3.0* technology in 2014 by Gavin Wood, co-founder of Ethereum, these advances promise to integrate self-sovereign digital identity, reputation-based authentication, decentralised storage, decentralised communication and user interface design to re-engineer a *"fundamentally different model for the interactions between parties"*[7]. In other words, through the development of web3 protocols and technology we can change the architecture of the web to break up data monopolies and restore our privacy.

2. Introduction

Web3 technology is in it's infancy. Projects like Ontology, 3BoxLabs, Tim Berners-Lee's Solid, and many others, are exploring innovative solutions to the problem of online privacy. Decentralised file storage networks like Filecoin, STORJ and Sia Skynet are providing incentivised, permissionless storage, and projects from companies like IBM, Civic and Sovrin are specialising in digital identity solutions.

While these are exciting and admirable projects, at Bubble Protocol we believe that the web's transition from web2 to web3 will be slow coming. The cost and risk barriers of jumping straight from web2 to web3 technology stacks and the corresponding change in user experience is simply too great for organisations and their customers to adopt in the near term. This will slow the adoption of web3 technology and along with it the arrival of those crucial privacy benefits.

We believe that there is a way to fast-track web3 privacy innovation by developing a hybrid web2/web3 stack that merges many of the key features of web3 with the familiarity of web2. We envision such a hybrid stack as a stepping stone towards web3, one that helps organisations and developers step lightly in the direction of a fully decentralised web without needing a complete change of paradigm. Over time, as web3 technology matures, organisations can take small, steady steps deeper into the technology having already made themselves familiar with its core features. In the meantime, a hybrid stack will give progressive organisations and entrepreneurs a head start innovating with web3 privacy technology, self-sovereign data, monetisation and decentralised applications.

2.1. Overview of The Bubble Protocol

As stated, Bubble Protocol's mission is to help transition organisations and developers from web2 to web3 for the purpose of improving privacy. The project is divided into four initiatives as shown in figure 1. At it's core are Bubble ID and Bubble FS. Together these provide the core protocols that implement a fully decentralized self-sovereign digital identity solution integrated with a decentralized, hybrid private storage layer.

Alongside these two core initiatives are the Bubble Services, which provide optional edge services that developers can use to deliver a smooth user experience and familiar web2 feel.

Finally, Bubble Privacy is a research and development initiative to spearhead privacy, dapp and monetisation innovation on top of the Bubble Protocol.

We'll take each of these initiatives in turn and summarise them.

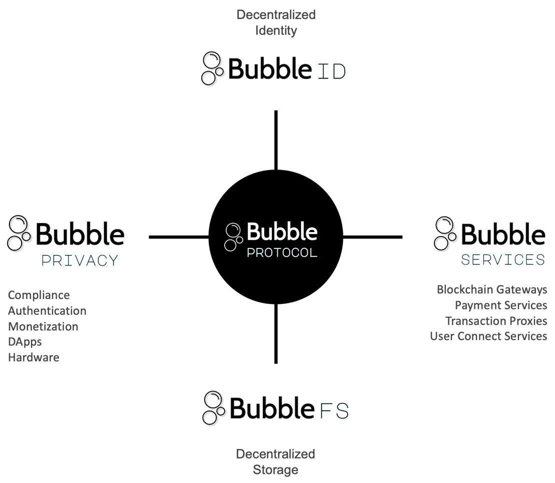


Figure 1: *Bubble Protocol's four initiatives*

3. Bubble ID

Federated identity protocols, like OAuth, let users of web2 services log in with a third party identity provider such as Google, Apple, Facebook, Microsoft and others. The single sign-on (SSO) flow is an improvement over the username/password login process for most users, who benefit from a simpler user experience and better security. It is also an improvement for the web service, which can rely on the authentication of the identity provider without the time, cost and infrastructure required to authenticate the user themselves. When an identity provider applies a rigorous authentication process on its users, like collecting identity documents and running identity checks, it can provide web services with assurances of specific aspects of the user's identity, like their name, age or address. In fact, it can provide this assurance without exposing the user's data, provided the web service trusts it. Unfortunately, the problem with this architecture is clear: it further centralises personal data and the erosion of privacy.

Self sovereign identity (SSI) promises to replace centralised identity platforms with decentralised, open source technology, bringing a user's digital identity back under their control and further improving security. Broadly speaking, public key cryptography replaces the login process; decentralised identifiers and verified credentials replace the need for identity providers; and decentralised key management replaces SSO.

When SSI is combined with decentralised storage and communications, web3 has the opportunity to replace any web2 online service with a decentralised application (DApp). Much progress has been made towards this goal but, on the whole, it has been limited to decentralised finance and the emergent NFT market. We believe there are two reasons for this: 1) decentralised key management is immature; and 2) it is very difficult for non-financial DApps to incorporate the financial incentives needed to cover costs of development. Bubble ID attempts to address both of these problems.

3.1. Decentralised Key Management

It is standard practice for decentralised applications to integrate with a third party wallet. It simplifies security, keeps the user's circle of trust small and allows developers to offload the complexity of key management. Metamask has been instrumental for developers in this regard. While this may be desirable for applications working with infrequent or high-value transactions, requesting permission from the user for frequent transactions or for off-chain authorisation can destroy the user experience.

Take a decentralised communications protocol, for example. For a smooth user experience it is critical that an application has the authority to post and retrieve messages without needing to ask the user repeatedly for permission. This means wallet features must be encoded directly into the application or the communication feature must be encoded into the wallet. Private keys could be shared across devices but this has critical security concerns. These problems severely limit the types of decentralised applications that developers can build.

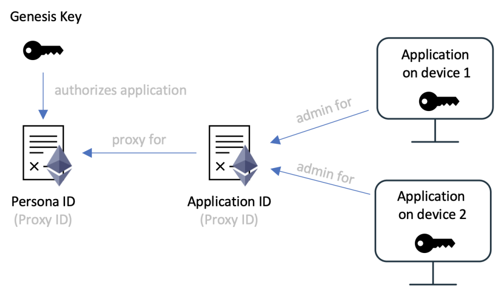


Figure 2: *Chaining Proxy ID smart contracts for decentralised key management*

Bubble ID attempts to address these problems with it's on-chain Proxy ID technology. Proxy IDs are smart contracts that allow the user to delegate role-based permissions to a hierarchy of decentralised applications. It is similar to Ceramic's object capability model[8] but on-chain. Using a process similar to OAuth 2.0, applications can request to connect to a user's Bubble ID, to act on behalf of the user under limits described by specific roles. By approving the request in their wallet, the user delegates those roles to the application thereby permitting it to use its local private key to act on the user's behalf under those roles. Hence, this lets the user delegate identity-related capabilities to applications without the security risk of sharing private keys and without the application needing wallet capabilities.

