Identity Management: State of the Art, Challenges and Perspectives *

Tore Kasper Frederiksen[1], Julia Hesse[2], Anja Lehmann[3], and Rafael Torres Moreno[4]

[1] Alexandra Instituttet, Aarhus, Denmark
tore.frederiksen@alexandra.dk
[2] IBM Research – Zurich, Switzerland
jhs@zurich.ibm.com
[3] Hasso-Plattner-Institute, University of Potsdam, Germany
anja.lehmann@hpi.de
[4] University of Murcia, Spain
rtorres@um.com

Abstract. Passwords are still the primary means for achieving user authentication online. However, using a username-password combination at every service provider someone wants to connect to introduces several possibilities for vulnerabilities. A combination of password reuse and a compromise of an iffy provider can quickly lead to financial and identity theft. Further, the username-password paradigm also makes it hard to distribute authorized and up-to-date attributes about users; like residency or age. Being able to share such authorized information is becoming increasingly more relevant as more real-world services become connected online. A number of alternative approaches such as individual user certificates, Single Sign-On (SSO), and Privacy-Enhancing Attribute-Based Credentials (P-ABCs) exist. We will discuss these different strategies and highlight their individual benefits and shortcomings. In short, their strengths are highly complementary: P-ABC based solutions are strongly secure and privacy-friendly but cumbersome to use; whereas SSO provides a convenient and user-friendly solution, but requires a fully trusted identity provider, as it learns all users' online activities and could impersonate users towards other providers.
The vision of the Olympus project is to combine the advantages of these approaches into a secure and user-friendly identity management system using distributed and advanced cryptography. The distributed aspect will avoid the need of a single trusted party that is inherent in SSO, yet maintain its usability advantages for the end users. We will sketch our vision and outline the design of Olympus' distributed identity management system.

Keywords: Privacy · Security · Identity management · Digital identities · Distributed cryptography

* This is a workshop paper for the Olympus project. Olympus has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 786725.

1 Introduction

Data has become the new oil: the advent of advanced AI algorithms for data analytics and cheap storage has allowed analysis based on millions of individuals' personal information. This includes health, location and perhaps most importantly; any kind of online activity. The latter is of particular importance due to the sheer magnitude of data in that category. An important example in this setting is Facebook, who with its user base of over one billion, holds a treasure trove of personal information, ready to be used for social analysis [24] and manipulation. In this setting the Cambridge-Analytica scandal provides interesting insight into how high a value personal data, and in particular subjective attributes, such as likes, can have [5].

A crucial lesson learned from this scandal is that we must become more aware of how much personal information is available about us online and how it is possible for us to limit this. One approach is of course to simply not share anything with sites we do not fully trust. However, even trusted sites suffer from breaches that leak personal information [19, 22] and sometimes this information is even collected without our knowledge [26]. If we still wish to be part of the connected world the Internet facilitates then we cannot be completely safe and protect against all non-consensual collection of information. However, we can try to prevent theft of personal information by limiting the locations at which our data is stored and by tuning how easy it is to gain access to this information without a complete take-over of the service provider.

The Need for User Authentication. Personal data at different service providers are usually protected through some sort of authentication mechanism which allows users to prove that they are who they say they are. Such authentication is done by proving ownership of a given username, meaning that a user must prove that it holds some secret information that was used during account creation.

This secret information is most often a password that was picked by the user. Unfortunately, despite advised otherwise, people tend to either use low-entropy passwords or reuse their passwords (or slight variations thereof) at different service providers [31]. This introduces a large attack surface for an adversary wishing to take-over a user's account; he/she could try to guess low entropy passwords or try to find users who reused a password between a fully compromised service provider and an un-compromised service provider.

In fact, the latter not only leverages the users' tendency of password re-use, but also the availability of massive password leaks through compromised service providers. More than one billion personal data records including email address and password combination have been reported stolen to date [27]. Even when properly salted and hashed, the low entropy in human-memorizable passwords makes it trivial to brute-force the plaintext passwords using modern hardware. Increasing entropy through measures such as two-factor authentication can be seen as a trade-off between security and usability. Such methods are often not adequately efficient to handle for users quickly wanting to access a service.

Using a cryptographic key for a digital signature scheme, as is the case when authentication is based on X.509 certificates, or an access token authorized to the

user through a different service provider, known as an Identity Provider (IdP), alleviates some of the worst security issues with passwords. In particular these approaches do not require users to trust every single service provider they wish to access, to keep our password safe.

However these solutions have other drawbacks: Certificates, for example, are not very user-friendly and require users to manage large, non-memorizable keys between all the devices they use. When the certificates contain user-specific attributes they also require disclosure of all these attributes with every usage, which violates the privacy paradigm of data minimization. Online IdPs and Single Sign-On (SSO) are quite complementary: they are easy and convenient to use, as users only need to authenticate to the IdP which then issues short-term authentication tokens towards service providers. The IdP can thereby certify only the minimal amount of attributes needed for each access request. On the negative side, the IdP must be a fully trusted entity, as it learns about all the services its users accesses, and — if compromised — could impersonate the users towards all service providers.