Proxy IDs can be chained to create a hierarchy, allowing applications to delegate their roles to installations across a user's devices (see figure 2). Roles passed down the chain can be restricted but never expanded.

Proxy IDs let application developers offload key management to a Proxy ID enabled wallet. Since Proxy IDs are on-chain, the user can manage application permissions across all of their devices from within their wallet and can recover their ID from their private *Genesis* key. This means they always have visibility of their applications and devices and can revoke access to any at any time.

Our first wallet implementation the *Bubble Dashboard*, together with the decentralised applications *OpenSig* and *Bubble Pass*, demonstrate key management, role delegation and application management along with bubbles for backend storage and device syncing (see section 4.1).

It's important to recognise that a wallet is itself a decentralised application working under the restrictions of a Proxy ID, having been granted proxy permissions by the user in the same way as any other application (figure 7). The only difference between a wallet and another application is that the wallet is granted admin permissions, which allows it to (almost) fully control the user's identity from any device it is installed on.

It's also important to recognise that the use of Proxy IDs to delegate permissions to applications is just a specific case. Decentralised key management can be applied more generally to relationships between individuals or other entities. For example, a user could grant power of attorney to someone else or a corporate entity could design a Proxy ID structure to represent their company and allow employees to act according to their corporate roles (figure 3). It can also be applied to the internet of things, or a combination of individuals and non-human entities.

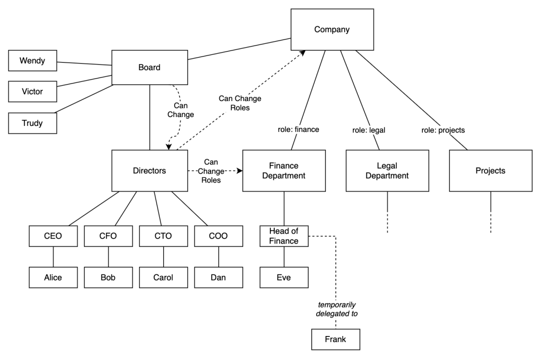


Figure 3: *Chaining Proxy IDs in a corporate identity. Each box is a Proxy ID. Each line is a role delegation. Each arrow indicates an action. The Directors Proxy ID, for example, can change the Head of the Finance Department, perhaps requiring a majority vote between the directors.*

3.2. Enabling Financial Incentives

Proxy IDs allow applications to identify as a user and act on their behalf under specific roles. Some of those roles can be financial, including the ability to spend on behalf of the user. This means applications can be pre-authorised to make transactions directly from the user's wallet without holding the wallet's private key. To enable this, the layer 1 platform or the token smart contract must support Proxy ID smart contracts. The Bubble Services initiative will provide plugins and services to allow applications to accept 'in-DApp' payments in a manner familiar to web2 users. This is explained more in section 7.

4. Bubble FS

One of the core principles of decentralisation is to remove the web's reliance on central databases where personal data is held outside of user control. Projects, like IPFS, Filecoin, STORJ, Arweave, Internxt and Sia Skynet, have built successful technical implementations of permissionless, decentralised cloud storage networks and incentive structures for storage providers. Their innovations are critical steps towards the mass adoption of decentralised cloud storage.

The concept of storing personal data on a public network, though, is counterintuitive. How can a user trust a public network to retain their data and make it available when they need it? How can they be sure their private files cannot be read by someone else, either now or in the future? Users of open storage networks rely on two things to secure their data: 1) the integrity of the network (the reliability and availability of the service); 2) the encryption technology (the encryption of the data to prevent unauthorised access combined with the anonymity of the addressing scheme). A third consideration, not strictly related to security, is the speed of access.

Trust in the integrity of the network will come in time but in the meantime users are forced to backup their data, either on centralised services or on different networks for redundancy. Without trust in the encryption technology, use of the network will be limited to non-sensitive data. It's a big ask for non-technical users to trust an open network with their personal or commercially sensitive files, and even for technical users it is far from certain that encrypted data held in the public domain for decades or longer will remain secure. For the storage of sensitive personal data, these are huge barriers to entry for individuals and organisations alike. We believe this will stifle adoption.

Bubble FS attempts to overcome these barriers with a flexible hybrid storage protocol that we call Smart Data Access. Smart Data Access (SDA) combines digital identity with private storage servers and on-chain access controls to create private 'bubbles' - protected containers of files and directories in which users can store and share private data.

4.1. Bubbles

We define a bubble as a private off-chain storage container protected by an on-chain smart contract. The smart contract integrates with Proxy IDs to control who (and, by proxy, which of a user's applications) can access, update or delete specific files and directories within the bubble. The bubble itself - the data - is held on a bubble-compatible private server. This is any server that implements the Smart Data Access protocol, which could be a home server, cloud server or company server. It is, in principle, possible to implement the protocol as an offline process and use it to manage offline data stores, albeit with some limitations. It is also possible to create a gateway to an existing decentralised storage network, as is demonstrated with our Bubble IPFS Gateway and Bubble Skynet Gateway servers.

The choice of which server to use for a bubble is down to the developer. They can opt to limit users to a specific server, give them a choice out of a list of trusted cloud services or give them the freedom to choose whichever server they trust to hold their data. This flexibility over the storage model is designed to help developers and companies step lightly into web3 since they can integrate Bubble Protocol directly into their existing servers and database architecture or use a bubble-compatible cloud service. They can keep their customer data under their control in the short term while exploring the privacy benefits of decentralised identity and smart data access one step at a time. They can run Bubble Protocol in parallel with their existing web2 service if they wish with both using the same customer database architecture. Over time they can transition to decentralised storage networks.

4.2. Uses for Bubbles

Bubbles can be large or trivially small; long lived or transient. They are general purpose file containers designed to be easily created and instantly accessible so that they can be used in everyday data-sharing transactions and decentralised applications. They can be used for

DApp storage, customer account data, social media applications, communications DApps, data backups, locked archives, photo libraries, document sharing, paywalled online content and NFT content, amongst other purposes. See section 6 for examples.

Our Bubble NFTs web page is an example of NFT controlled web content. NFT images are held in a bubble protected by an online smart contract that interfaces with the Bubble NFT ERC721 smart contract to identify the owners of the NFTs. The web page interacts with the *Bubble Pass* browser extension (a DApp that the user has given permission to identify on their behalf) to identify the user in the browser and to fetch their NFT images. When Bubble Pass fetches an NFT image it signs the fetch request with it's local private key which the bubble server confirms is authorised to identify as the NFT's owner. If the user transfers their NFT to someone else they will instantly lose access to its image.