Modern cryptography has afforded another approach to identity management, handling some of the issues with passwords, certificates and IdPs. This approach is known as Privacy-Enhancing Attribute-Based Credential (P-ABC) [10, 3, 9, 28]. A P-ABC scheme allows a user to hold a special token, known as a credential, which contains an assertion on the user's identity and features from a trusted provider. The user can use the credential to generate single-use tokens for authentication towards different service providers. Like for certificates and IdPs this approach also avoids having to trust all the service providers a user wishes to access *and* it handles the privacy issues that are otherwise associated with IdPs and conventional certificates. Unfortunately, P-ABCs struggle when it comes to user-friendliness: they have the same limitations as certificates, and on top require complex cryptography and policy management that needs to be handled by users and service providers.

The Olympus project is aiming to combine the best of each to create a more secure approach to online identity management, while avoiding the drawbacks of the current solutions. The idea behind it is very simple: It follows the SSO paradigm, but instead of relying on a single trusted IdP, the trust is distributed among a set of servers, which might be run by different institutions. This removes the single point of failure in a system where compromise of the single trusted server has devastating consequences for the user [19]. Further, using techniques from P-ABC systems, one can realize the distributed IdP in an *oblivious* manner, i.e., the IdPs won't be able to learn or track users' online activities.

Roadmap. The document starts with an introduction to identity management and deployed technologies such as X.509 certificates and Single Sign-On (SSO) in Section 2. Afterwards, we discuss the idea of P-ABCs in Section 3, which constitutes a more privacy-friendly but also costly way for the user to demonstrate her identity herself. Lastly, we sketch the project's vision of creating privacy-friendly *and* user-friendly identity management in Section 4, including a summary of the discussion at the workshop.

2 Identity Management and Existing Solutions

As things are now, most people have several different online accounts with several different providers, each offering a specific service which requires knowledge of some specific attributes about its user. For example, Tinder wants pictures, age, location, residency, occupation and sexual preferences, an airline wants knowledge of your residency, citizenship and passport number and Facebook wants whatever you are willing to give. These are just examples of a few of the potential online accounts we can have. Most people have many such accounts and thus must repeatedly supply the same information/personal attributes to each of these different providers.

2.1 The Problem with Online Identity

Besides the annoyance of having to supply the same information several times to different parties, the main issue with this is how our information is protected at these sites. The trivial approach to account management is to pick a username and password for each account and then upload attributes to the provider. However, this yields significant issues in relation to breaches and linkability as discussed below.

Breach Issues. Despite the risk of heavy fines through legislation such as the GDPR, some providers do not ensure the safety of personal data and might, either through negligence or financial motivation, leak users' personal information [5]. This might in particular be true for smaller and more sleazy providers. However, sharing of personal information to a provider is not the only issue. A leakage of an account database, even without personal information, can have tremendous consequences. A database leakage of username and password pairs (regardless of whether hashing was used) can provide an adversary amble opportunity to learn a user's password for another provider. If the user has reused its password, this can then be used to compromise the user at other sites who might otherwise employ good security measures. However, once an adversary has the password, even strong security measures might not prevent illegitimate access.

In the recent years, companies have started to add measures to prevent illegitimate take-over of accounts, *even* if the adversary is in possession of a user's password. These include things such as two-factor authentication and machine learning. However, there are not employed globally and may not be sufficient to quench all compromises. Especially not targeted attacks, where combinations of security issues might also cause a compromise, even without a single compromised password [21].

Linkability Issues. The fact that the same personal attributes are stored at distinct providers allows collaborating providers to link users. The most simple way of linking a user is directly through the same username/mail address. However, the linking can also be done by simply comparing attributes. For example, gender, age and address is often enough to uniquely identify an individual.

2.2 Current Identity Management Solutions

In the following we will discuss some of the currently used ways of doing online identity management in a more integrated fashion than simply constructing new username/password-based accounts at every provider.

2.2.1 X.509 Certificates

It is possible to use a standard digital signature scheme in a Public Key Infrastructure (PKI). This can either be based on a global PKI (as is for example done on the web through root Certificate Authorities (CAs)) or based on a self-constructed PKI within a network or organization. In this setting all service providers will trust root certificates which are used to sign one or more intermediary CA certificates. Each of these intermediary CA certificates are used to sign certificates for each user on the network. Such a user certificate contains a handle on a specific user and could also include specific attributes asserted by the intermediary CA. When a user wishes to use a certain service provider it sends it an access request. The service provider then returns a message which it wishes to get signed by the user, in order to verify its identity. The user can then use the private signing key associated with its certificate, and return a signature on the message requested, along with its public certificate to the service provider. The service provider can then verify the *chain of trust* of the certificate. That is, if it trusts the root CA and the certificate of the intermediary CA. If so, it can accept the identity of the user.

Advantages. The main advantage of this approach over the username/password approach is that this offers a minimal need for trust. It is only necessary for the user to trust the CAs, in particular a compromise of a service provider will not yield any adversarial advantage in user impersonation. That said, if the signed info the user provides to the service provider includes personal attributes of the user and the service provider stores these offline, then we still have the issue that a compromise of the service provider would leak the user's attributes.

A great advantage of this approach however is that the user is only dependent on the CA being online when it constructs its certificate. That is, the user and the service provider are the only parties required to be online when a user wishes to sign in to the service provider.

Problems. Despite being simple and easy to implement on top of the PKI we already have in place in all web browsers, this solution still has several significant issues that make it unsuitable to be used in a non-enterprise setting.