Section 8 presents four case studies that examine the use of bubbles and Bubble ID for DApp storage, customer account data, paywalled content and over-the-counter transactions.

5. Bubble Services

Despite great promise, decentralised applications have not, on the whole, demonstrated themselves as viable alternatives to web2 applications. Only those that offer a sufficient ideological or financial motive have managed to attract users to go through the pain of learning a new paradigm and buying cryptocurrencies. Wallets are complicated. Buying crypto is complicated and largely unregulated. Private keys cannot be recovered if lost. These are fatal barriers to entry for the average user.

There are efforts within the crypto community to solve these problems in a decentralised way. For example, Open Zeppelin's Meta Transaction standard combined with relayer networks allows developers to pay for their customer's smart contract transactions. Technologies like Web3Auth and the Torus Wallet are helping to make user onboarding familiar to web2 users, provide in-DApp credit card payments, and backup keys to a decentralised network.

The goal of the Bubble Services initiative is to use technologies like these to provide reliable off-chain services that provide practical support to DApps built on Bubble ID and Bubble FS. To align with Bubble Protocol's pragmatic approach, if decentralised technology is not available or is not sufficiently mature then Bubble Services may provide central services until such time as they can be replaced with decentralised technology.

Today, Bubble Services provides three services on the Bubble Testnet. These are:

- *The Bubble Blockchain Gateway* - an alternative to Infura. A service for publishing transactions and querying the ledger.
- *The Bubble Proxy Transaction Server* - allows DApp developers to pay for their customer's transactions.
- *Bubble Connect* - a public bubble that lets users store tiny, transient files for the purpose of connecting DApps and users.

Together with these three, Bubble Services will work alongside Bubble ID to provide support for five more services:

- Private key backup.
- In-DApp credit card payments.
- FIAT stable coin payments.
- Verified ID registry.
- Cloud server reputation.

6. Bubble Privacy

As described, Bubble ID, Bubble FS and Bubble Services together form a platform designed for developers and entrepreneurs to build privacy-focused DApps that are of value for everyday users today. They aim to do this in three ways:

1. By breaking down the barriers of entry that have plagued DApps in the past with a pragmatic web2/web3 (centralised/decentralised) approach.

2. By introducing bubbles - private data sharing technology that is deployable on both decentralised and existing centralised storage.
3. By creating a wallet - the *Bubble Dashboard* - that lets users manage their Bubble ID, personas, DApps and bubbles in a self-sovereign way.

The mission of the Bubble Privacy initiative is to innovate on this platform; to research, develop and promote new opportunities in privacy compliance, authentication and data monetization. The goals of this initiative are to bring privacy, useability and financial benefits to the end users of web3 technology: individuals and organisations.

Some examples of use cases being explored by Bubble Privacy can be found below. See the case studies in section 8 for detailed examples.

- **DApps**. Privacy focused standalone DApps built on the platform, such as end-to-end encrypted messaging, secure file sharing and backup, decentralised social media and digital signatures (see case study 8.1).
- **Verified Credentials**. Bubble Protocol's verified credential (VC) solution: using bubbles to hold VCs and manage the verification process, optionally allowing a regulator access to audit the process.
- **Verified Credential Payments**. Extending VC bubbles to require direct token payment from the consumer, splitting the payment between the owner and the verifier.
- **Bubble Protected User Accounts**. Replacing online web accounts with user-owned bubbles to benefit privacy and compliance (see case study 8.2).
- **Secure Customer Messaging** as part of Bubble Protected User Accounts. Incorporating end-to-end encrypted messaging between organisation and customer directly in the bubble to provide a secure alternative to email. Prevents phishing attacks.
- **Consent Management** and other account options as part of Bubble Protected User Accounts managed by the user's wallet.
- **Proof Of ID**. Using cryptography, zero-knowledge proofs and VCs to provide ID solutions that minimise data exposure (see case study 8.4).
- **Dynamic Data Access**. Harnessing the inherent bubble feature of authenticated state transitions to minimise data exposure in any data transaction between data owner and organisation (see case studies 8.2 and 8.4).
- **Personal Data Monetisation**. Harnessing the inherent bubble feature of accepting payment for access to the bubble's private data with permission from the owner.
- **Compute-To-Data**. Harnessing a bubble's *execute* access permission bit to permit organisations to execute authorised algorithms over a bubble's data with optional payment.
- **Paywall**. Using the features of a bubble to permit access to its data based on payments, subscriptions and NFT ownership (see case study 8.3).
- **NFTs**. Using bubbles to control access to private NFT content based on NFT ownership.

7. Implementation Detail

The core protocols behind Bubble ID and Bubble FS - Proxy IDs and Smart Data Access - are blockchain agnostic. Together they provide the platform's primary innovation.

7.1. Bubble ID

A user's Bubble ID is a set of *Personas* derived from a single private key - the *Genesis Key* - and managed by the user's wallet. The user permits applications to act on behalf of one or more of their personas under strict permissions.

7.1.1 Genesis Key

The root of a user's Bubble ID is a single private key - what we call their *Genesis Key*. Each of a user's personas is owned and controlled by a unique key derived from their genesis key with BIP32. The genesis key's public key and address are never exposed. The *Bubble Dashboard* wallet will randomly create a genesis key for a new user but if preferred they will be able to use an external hardware or software wallet instead.

The genesis key is used to create a new persona and to give the user's wallet - a decentralised application - permission to administer the persona. After this, the genesis key is no longer required until a new persona is needed or the Bubble ID needs to be recovered. Should the user lose access to their devices they can recover all of their personas, applications and bubbles using just their genesis key.

Backup of the genesis key is critical. In web2, password resets due to lost or forgotten passwords are commonplace[10] so a wallet that does not have a simple identity recovery option will be unlikely to break into the mass market. The Bubble Dashboard, with support from Bubble Services, is taking a pragmatic, hybrid approach to identity recovery. While technical users can use existing software and hardware wallets, new users will be offered a choice of backup options, including options that balance security with familiarity and ease of use. Users will be encouraged to keep improving their security level and wallet features may be restricted to higher levels by default. Friend networks will be used for decentralised backup options and Bubble Services will provide a central service for verified recovery that can be used on its own (least secure) or in conjunction with a friend network.

7.1.2 Personas

Personas are individual profiles designed to keep aspects of a user's life separate and reduce the risk of accidentally exposing personal data to the wrong people. Whenever the user interacts digitally using their Bubble ID they will do so as one of their personas. A user can have as many personas as they like - for example one for personal use, one for business use and a couple of anonymous personas for social media. There is no method for an outsider to directly identify personas as being from the same user.

A persona is deployed to a blockchain as an implementation of a Proxy ID smart contract (see 7.1.5). This smart contract is known as the persona's *Persona ID*. The persona contract is designed to delegate roles to decentralised applications and to be administered by a Proxy ID wallet, like the *Bubble Dashboard*. The persona smart contract is deployed by the user's wallet and, with the user's permission, will delegate the admin role to the wallet making the wallet an administrator of that persona. Once deployed, the genesis key is no longer required since the wallet can administer the persona without it. Only if the user needs to recover their account or create a new persona will the genesis key be needed again.

Each persona contract is designed to be the protector of it's own private bubble. The bubble contains all of the persona's data such as it's public ID (nickname and icon), DID documents, verified credentials, personal details and identity documents. Most of the data is private and only available to the user's wallet or genesis key. The bubble is encrypted and stored on a bubble-compatible server of the user's choosing.

7.1.3 Applications

Applications are Proxy ID-compatible decentralised applications that the user connects to their Bubble ID. Through their wallet, the user can authorise an application to act on behalf of each persona under strict roles, giving permission for the application to identify as a persona and to have other capabilities, such as being able to notarise to the blockchain, publish transactions, make payments from the user's wallet, or read or write to the user's bubbles.

Like personas, each application is deployed to the blockchain as a Proxy ID smart contract, known as the application's *Application ID*. The application is given permission to act on behalf of the user's personas by adding the Application ID as a proxy to each Persona ID. Each installation of the application on the user's devices is known as an *Application Instance* and is given permission to act as a proxy for the Application ID. This chaining of Proxy IDs from Persona to Application ID to Application Instance allows applications installed on multiple devices to use their own local private key to act as a user's personas and allows applications and instances to be managed from the user's wallet. Chaining of Proxy IDs and the restriction of roles is described in section 7.1.5.

To connect to a user's Bubble ID, applications must request access through an OAuth 2.0-like request process (figure 4). The request protocol is too complex to describe here and will be specified in a future *Application Request Standard*. The request itself is a JSON file containing descriptive content, the requested roles, and other metadata. It is processed by the user's wallet, which will ask the user for confirmation and to choose which personas to authorise the application to use.

If approved, the wallet is responsible for deploying the Application ID and adding it as a proxy to each authorised Persona ID. The wallet will also create an application bubble (if part of the request). Like personas, the Application ID can be the protector of it's own private bubble, which can be used by the developer to hold application metadata and for syncing application instances across devices. Figure 7 shows two application instances on different devices using their own local private keys to act on behalf of a persona via an Application ID.

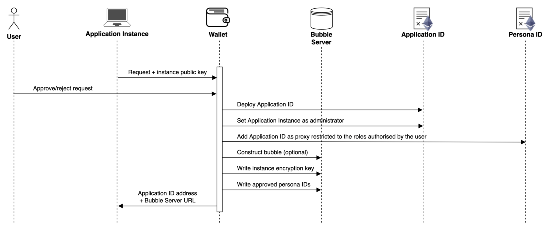

User
Application Instance
Wallet
Bubble Server
Application ID
Persona ID
Request + instance public key
Approve/reject request
Deploy Application ID
Set Application Instance as administrator
Add Application ID as proxy restricted to the roles authorised by the user
Construct bubble (optional)
Write instance encryption key
Write approved persona IDs
Application ID address + Bubble Server URL

Figure 4: *Application request and response. The response can be passed as an OAuth 2.0-like redirect url, via a bubble or in-person via a QR code or NFT.*

7.1.4 Data Shares

A persona is designed to allow the user to share personal data with organisations and other users using bubbles. An organisation can request access to items of a user's personal data by presenting an OAuth 2.0-like request containing a description of the purpose of the data sharing, the parties involved and the terms of use. With the request, the organisation must provide an *Access Control Contract* (ACC - see section 7.2) - a smart contract that implements those terms. The ACC will control the bubble in which the data is shared, enforcing the terms and acting as a service level agreement (see section 7.2.2) between the user, the organisation and any third party data processors involved in the service being provided. From a regulatory perspective, the ACC can enforce the GDPR data sharing agreement[11] and any data processing agreements[12].

Requests are handled by the user's wallet (figure 5). The wallet will let the user select which of their personas they want to use to approve the request. Wallets should consider presenting the request in a form that allows the user to make a quick and informed decision, such as using reputation scores, warning indicators, etc. If the user accepts the request the wallet will deploy the ACC, construct the bubble and copy the requested data from the persona bubble to the request bubble. The requester can automatically detect the deployment of the ACC on the blockchain or can include a response url in the request. The location of a bubble (which bubble server it is stored on) is chosen by the user. Requests can restrict which server or servers can be chosen, allowing organisations to constrain customers to use their own servers or a set of trusted, reliable cloud servers.

Requests are not restricted to online services, they can be used for over-the-counter purposes, like hotel check-in or form filling, via a QR code or NFC.

With bubbles controlled by on-chain ACCs, the user's wallet can query for and display all of a user's bubbles. A goal of the Bubble

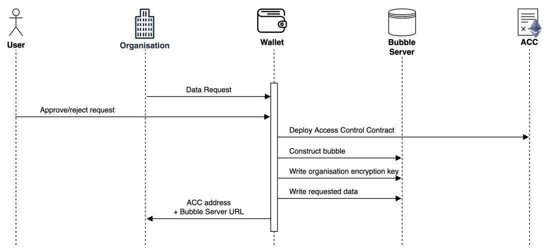


Figure 5: *Data request and response. As for application requests (figure 4), the response can be passed as an OAuth 2.0-like redirect url, via a bubble or in-person via a QR code or NFT. The organisation can also automatically detect the ACC deployed on the blockchain.*

Dashboard is to allow the user to manage their global data footprint directly from their wallet. Provided the terms of the ACC allow it, the user ought to be able to delete a bubble and keep their data up to date directly from their wallet. Updating a piece of personal data in their wallet, like their home address, should automatically update all bubbles that are sharing that data.

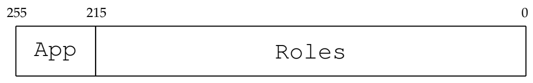
7.1.5 Proxy IDs

Proxy IDs are smart contracts that implement the `ProxyID` interface. The interface has a single function, the implementation of which is at the discretion of the developer.

```
function isAuthorized(
  address requester,
  bytes32 roles
) returns bool;
```

The function takes the public address or Proxy ID of the user requesting authorisation and the roles the user is requesting authorisation for. It is designed to return true if the requested user is authorised to act on behalf of the Proxy ID under all the roles requested.

The `roles` parameter is a bitmap containing a 40-bit *application code* and 216 pre-defined roles.