First of all, all different service providers must agree to use this system (although any approach besides the individual username/password approach will require this) and to trust any *credible* intermediary CA that a user could use. However, a more pressing issue is that an individual user now has responsibility for its own private keys. That is, if they are lost the user will not be able to sign in anywhere and would need to create a new account. Even creating a new account would still pose a problem as an adversary would now be able to claim

that they lost the private key of a legitimate user's account. Thus the request to create a new account (and close the old one) must be verified. Still, in case it is a sincere request then the legitimate user will not have its private key anymore and thus will not be able to verify the request. Even if a user-friendly solution is constructed to this problem, a perpetual issue of any PKI system still remains; distribution of revocation information. Specifically the service providers must have a method to check whether a user's certificate has been revoked. This could be done using an online service. However, that would remove the main advantage of this approach; that pervasive availability is not otherwise necessary.

Perhaps the biggest issue with using this approach with regular end-users is that real-world evidence has shown [14] users are terrible at protecting and remembering secrets. Furthermore, as most users have more than a single device, it would require that the private key must be distributed across the user's different devices. Thus the usability of this solution leave a lot to be desired. Finally we note that unless the user has a pseudonym and certificate for each service provider, the issue of linkability remains since the public certificate of the user will allow for unique identification across service providers.

2.2.2 Federated Identity

Instead of ensuring security using the weakest link among all the service providers, we might instead choose to centralize trust to a single, strongly trusted party. This is the idea behind using a *Federated Identity Provider* (Figure 1). Basically we store all the attributes associated with a user at a trusted identity provider (IdP), and service providers rely on identity assertions of the IdP when granting users access to their services.

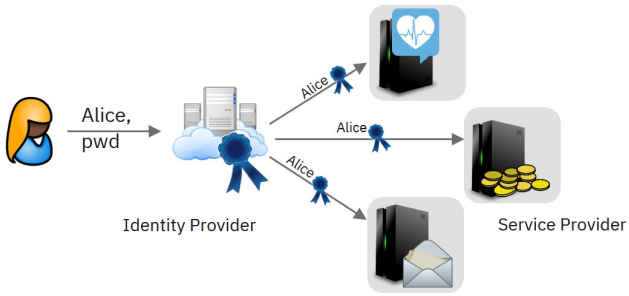

Fig. 1. Federated Identity Management

Whenever a user wishes to sign in to a service provider (perhaps including some attributes) it establishes contact with the service provider, who returns a request of the user's identity (and perhaps certain attributes). The user then checks the request and then signs in to the IdP (using a username/password approach). After signing in, it passes on the request to the IdP. The IdP then constructs a token based on the request and a timestamp (and potentially relevant attributes), and returns it to the user. The user can then pass it on to the service provider. The service provider can then verify the token either based on a

signature of the IdP (known as a *bearer token*) or through a round of communication with the IdP. If the verification goes through and the attributes that have been asserted fit the information the service provider requires, then the service provider accepts the user and signs it in. This process is sketched in Figure 2.

Federated Identity can, under a liberal definition, be considered *Single Sign-On* (SSO) as we only require signing on to the IdP in order to provide authentication for other servers (the service providers). However, under the "conservative" definition of SSO, it is required that a user only signs in once to the IdP, no matter how many different service providers it wishes to access. However, since tokens are often going to be short-lived and specific to a single service provider it is generally required of a user to contact the IdP every time it wishes to sign on to an new service provider (although it might not require singing in to the IdP again if it already has an open session).

Advantages. Similar to the X.509 solution, the main advantage of this approach is that we remove the requirement of trust of every service provider we wish to use and instead centralize the trust at an IdP. But also similar to the X.509 solution is that user attributes might still be stored at a service provider and could thus be leaked in case of a service provider compromise.

A great advantage of this approach, handling one of the main shortcomings of X.509, is that there is no need to store anything confidential user-side. Everything is based on a *single* username and password. This removes the burden of the user to manage secrets, or even multiple passwords, and allows it to sign in on any, possibly new, device.

Furthermore, since log-ins are now performed by signed single-use tokens that have a short validity only, this solution removes most of the complication of providing a reliable revocation mechanism across all service providers. Only the account with the IdP must provide an appropriate revocation or closing mechanism, which will automatically disable all other accesses. Since all relevant user attributes are now stored at a central IdP, it also removes the burden of the user to repeatedly upload and authenticate these with every service provider.

However, even more advantageous is the fact that since a trusted party (IdP) is now in possession of authenticated attributes, it can facilitate a great level of granularity on these in requests. Say for example that a user is born November 26, 1978. This might be asserted by a government authority and shared (and verified) by an IdP. Now when a service wishes to verify that a user is above the age of 18, it is not necessary to leak the birthday and year, even though it might have been asserted and signed in a monolithic manner, but instead the IdP can now simply assert that the user is above 18 based on the info it has. This of course applies to other attributes as well, for example location. Instead of sharing the exact address of a user, it might be enough to simply share which city, or even country, the user is residing in. This is a great way to ensure *granular and selective disclosure.*

Problems. Having all attributes at an IdP is not only an advantage, but also a liability. In case the IdP does get compromised, all the private information of