The *application code* frees developers to define roles specific to their application or to define role standards for general use. Application codes will be registered with and traded using the Bubble Protocol Roles Registry smart contract. Application code zero is registered to Bubble Protocol and used for the Bubble Protocol Role Standard, which defines specific roles used by the Bubble Dashboard wallet and Bubble ID Personas.

Role-Based Permissions

A requested user u_r (blockchain account or Proxy ID contract address) is authorised (A) to act on behalf of a proxy ID under requested roles r_r (bitmap) if:

$$A(u_r, r_r) = \exists a \in U : P(u_r, r_r, a, r_a) \tag{1}$$

$$P(u_r, r_r, a, r_a) = (a = u_r) \wedge (r_r \wedge r_a = r_r) \tag{2}$$

where U is the set of blockchain addresses that have authorisation to act on behalf of the Proxy ID and r_a is a bitmap describing all the roles that address a is authorised to perform.

Chaining Proxy IDs

If we extend the definition of U so that it can also contain Proxy IDs (addresses of Proxy ID smart contracts) then we can redefine (2) to chain Proxy IDs together:

$$P(u_r, r_r, a, r_a) = ((a = u_r) \vee A_a(u_r, r_r)) \wedge (r_r \wedge r_a = r_r) \tag{3}$$

where A_a is the function A within Proxy ID contract at address a.

The definition in (3) ensures child IDs have at most the same permissions as their parent. For example, if key k has admin rights (all roles) over Proxy ID P_2, and P_2 has the *identify_as* role within Proxy ID P_1, i.e.:

$$k \xrightarrow{admin} P_2 \xrightarrow{identify\_as} P_1$$

then k will have only the *identify_as* role within P_1.

```
contract MyProxyId is ProxyId, Proxyable {

  struct Proxy {
    address id;
    bytes32 roles;
  }

  Proxy[] proxies;

  function isAuthorized(address requester, bytes32 roles) returns bool {
    for (uint i=0; i<proxies.length; i++) {
        if (_isAuthorizedFor(requester, roles, proxies[i].id, proxies[i].roles)) return true;
    }
    return false;
  }

  ...

}
```

Figure 6: *Example of a chained ProxyId implementation using the* `_isAuthorizedFor` *function inherited from the* `Proxyable` *contract. In this case, the ProxyId has an array of accounts and/or Proxy IDs that can act on its behalf, each with a specific set of roles.*

Implementing Proxy ID Contracts

Proxy ID contracts need only implement the `isAuthorized` function. The developer is free to implement this however they like but Bubble Protocol provides the function `_isAuthorizedFor`, which implements equation (3) and is available as a library or can be inherited from the `Proxyable` abstract smart contract. See the example `isAuthorized` implementation (figure 6) and the example Access Control Contract (figure 10).

7.2. Smart Data Access

Each bubble is a separate secure container stored on any private *'Bubble Server'* whose access is controlled by exactly one smart contract - it's *Access Control Contract* (ACC). The smart contract's address and the host blockchain's network and chain IDs together form the bubble's globally unique identifier (although not where the bubble is stored). Since a bubble contains files and directories, the smart contract controls access at the file level, implementing the bubble's access control list (ACL) - in this case the read, write, append and execute (xrwa) permissions for each file and directory in the bubble. Permissions are granted by the smart contract on a user-by-user basis and enforced by the bubble server. The user is identified by their public key (account address) recovered from the signature in their request to the bubble server.

7.2.1 Access Control Contracts

By integrating Proxy IDs and roles into the smart contract the developer can control access for specific Bubble IDs, groups of IDs and more complex structures (see figures 7 and 3 as examples). Access can be granted based on time, smart contract state, NFT ownership, blockchain oracles - anything that can be described in code on the platform deployed. Access can be modified by user interaction - authorised users can transact with the contract to transition the contract state, using Proxy IDs to authenticate the user. This is a more flexible permissions scheme than ACL alone, being more akin to ACL + shell scripts. Being on the blockchain the access control state is transparent to all parties concerned, providing trust and visibility of any state transitions. In essence, Access Control Contracts (ACCs) provide a way for developers to encode the life cycle of the bubble's data, and to innovate with dynamic, user interactive, transparent access controls. For an example of this see case study 8.2.

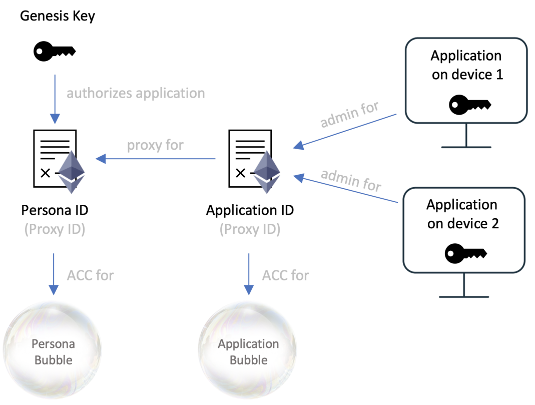


Figure 7: *Extension of figure 2 showing Proxy ID contracts acting as Smart Data Access Contracts*

The SDA protocol specifies the ACC interface. The interface is very simple, the primary method being the *getPermissions* function. The implementation of this function is up to the developer.

```
function getPermissions(
    address requester,
    uint file
) returns byte;
```

The function takes the public address of the user requesting the data, and the id of the file within the bubble. It returns a bitmap containing the permissions `d---xrwa`.

If the developer incorporates Proxy IDs into this function, access can be granted to a specific user and role(s). By proxy, this means any application the user has permitted to act under those roles will have access. The example in figure 10 shows an implementation that controls access to NFT images, restricting access to NFT owners and the owner of the bubble.

7.2.2 Service Level Agreements

Being on-chain, the ACC is transparent and immutable. It is deployed by the data owner's wallet or with signed permission from the wallet. It can therefore be considered a service level agreement between the data owner and the data requester(s) and used as an irrepudiable way to enforce data protection rights. The developer can encode the owner's rights[9] to rectification, erasure and portability directly into the smart contract, letting the bubble itself automatically manage some of the compliance process. Termination of the smart contract (if permitted) can act as a withdrawal of consent for the use of the data in the bubble. Similarly, transitioning the smart contract to a different state can modify consent.

7.2.3 Bubble Servers

With all the access control intelligence placed on-chain, bubble servers become little more than unintelligent, network storage devices. Their primary function is to serve data access requests based on the permissions returned by the smart contract, see figure 8. Bubble servers are therefore essentially generic bubble storage servers. Any server can, in principle, store any bubble. Servers have no knowledge of the purpose of the bubble or who owns it. This separation of concerns combined with the anonymity of Proxy IDs and the encryption of the bubble data helps to minimise opportunities for abuses of power.

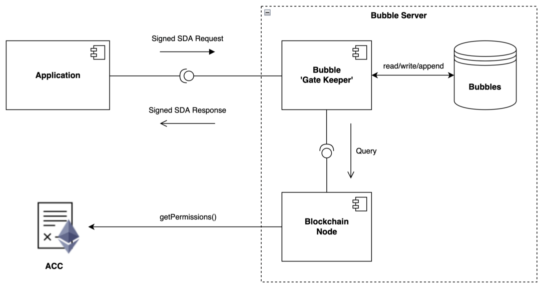


Figure 8: *Bubble Server Architecture & Access Request*

Smart Data Access includes an application layer protocol for DApps to request access to bubble data. *Smart data access requests* sent by the client are JSON-formatted packets containing the request type, ACC address, network id, chain id, data (if a put request) and signature of the requester. They are used to create and delete bubbles and for read, write, append, delete and execute file requests. The bubble server recovers the signatory address from the packet signature and queries the ACC at the address given in the request. If the ACC permits it, the request is served, otherwise it is rejected.

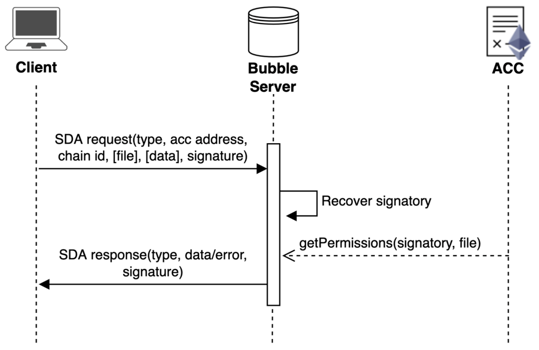

Client
Bubble Server
ACC
SDA request(type, acc address, chain id, [file], [data], signature)
Recover signatory
getPermissions(signatory, file)
SDA response(type, data/error, signature)

Figure 9: *SDA request and response. The bubble server contains a blockchain node (or uses a 3rd party service) to query the blockchain.*

The Bubble FS initiative provides an implementation of the Bubble Protocol for easy deployment of a bubble-compatible server - the *Bubble Gate Keeper* library. The Bubble Private Cloud used by the Bubble Dashboard uses this library.

Developers and miners are encouraged to deploy their own cloud servers to give users diverse storage options and thereby support the decentralisation of the platform. Users and developers can pay bubble servers for hosting their bubbles by allowing the server to claim payment directly from the ACC.

Organisations are encouraged to integrate the Bubble Gate Keeper library into their existing infrastructure and experiment with private bubbles for customers, clients, suppliers and employees.

Bubble Protocol places no restrictions on how data is stored within a bubble server. The storage technology and the security, backup and redundancy solutions are left to the server developer. It's feasible this could lead to a marketplace of cloud bubble services, each competing on cost, security, reliability and privacy innovation.

Execute Permissions

Bubble FS will eventually allow files to be executed to perform actions over the data in the bubble. The execute permission is built into the smart data access protocol. Bubble Servers can choose to support the feature or not. The file will be executed in an isolated environment with access only to the data from the bubble and to any requested remote oracle services. To execute a file the client will provide a list of bubble files and directories that it wants to interact with and the read, write or append permissions it wants for each. The Bubble Server will query the bubble's ACC to confirm the client key has all the necessary permissions before copying the data to the isolation environment and executing the file with any parameters passed by the client.

Executable files allow bubbles to support complex server-side functions without needing to copy all data to the client. They also provide support for compute-to-data use cases, optionally charging clients via the ACC before granting permission.

7.2.4 Bubble Storage Networks

Since the design of a bubble server is not prescribed, developers are free to explore different storage architectures. One such architecture is present in the Bubble IPFS Gateway and Bubble Skynet Gateway servers, which use smart data access as gateways to existing decentralised storage networks. Instead of storing bubble files locally, these bubble servers store them on the storage network and just store the network's content IDs. When a file is read by a client, the data is read from the network and passed to the client without exposing the file's content ID. In effect, this allows users and developers to control access to files on decen-

```
function getPermissions(address requester, uint file) returns byte {
  if (_isAuthorizedFor(requester, ADMIN_ROLE, _ownerId)) {
    return READ_BIT | WRITE_BIT | APPEND_BIT;
  }
  if (_isAuthorizedFor(requester, IDENTIFY_AS_ROLE, _nftContract.ownerOf(file)) {
    return READ_BIT;
  }
  return NO_PERMISSIONS;
}
```

Figure 10: *Example ACC giving owners of an NFT read access to their NFT image contained in a file named after the token id. The NFT can be owned by a Proxy ID or standard account. In this case, the contract owner (also a Proxy ID) has full access to all files.*

tralised storage networks by wrapping those files in a bubble.

Another potential architecture is for developers to join forces with others to create a permissioned network of independent bubble servers. Such networks would help to further decentralise the platform and to improve backup and redundancy. The Bubble FS and Bubble Privacy initiatives will explore options for creating such a network.

8. Case Studies

8.1. OpenSig: Using a bubble for DApp storage

The client-server model is the standard architecture for web2 applications. Web3 decentralised applications, on the other hand, use a combination of smart contracts and decentralised storage networks plus other decentralised services to provide the backend.

DApps installed on a single device - whether an application, a web app or a browser extension - can simply use local storage to persist user settings, usage history and other data between user sessions. Conversely, for DApps to function smoothly across multiple devices, off-device storage is needed. Decentralised storage networks, like IPFS and Filecoin, are currently used to provide this storage, however each technology has its positives and negatives. For example, IPFS is free and simple to use but cannot promise that the data will be stored for long; Filecoin, built on IPFS, is incentivised for long term storage but is tailored for large data sets and does not guarantee fast access. Developers must think carefully over their choice of storage solution.

OpenSig is a digital signature DApp that lets users sign documents on the blockchain using their Bubble ID personas. OpenSig is designed to work across a user's devices, such as on their computer, at work and on their mobile phone. To accomplish this it needs cloud storage to backup signed documents and to sync between devices. It uses a bubble for this purpose.

Each installation of the OpenSig DApp connects to the user's Bubble ID via the application request process described in section 7.1.3, which permits all installations to authenticate as a single Application ID (see figure 11). This Application ID also controls access to an application bubble, giving each installation administrator rights over the files in the bubble. The bubble is accessible only to the user's OpenSig installations and their wallet. Data in the bubble is encrypted.

Through the Bubble Dashboard (or other Proxy ID wallet), the user can choose which bubble server they trust to hold their bubble. They can use a cloud bubble service or their own server. Similarly, organisations can use their own private servers or integrate bubbles into their document control system to ensure documents signed by employees are kept in house.