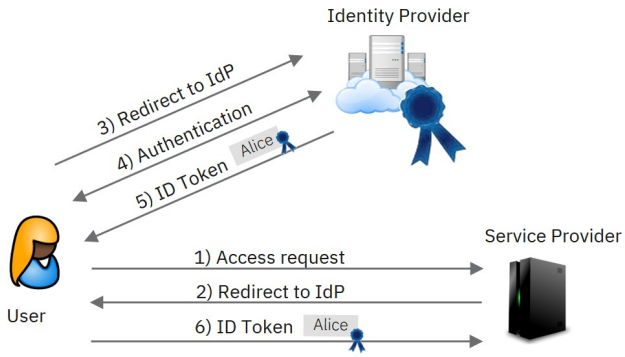

Fig. 2. Single Sign-On Process in Federated Identity Management

the users is leaked. Like for the X.509 solution, the problem of linkability still remains. The user's account name is supplied to every service provider. Thus, any two service providers that compare logs will be able to identify if the same user has been using both services. Besides linkability, we now also have the issue of *traceability*, meaning that we now have a single party, the IdP, which will learn about *every* service provider that the user wishes to access.

SAML. There are different possibilities for realizing a federated identity solution. One of the oldest standardized approaches to this is SAML [29]. This approach is mainly focused on the enterprise setting rather than catering for end-users on the Internet. SAML has been specified by the Internet Engineering Task Force (IETF) in a Request For Comments (RFC) and also in an OASIS standard. The first version was standardized in 2002 and the second, major, revision was standardized in 2005. Messages in SAML are XML and communication between the different parties take part over HTTP, using the SOAP protocol.

The focus of this standard is not really efficiency, as can be seen from the high reliance on XML. That said, the protocol is relatively straightforward and specifies the token issued by the IdP to be signed. This means that the service provider must hold the public certificate of the IdP and checks the validity of the token by verifying the signature of the IdP, along with its timestamp. In particular this means that anyone holding the token will be able to impersonate the user towards the specific service provider. Hence it is crucial that such tokens have a sufficiently time-constrained validity.

OAuth. OAuth is a new standard used to achieve a federated identity solution. The standard leaves open a lot of aspects for deployment and is thus not as straightforward to implement without first making certain decision about which of the plethora of options to implement and support. The main spec of OAuth stems from 2012 [20] and is also specified by the IETF through an RFC. Unlike SAML, messages are sent in OAuth are based on JSON and is working directly on top of TLS.

The specification allows different flows and types of tokens. In one flow, which is for example used with OpenID Connect, bearer tokens are used, such that they are time-constrained and authenticated based on a signature of the IdP (similar to SAML). However, the OAuth specification also allows for verification of token validity through the service provider contacting the IdP. The OpenID Connect flow of OAuth is of particular interest as this has been implemented by several social federated identity providers such as Facebook, Google and Twitter.

3 Privacy-Enhancing Attribute-Based Credentials

In the previous section we have shown the benefits and drawbacks of deployed technologies such as X.509 certificates and federated identity solutions. A shared drawback is their little to no guarantees when it comes to the users' *privacy*: they either expose more user information than necessary or require a trusted party that can track all the users' online behaviour.

In response to this problem, Privacy-Enhancing Attribute-Based Credentials (P-ABCs) [3, 9] have been proposed as a possible solution. P-ABCs follow the same "offline" approach as X.509 certificates, i.e., users receive a *credential* from a trusted issuer. The credential contains a set of certified user attributes that the issuer vouches for, e.g., the user's name, age or address, and that can be used to convince a service provider of the validity of such claimed attributes. Unlike classical X.509 certificates, which have to be disclosed entirely and expose a static value whenever used, these P-ABC credentials allow the user to derive dedicated one-time use tokens that reveal only the information that is minimally necessary.

P-ABCs in a Nutshell. More precisely, when the user wants to access an account or resources at a service provider (SP), the provider first responds with a *presentation policy*, stating the requirements the user has to fulfill. These requirements can range from simple re-authentication requests w.r.t. a certain account, to requests for proofs of certain attributes, e.g., that the user is older than 18 years and lives in a European country. If the user's credential can satisfy the policy, it derives a so-called *presentation token* from the credential. The presentation token contains only the minimal amount of attributes requested by the policy. P-ABCs thereby support not only minimal disclosure techniques, but also predicate proofs. That is, instead of revealing the full date of birth for proving that she is over 18, the user can compute a dedicated proof revealing nothing beyond that fact. Importantly, the user can derive only presentation tokens that are consistent with the information certified in the credential, and the service provider can verify the token against the policy and be convinced about its correctness.

A crucial concept in P-ABCs is that of *user-controlled linkability*. Per default, presentation tokens are unlinkable, which means that a receiver cannot tell whether two presentation tokens are stemming from the same or two different users (unless this is revealed by the disclosed attributes). As full unlinkability would render P-ABCs useless in many applications, the user can also choose to create tokens w.r.t. a certain *pseudonym*. The pseudonym can be seen as a

privacy-enhancing version of standard public keys: they are derived from a user secret key that is implicitly embedded in the credential, but the user can derive arbitrarily many and unlinkable pseudonyms from the same secret key. The user can choose to re-use an established pseudonym, which makes all tokens released for the same pseudonym linkable, but unlinkability across different pseudonyms is still guaranteed.