DApp permissions are granted by the on-chain OpenSig Application ID smart contract. The Application ID controls which installations

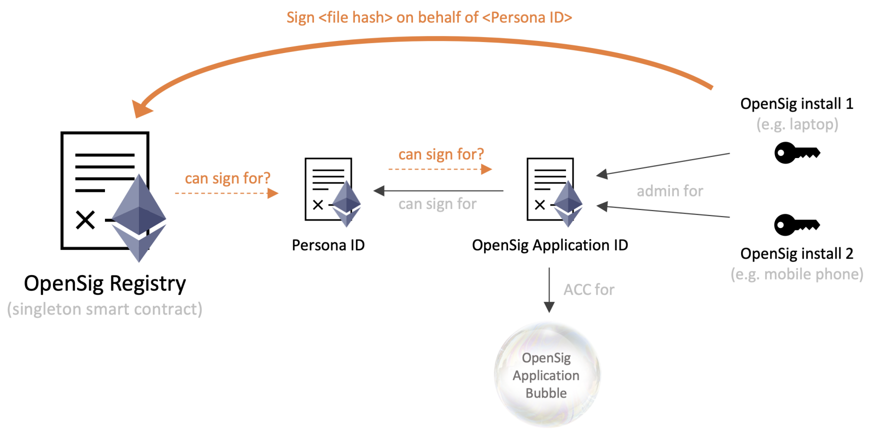


Figure 11: *Case Study 8.1: OpenSig smart contract architecture and process flow. The user has OpenSig installed on two devices. Both devices share the same Application ID and bubble but use their own local private keys. In this example, the user signs a document using the OpenSig DApp installed on device 1. The DApp uses its local private key to register the document hash with the global OpenSig Registry on behalf of the user's persona. Through the two chained Proxy IDs (the Persona ID and the Application ID), the OpenSig Registry smart contract confirms that the key is authorised to sign on behalf of the persona. The DApp saves the document to the bubble so that it is privately backed up and synced across all devices.*

(which local private keys) can sign and access the bubble. The user's wallet has administrator rights over the Application ID allowing the user to manage permissions for installations and for the whole DApp. From their wallet they can change which personas the DApp can authenticate as, disable particular installations or terminate the contract to remove permissions for the whole DApp. Disabling an installation will remove it's permissions from the on-chain Application ID preventing it from signing as the user's persona and from accessing the bubble. Terminating the contract will disable all installations and delete the bubble, permanently deleting all data within.

This example demonstrates the use of a bubble for a simple decentralised application. It describes how the user, via their wallet, can control which of their DApps are authorised to act as which of their personas, and can manage their DApp installations across all their devices.

8.2. Broadvale Football Club: Onboarding new members with Bubble Protected User Accounts

Broadvale is a fictitious small town football club with local amateur adult and youth teams, and training for all ages. It is affiliated with a national football association, NAFA, which provides standards and insurance for clubs and paid up members. Broadvale is working with NAFA to experiment with digital identity to help with player records, consent management, communication and data protection regulation compliance.

To join Broadvale, new club members currently have to fill in a paper form and return it to the club. The form is checked and the data is manually entered into NAFA's national database by a club administrator via an online portal. The data is held on Amazon AWS.

Broadvale and NAFA are experimenting with using Bubble ID to onboard new members. Prospective members can use their Bubble ID mobile wallet to scan a sign-up QR code posted at various locations around the club, or can click the link on the website to open it in their desktop wallet (the wallets are synced with each other). The QR code encodes a DID referring to a DID document containing a data request (see section 7.1.4). The member's wallet displays the request details, indicating clearly that it has verified that the signature in the request is from NAFA. The wallet displays the terms of membership along with the sign-up form and prompts the user to choose which of their personas they want to use to sign up. Once chosen, the wallet automatically fills in the form with the user's personal data read from the persona's bubble. Finally, the wallet prompts the user to choose which bubble server they trust to hold their member's account. In this case, the data request limits the user to a handful of servers that NAFA trusts for security and reliability. The user chooses the Bubble Private Cloud.

After confirming the user wishes to complete the sign-up process, the wallet deploys a new access control contract (ACC) on-chain (see section 7.2.1) as specified in the data request. In this case, the ACC is based on the Bubble Protected User Account template designed by Bubble Protocol. The ACC permits NAFA, Braodvale and NAFA's insurance company to access just the information they need during the lifetime of the membership. Once deployed, the wallet creates a bubble on the Bubble Private Cloud and writes the online form data to the bubble along with their consent to be contacted by email. The bubble is encrypted.

At NAFA, their servers are monitoring the blockchain for newly deployed contracts. The new ACC is detected and, from a server ID code read from it, the Bubble Private Cloud is determined as the bubble's server location. NAFA records the new member in it's member's database storing just the ACC's blockchain address. For data protection purposes the user's data will never be stored on NAFA's servers and will instead be read from the bubble whenever it is needed. The NAFA servers read the user's name and consent options from the bubble and send the user a personalised welcome email[1].

At the same time, NAFA's insurance company also detects the new ACC and automatically processes the user's data read from the bubble. The company confirms it accepts the new member by transitioning the ACC to the `Insured` state, a transition that only the insurance company is permitted to make. Being visible on the blockchain, this transition triggers NAFA's servers to send the user a second email confirming their acceptance as a new member. It also sends an email to Broadvale's club administrator informing them of the new member, whose name and contact details they can view from a special page on the online portal using their Bubble ID for authentication. The onboarding process has taken just a few minutes.

This case study describes only the onboarding process. It could continue to describe how the bubble will protect the member's data throughout their membership, allowing them to keep their personal data up to date in line with data protection regulations, and deleting their data permanently if they terminate the bubble from their wallet[2]*. The bubble will be used to hold all metadata related to the member, such as courses, achievements, fitness data and match data. In essence, their entire account will be held in and protected by the bubble throughout the life of their membership.*

[1]The example uses email to communicate with the member. Further improvements to privacy can be made by using the bubble itself as a vehicle for secure messaging. This approach prevents spam and phishing attempts since all parties to the bubble are securely authenticated. Our early proof-of-concept dashboard includes such a secure messaging feature. It uses a secure messaging protocol that would, in principle, allow a messaging DApp of the user's choosing to read and post messages within all of a their compatible bubbles. It would provide the user with secure, private communications with all companies they engage with.

[2]Should NAFA or the insurance company have a legal obligation to retain the user's data for a period of time after membership has ended then this lock-up period can be built into the bubble's ACC and the data automatically deleted when it expires.