P-ABCs also support a number of additional features and concepts, such as privacy-friendly revocation or conditional disclosure and inspection. We refer to [3] for a detailed overview of these concepts and possible realizations.

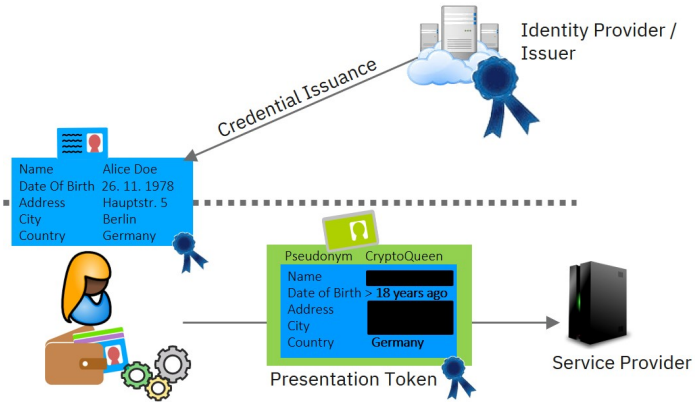

Fig. 3. Privacy-Preserving Authentication via P-ABCs

Advantages. The clear advantage of P-ABCs is their user centric and privacy-preserving behaviour. As P-ABCs follow the offline approach, the Identity Provider is only involved once, at credential issuance, but does not need to be contacted at every authentication request. This immediately solves the privacy problem posed by SSO solutions, but at the same time offers the same (or even better) minimal disclosure capabilities as an online IdP. The user has full control over the information she discloses and can authenticate in an unlinkable yet certified manner. The latter refers to the use of pseudonyms, which allow users to generate certified partial identities when desired, which can be backed-up by authenticated attributes. Thus, in terms of security and privacy, P-ABCs are clearly superior to conventional certificates and SSO-based solutions.

Problems. Despite being around for almost two decades now, and being available through mature realizations such as IBM's Identity Mixer [6, 7] or Microsoft's U-Prove [28], P-ABCs have not seen noticeable adoption yet. The main reason is their bad usability. P-ABCs inherit the same struggle that conventional certificate-based solutions such as X.509 have, as it requires users to securely manage their credentials and key material. Any compromise of these credentials and keys will allow the attacker to impersonate the user, hence they must enjoy strong protection, e.g., be kept on trusted hardware tokens, such as smart cards.

While being sufficiently secure, this is not a user-friendly approach as users need to have appropriate smart-card readers and apps for all their devices.

Furthermore, the handling and verification of P-ABCs tokens is more complex than of conventional signatures and credentials. The multitude of features and privacy-enhancing capabilities require an expressive policy language, credential matching and ontologies to verify whether the revealed information satisfies the required policy. The realizations of P-ABCs also require a careful combination of advanced cryptographic techniques, such as zero-knowledge proofs, which is not available through simple API calls of widespread cryptographic libraries. Instead, users and service provides must use tailored software packages in order to parse, create or verify such P-ABCs.

Another and more subtle problem of existing P-ABC scheme is their reliance on a single Identity Provider that issues the attribute-based credentials. While, for most schemes, the issuer does not have to be trusted for the privacy-related properties to hold, it still is the main entity and root of trust for the provided identity assertions. Thus, the issuer also presents a single point of failure, although is by far not as critical as in SSO solutions.

4 Single Sign-On with Distributed Trust

The Olympus project is aiming to combine the best of the discussed approaches to create a more secure approach to online identity management, while avoiding the drawbacks of the current solutions. Roughly, it follows the Single Sign-On identity management approach, and uses tools from distributed and privacy-enhancing cryptography to obliterate a single point of failure. The resulting system's goals are as follows.

- Feature strongly secure and privacy-preserving authentication (e.g., minimal disclosure of attributes).
- No single point of failure or fully trusted third party.
- No secure hardware and credential management on the side of the user.
- Simple integration for service providers (integration into simple authentication standards such as, e.g., SAML and OAuth)

Let us first recap why existing solutions described in the preceding sections fail to achieve all these goals simultaneously.

X.509 certificates require the user to manage her secret keys and use a trusted device. The method is also not integratable into authentication standards such as SAML or OAUTH. While all this is not an issue for the federated identity approach, here the problem lies in the single point of failure that the fully trusted IdP constitutes. The Privacy-ABCs described in the previous section enable strong user privacy and minimal attribute disclosure, but require heavy machinery on the side of the user as well as the SP when verifying the credential – and there is no hope to make these systems work with simple authentication token standards. As we see, all systems have their advantages and excel in fulfilling some of the above listed goals – but none of them is able to fulfill all of them simultaneously.

4.1 The OLYMPUS Approach

Within the Olympus project, an SSO scheme is designed that fulfills all of the above mentioned goals. The main idea is to modify the federated identity solution described in Section 2 and remove the single point of failure from it. To remind the reader, this single point of failure is the Identity Provider (IdP) which is in charge of storing the user's attributes, verifying its identity and issuing access tokens for the authenticated user. Within Olympus, advanced cryptographic protocols are used to *distribute* the task of the single IdP to a set of n *different* IdPs. None of these IdPs needs to be fully trusted. The system's privacy and security guarantees hold as long as *not all of the IdPs are corrupted*. The set of all n IdPs is called a *virtual IdP* (vIdP).

Essentially, the vIdP needs to perform all activities jointly that the former IdP of the federated identity scheme took care of. This boils down to two main tasks: (1) Verifying the password of the user and (2) issuing an access token. Within Olympus, cryptographic solutions for distributing these two tasks among a set of servers are selected and carefully combined. We will give more details on the cryptographic part next.

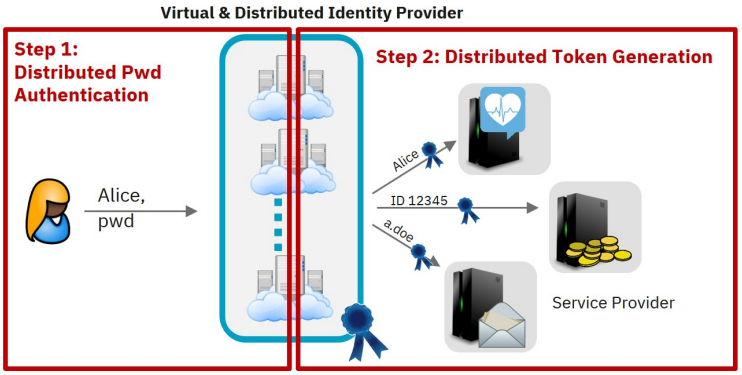

Fig. 4. The Olympus privacy-preserving SSO system: the core concept is the virtual Identity Provider that consists of several servers of which none has to be fully trusted. The single servers need to jointly perform tasks such as password verification and token generation of the single IdP in a federated identity system.

Cryptography Part I: Distributed Password Verification. Password-based authentication is the most prominent form of user authentication towards an IdP. Normally, the IdP stores the password in form of, e.g., a hash $H(pw)$ – this is often called a *password file*. If the password is chosen only from a small dictionary, a malicious or compromised IdP can easily find out the user's password and potentially impersonate the user towards other IdPs. Server compromise, where attackers can steal billions of user passwords, is nowadays the main threat to password security.

Remembering long passwords and moreover typing them error-free into mobile devices puts a lot of burden on the user. However, password size can be dramatically reduced if (a) servers throttle login attempts and (b) the password file is protected from getting leaked through a server compromise attack. Within the OLYMPUS project, a system is designed where the role of the single and fully trusted IdP is distributed among n different servers. Intuitively, one can think of sharing the password file $H(pw)$ among all n IdPs, thus keeping an attacker from offline attacking pw as long as not all n of them are compromised. Realizations for such distributed password-verification protocols have already been proposed in [1, 12, 8] and make use of a *(distributed) oblivious pseudorandom function.*

Let us look at this primitive in a bit more detail, assuming the reader is already familiar with the concept of a pseudorandom function (PRF). An **oblivious** PRF (OPRF) is simply a PRF interpreted as an interactive 2-party protocol, where a user provides the input x to the function, the server contributes with a PRF key K and the user obtains $y \leftarrow \mathsf{PRF}_K(x)$. The server learns neither x nor y, and the user does not learn anything about K besides what he can derive from y. Oblivious PRFs have been introduced in [16] and have since then been extensively used in password-based protocols. Such OPRFs already allow the server to generate password files of the form $\mathsf{PRF}_K(pw)$ without the user ever having to reveal the password towards the server.

To protect against a single server running offline attacks by continuously evaluating the PRF and comparing it against $\mathsf{PRF}_K(pw)$, a **distributed** OPRF [23] (also implicitly used in [8]) can be used. Here, the PRF key K is *shared* among many servers, who then need to participate in the interactive evaluation protocol, contributing with their key share. As long as not all of them are compromised, offline attacks remain infeasible. Further, as every evaluation of the OPRF requires participation of all n servers, the remaining honest servers can refuse or pause the evaluation when they detect suspicious behaviour such as unusually high amounts of evaluation requests, which might indicate online guessing attacks.

Overall, the oblivious aspect of OPRFs ensures that the servers can generate password-derived information without learning anything about the underlying password; and the distributed realization guarantees that both offline- and online attacks against the password-derived values are infeasible as long as not all servers are corrupted.

Cryptography Part II: Distributed Token Generation. Generating the access token in the OLYMPUS system is done by a set of n IdPs instead of a single trusted IdP such as in a federated identity scheme. Token generation, from a cryptographic viewpoint, is essentially digital signing done by the IdP. Thus, for the OLYMPUS system a *distributed* digital signature scheme (DSIG) is required. Such a scheme allows sharing the signing key among the n IdPs in a way such that only if all of the IdPs participate in the distributed signing protocol the user will obtain a valid signature on his access request.

Distributed signature schemes can be obtained from the RSA assumption [4, 13, 2, 15]. The main benefit is that verification of distributed RSA signatures

does not introduce an overhead over verification in the non-distributed case. It is also possible to construct a distributed signature scheme based on (EC)DSA [25, 11], where distributed key generation is more straightforward, but distributed signing less natural than with RSA-based signatures. Regardless of whether the underlying cryptographic assumption is RSA or a discrete-log type assumption, any DSIG scheme should guarantee *unforgeability* of signatures, even though the attacker might hold up to all but one signing key share. This property yields a system where access token cannot be forged unless all servers are corrupted.