8.3. The Daily Chronicle: Paywalled Web Content

The Daily Chronicle is a fictitious online news outlet whose primary reader interface is its website. Readers can subscribe on a monthly or annual basis with their first month free. With rising pressure from digital-born sites affecting overall sales, The Daily Chronicle wishes to explore how web3 can provide new ways for readers to pay for online news.

Developers at The Daily Chronicle setup Bubble Protocol alongside their existing infrastructure to ensure no impact on their existing customer base. They add a new *Subscribe With Bubble* method alongside their existing subscription methods offering new daily subscriptions, weekly subscriptions and a *click-to-view* micro-payment option to access individual articles [3].

To implement this, the developers integrate Bubble Protocol with their article database, creating a Bubble Server (see section 7.2.3) that exposes each article's content as a separate file within a bubble (see section 4.1). To control who has access to each article they deploy a bespoke Access Control Contract (see section 7.2.1) derived from Bubble Protocol's paywall ACC template, which implements the subscription model. Finally, they paste the off-the-shelf *Subscribe With Bubble* button code into their subscription content and insert a new HTML `div` tag in their article pages, wrapping their current HTML content in the tag. The tag wrapper contains metadata about the bubble and the article's file ID but is transparent to existing customers who experience the website just as before.

Now, when Bubble users visit a page they have the new subscription options available to them. No account creation is needed to subscribe and no login is required once subscribed. Subscriptions are handled by the bubble's ACC, which accepts payments in Bubble tokens. No payment infrastructure is needed on the website and no credit cards are needed - the user's Bubble Pass browser extension pays the ACC directly and the ACC's code implements the subscription process on the blockchain.

When visiting an article page, the user's Bubble Pass browser extension attempts to read the article directly from the bubble on page load. Bubble Pass authenticates the user with a signed read request to the Bubble Server, which in turn checks the user's access permissions against the ACC on the blockchain. The ACC ensures only subscribed users are granted access to the article and handles subscription expiry. If successfully read, Bubble Pass replaces the tag content with the article content. If unsuccessful, the tag is untouched and the original content is displayed.

This paywall example can equally be applied to other types of online content, such as music, images, videos and files, allowing sites to provide paywalled content with no need for any payment infrastructure. NFTs can be used instead of direct user authentication to allow access to be passed or sold on to others. Indeed, Bubble Protocol's project NFT web page works this way. In principle, any access control model that can be implemented in a smart contract can be used to control a paywall.

8.4. Case Study: Over-the-counter Proof of ID

The Gritz is a fictitious hotel in the heart of London. In accordance with local immigration law[13], The Gritz must keep a record of the identity of foreign travelers for 12 months from the time of check-in and provide it to the police on request. To comply with this law, the desk clerk requests the traveler's nationality on check-in and takes a photocopy of their passport or identity document. The paper copy, along with the check-in details, are held in the hotel filing system behind the check-in desk until it is manually purged 12 months or so later.

The Gritz recognises two problems with this

[3]These are basic subscription methods. More complex examples can be explored such as allowing blockchain-based news aggregators to algorithmically pay a proportion of their member's fees for access to the site's articles, whether directly on The Daily Chronicle's website or embedded in the news aggregators app.

process. Firstly, it slows the check-in process, which in peak times can lead to long queues and frustrated customers. Secondly, there is an unnecessary privacy risk with having personal identity data available to anyone with access to the hotel office, which requires management through the hotel's GDPR compliance process. The hotel manager wants to explore the benefits of using web3 technology to streamline the process.

The hotel installs a *fast check-in* process provided by Bubble Protocol. This consists of a desk-mounted contactless reader and a behind-the-desk tablet screen. Travelers choosing to *Check-In With Bubble* hold their phone to the reader and approve the data request that appears on the Bubble ID wallet on their phone. The desk clerk confirms a photo of the traveler on their tablet screen and hands them the room key[4], welcoming them by name if the traveler has opted to share that information. If the traveler is not a UK citizen, their identity data (a verified credential) is written to a dedicated bubble that protects the data from unauthorised access and deletes it automatically after 12 months. Access to the identity data is only available to the police and only after they transition the bubble's on-chain ACC to gain access (a visible and auditable event).

This process is designed to check in the traveler quickly without any form filling and without exposing any information about the traveler to the hotel staff whilst complying with the law and simplifying GDPR compliance. Throughout the life cycle of the bubble the traveler's Bubble ID wallet monitors the bubble state on the blockchain informing the traveler of any state transitions, including expiry. Through their wallet the traveler has visibility of this and every other bubble they are a part of. The traveler can update, delete or transition each of their bubbles at any time, provided it's ACC permits it.

[4]In future it may be possible for the traveler to use their mobile phone as a room key, authenticating themselves with their digital ID.

References

[1] Stanford Encyclopedia of Philosophy. Autonomy in Moral and Political Philosophy. `https://plato.stanford.edu/entries/autonomy-moral/`.

[2] Privacy International (2018). It's about human dignity and autonomy. `https://privacyinternational.org/long-read/2208/its-about-human-dignity-and-autonomy`.

[3] Information Is Beautiful. World's Biggest Data Breaches & Hacks. `https://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/`.

[4] Wired (2019). How Cambridge Analytica Sparked the Great Privacy Awakening. `https://www.wired.com/story/cambridge-analytica-facebook-privacy-awakening`.

[5] Amnesty International (2019). 'The Great Hack': Cambridge Analytica is just the tip of the iceberg. `https://www.amnesty.org/en/latest/news/2019/07/the-great-hack-facebook-cambridge-analytica/`.

[6] Center For Data Innovation (2020). Two Recent Cases Show How The GDPR Is Failing European Businesses. `https://datainnovation.org/2022/02/two-recent-cases-show-how-the-gdpr-is-failing-european-businesses/`.

[7] Dr. Gavin Wood (2014). ĐApps: What Web 3.0 Looks Like. `http://gavwood.com/dappsweb3.html`.

[8] 3BoxLabs (2022). Building capability-based data security for Ceramic `https://blog.ceramic.network/capability-based-data-security-on-ceramic/`.

[9] European Union. GDPR Chapter 3: Rights of the data subject `https://gdpr.eu/tag/chapter-3/`.

[10] HYPR (2019). HYPR Password Study Findings `https://blog.hypr.com/hypr-password-study-findings`.

[11] UK Information Commissioner's Office. Data sharing agreements `https://ico.org.uk/for-organisations/guide-to-data-protection/ico-codes-of-practice/data-sharing-a-code-of-practice/data-sharing-agreements/`.

[12] European Union. What is a GDPR data processing agreement? `https://gdpr.eu/what-is-data-processing-agreement/`.

[13] UK government. The Immigration (Hotel Records) Order 1972 `https://www.legislation.gov.uk/uksi/1972/1689/made`.