Recently, also distributed versions for P-ABCs, or rather their main signature building block, have been proposed. More precisely, Sonnino et al. [30] and Gennaro et al. [18] have shown how distributed issuance can be realized for pairing-based credentials such as CL- or BBS-schemes. Despite the distributed issuance, the format of the resulting credential is preserved, which means that the user's derivation of presentation tokens is not impacted by this change. We sketch below how distributed P-ABCs could be used to instantiate the distributed token generation in a more privacy-preserving manner.

4.2 Open Questions

While the virtual IdP at the core of the Olympus system resolves many security concerns, the system does not satisfy all our goals yet. The most crucial drawback is that each IdP can track the online behavior of the user, even amplifying the privacy problem of standard SSO. We will therefore investigate how the distributed system needs to be extended to prevent such tracking and ensure unlinkability of the users' requests.

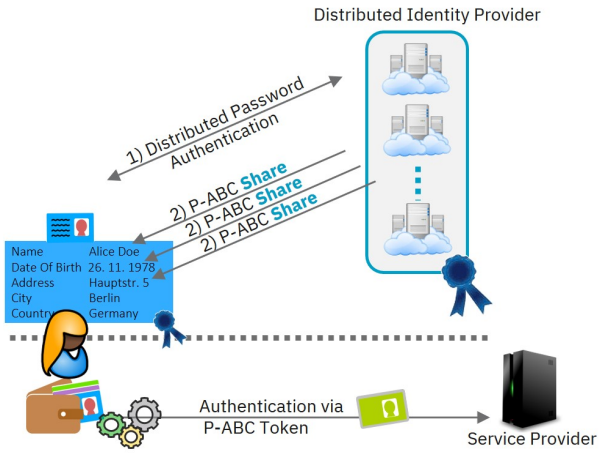

Fig. 5. A P-ABC based Olympus system

P-ABCs & Olympus? A straightforward approach to achieve the aforementioned privacy guarantees is to integrate P-ABCs into our Olympus solution

and replace the standard distributed signatures (as described above) with a distributed issuance protocol of a P-ABC credential (see Figure 5). That is, the user no longer receives signature shares of the final SSO token from the IdPs, but rather shares of a credential containing all her user attributes. We then leverage the power of P-ABCs to let the user derive the final SSO token as a P-ABC presentation token from the freshly received credential, inheriting the unlinkability and minimal disclosure features from P-ABCs.

A crucial design goal of Olympus is to avoid any trust assumptions for the clients' devices. Thus, we must ensure that a corruption of the user's device does not result in a permanent compromise of all her accounts. This can be realized by making the user credential *short-lived* only. More precisely, an additional *epoch* attribute gets introduced into every credential that must always be revealed in a presentation token and the service provider checks the epoch's validity.

However, avoiding the need for trusted hardware on the user side is pretty much the only advantage compared to standalone P-ABCs: users and service providers still need the full-blown P-ABC stack to generate and verify the presentation tokens. Thus, we will also investigate more lightweight alternatives to this approach.

Proactive Security. Last but not least, finding an adequate threat model is not straightforward for a distributed system such as Olympus. While it is clear that not all servers can be under full control of the adversary, the system could be strengthened by allowing the adversary to *transiently* compromise all servers, as long as not all of them are compromised at the same time. For this, the Olympus system would have to specify cryptographic primitives that allow *recovering* from adversarial corruption.

4.3 Discussions at the Workshop

One question raised at the workshop targeted the P-ABC-based Olympus system sketched above. While guaranteeing strong privacy properties, this solution would inherit almost all disadvantages of a standalone P-ABC system. In particular, service providers need to run dedicated P-ABC libraries to parse and verify the complex presentation tokens, which contradicts the initial intention of seamless integration of SSO. It was discussed whether the verification of such tokens can be outsourced to an oblivious, and possibly distributed, party as well. Intuitively, outsourcing the entire process will be hard to realize, as a considerable part of the verification deals with parsing (xml-based) policies and matching them against the statements that are actually proven in the presentation token. Thus, this part does not seem very amendable to a blinded operation. The verification of the cryptographic evidence might be a more promising target, possibly using techniques from commuting signatures which maintain their verifiability even when being encrypted [17].

Another question was whether usage of *digital online wallets* could be helpful with online identity management. A digital wallet is essentially a storage for cryptographic objects such as digital coins, secret keys or passwords. Surely,

using a cryptographic key from a wallet to authenticate to an IdP provides better security against a compromised IdP than authenticating with a password. Instead, the user protects access to her wallet with a password. Nonetheless, the wallet has to be stored on all devices the user wants to use a service from, or alternatively at a cloud or wallet provider. Essentially, using this approach, we mainly introduce more points of failure, since we now also have to trust the entity hosting the wallet, e.g., the wallet provider. Workshop participants further discussed whether a *distributed wallet solution* could remedy the situation. And this seems indeed the case, since then trust is distributed among a set of providers. Such a distributed wallet would likely require similar techniques as used within the OLYMPUS project.

References

1. Agrawal, S., Miao, P., Mohassel, P., Mukherjee, P.: PASTA: password-based threshold authentication. In: ACM CCS. pp. 2042–2059 (2018)
2. Algesheimer, J., Camenisch, J., Shoup, V.: Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In: Advances in Cryptology - CRYPTO. pp. 417–432 (2002)
3. Bichsel, P., Camenisch, J., Dubovitskaya, M., Enderlein, R.R., Krenn, S., Krontiris, I., Lehmann, A., Neven, G., Paquin, C., Preiss, F., Rannenberg, K., Sabouri, A.: An architecture for privacy-abcs. In: Attribute-based Credentials for Trust: Identity in the Information Society, pp. 11–78. Springer (2015)
4. Boneh, D., Franklin, M.K.: Efficient generation of shared RSA keys. J. ACM **48**(4), 702–722 (2001)
5. Cadwalladr, C., Graham-Harrison, E.: Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach. The Guardian `https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election`
6. Camenisch, J., Herreweghen, E.V.: Design and implementation of the *idemix* anonymous credential system. In: ACM CCS. pp. 21–30 (2002)
7. Camenisch, J., Krenn, S., Lehmann, A., Mikkelsen, G.L., Neven, G., Pedersen, M.Ø.: Formal treatment of privacy-enhancing credential systems. IACR Cryptology ePrint Archive **2014**, 708 (2014)
8. Camenisch, J., Lehmann, A., Neven, G.: Optimal distributed password verification. In: ACM CCS. pp. 182–194 (2015)
9. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Advances in Cryptology - EUROCRYPT. pp. 93–118 (2001)
10. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
11. Doerner, J., Kondi, Y., Lee, E., Shelat, A.: Threshold ECDSA from ECDSA assumptions: The multiparty case. In: IEEE Symposium on Security and Privacy, SP. pp. 1051–1066 (2019)
12. Everspaugh, A., Chatterjee, R., Scott, S., Juels, A., Ristenpart, T.: The pythia PRF service. In: 24th USENIX Security Symposium. pp. 547–562 (2015)
13. Frankel, Y., MacKenzie, P.D., Yung, M.: Robust efficient distributed rsa-key generation. In: Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing. pp. 663–672 (1998)

14. Frauenfelder, M.: 'I forgot my PIN': An epic tale of losing $30,000 in bitcoin. Wired https://www.wired.com/story/i-forgot-my-pin-an-epic-tale-of-losing-dollar30000-in-bitcoin/
15. Frederiksen, T.K., Lindell, Y., Osheter, V., Pinkas, B.: Fast distributed RSA key generation for semi-honest and malicious adversaries. In: Advances in Cryptology - CRYPTO. pp. 331–361 (2018)
16. Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O.: Keyword search and oblivious pseudorandom functions. In: Theory of Cryptography TCC. pp. 303–324 (2005)
17. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Advances in Cryptology – EUROCRYPT (2011)
18. Gennaro, R., Goldfeder, S., Ithurburn, B.: Fully distributed group signatures (2019), https://www.orbs.com/wp-content/uploads/2019/04/Crypto_Group_signatures-2.pdf
19. Goel, V., Perlroth, N.: Yahoo says 1 billion user accounts were hacked. The New York Times https://www.nytimes.com/2016/12/14/technology/yahoo-hack.html
20. Hardt, D.: The OAuth 2.0 Authorization Framework. RFC 6749 (October 2012), https://tools.ietf.org/html/rfc6749
21. Honan, M.: How apple and amazon security flaws led to my epic hacking. Wired https://www.wired.com/2012/08/apple-amazon-mat-honan-hacking/
22. Hong, N., Hoffman, L., Andriotis, A.: Capital one reports data breach affecting 100 million customers, applicants. The Wall Street Journal https://www.wsj.com/articles/capital-one-reports-data-breach-11564443355
23. Jarecki, S., Kiayias, A., Krawczyk, H.: Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In: Advances in Cryptology - ASIACRYPT. pp. 233–253 (2014)
24. Kosinski, M., Matz, S., Gosling, S., Popov, V., Stillwell, D.: Facebook as a research tool for the social sciences. The American psychologist **70**, 543–556 (09 2015). https://doi.org/10.1037/a0039210
25. Lindell, Y., Nof, A.: Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody. In: ACM CCS. pp. 1837–1854 (2018)
26. Newman, L.H.: Equifax officially has no excuse. Wired https://www.wired.com/story/equifax-breach-no-excuse/
27. O'Flaherty, K.: Collection 1 breach – how to find out if your password has been stolen. Forbes (2019), https://www.forbes.com/sites/kateoflahertyuk/2019/01/17/collection-1-breach-how-to-find-out-if-your-password-has-been-stolen/#37aa36382a2e
28. Paquin, C., Zaverucha, G.: U-prove cryptographic specification v1. 1. Technical Report, Microsoft Corporation (2011)
29. Rekhter, Y., Li, T.: Security Assertion Markup Language (SAML) V2.0. OASIS (March 2015), http://saml.xml.org/saml-specifications
30. Sonnino, A., Al-Bassam, M., Bano, S., Meiklejohn, S., Danezis, G.: Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. In: Network and Distributed System Security Symposium, NDSS (2019)
31. Wang, C., Jan, S.T.K., Hu, H., Bossart, D., Wang, G.: The next domino to fall: Empirical analysis of user passwords across online services. In: Proc. of CODASPY (2018)