emmy – Trust-Enhancing Authentication Library

Miha Stopar[1], Manca Bizjak[1], Jolanda Modic[1], Jan Hartman[1], Anže Žitnik[1], and Tilen Marc[1,2]

[1] XLAB d.o.o., Ljubljana, Slovenia
{name.surname}@xlab.si
[2] Institute of Mathematics, Physics and Mechanics, Ljubljana, Slovenia

Abstract. People, organizations, devices need to make many kinds of claims as part of their everyday activities. Digital credentials can enable to transmit instantly verifiable claims about their name, date of birth, gender, location, accomplishments. Some privacy-enhancing digital credentials enable revealing only part of your identity and thus hiding all information that is not necessarily needed for the online service. In the past two decades, several privacy- and trust-enhancing authentication techniques and approaches have been proposed to implement such verifiable digital credentials, mostly on the theoretical level. Some implementations exist, but either lack functionalities, rely on heavy computational machinery or are not available in open source. This paper presents emmy, a fully-fledged open source cryptographic library for secure, privacy-aware, and trust-enhancing authentication towards online services.

Keywords: Trust · Privacy · Zero-Knowledge Proofs · Identity Management · Anonymity · Cloud Services.

1 Introduction

Service providers progressively form their strategies and base their business decisions on the data they can collect through their everyday operations. They have more data at their disposal than ever before, thanks to the increasing use of highly evolved ICT systems. While the opportunities to benefit from data collection rise, the data protection requirements are becoming more and more strict. With the GDPR and increasingly more privacy-aware individuals, organisations are seeking a compromise that will enable them to collect and analyse their users' data to innovate, optimize, and grow their businesses, while at the same time comply with data protection regulations and keep trust and confidence of their users. Trust in online services can really only be achieved if their users are given full control over their privacy, digital identities, and personal data. Privacy-concerned individuals and trust-seeking organisations thus require novel technologies that enable user-controlled privacy and, where possible, full anonymity.

Credentials which consist of separate sets of individual claims which can be selectively revealed to the service provider and can be verified without contacting a centralized trust source are called verifiable credentials. There is a W3C

Verifiable Claims Working Group [50] which aims to provide standards for expressing and exchanging the verifiable credentials. Verifiable credentials provide users with a fine-grained control about which parts of their personal information they want to reveal. For some online services revealing date of birth might suffice. Or users might need to reveal date of birth, gender, and nationality, but not name and address. In extreme cases, the user might only reveal the entitlement to the service and no claims.

By using verifiable credentials users get more control over their personal data and service providers get an assurance of the authenticity and accuracy of the data. It is well-known [35] that due to privacy concerns and unwanted marketing the users are often giving false information when registering online. Even if only a small percentage of database entries is corrupted, the accuracy of analyses can heavily decline.

In this paper, we present a cryptographic library *emmy* that encapsulates primitives and protocols used in *anonymous attribute-based credentials* (AABCs) which are the underlying cryptographic primitives for the verifiable credentials. We present its implementation and an example of a complete cloud system that uses AABCs for a privacy- and trust-enhancing service. Note that emmy is not the first implementation of an anonymous authentication scheme. It will soon be two decades since the first one has been designed and implemented, however, this emmy's predecessor never really made it to any real-world application as it is based on (too) heavy computational machinery and lacks functionalities. The goal of this paper is to show how our library can be easily integrated into cloud services to provide controlled privacy to finally support and facilitate the development of privacy-aware and trust-enhancing cloud services.

The library presented in this paper was partially implemented in European H2020 research projects FENTEC [20] (grant number 780108) and mF2C [36] (grant number 730929). The code with guidelines is available online [19].

Contributions. This paper addresses the lack of privacy-friendly and trust-enabling technologies by the following contributions:

1. *Overview of privacy-enhancing and trust-enabling technologies.* We provide an overview of currently known privacy- and anonymity-enabling cryptographic approaches and implementations in Sections 2 and 3.
2. *Implementation of a library for AABCs.* In Section 4 we present a cryptographic library that enables trust-enhanced and anonymous authentication to cloud services. The library differs from others by incorporating an efficient communication layer which is crucial in systems that rely on complex interactions between clients and servers.
3. *Demonstration of a secure, privacy-aware, and trust-enhancing certificate authority system.* In Section 5 we present the emmy API and demonstrate how it can be integrated into a self-sovereign identity system where users create, control, own, and maintain identities as data stored on a smartphone or computer.

2 Related Theoretical Work

The fundamental primitive for the AABCs is a so-called *Zero-Knowledge Proof* (ZKP). This is an interactive, two-party protocol between a *prover* and a *verifier*, where the prover claims to know something and needs to convince the verifier about this fact in a private manner. Namely, the prover will prove to the verifier that some statement is true, and the verifier will be fully convinced that the statement is true, but will not learn anything about the statement as a result of this process (the verifier will obtain zero knowledge).

Any proof system must be complete and sound. *Completeness* means that an honest prover can always convince an honest verifier of a true statement, and *soundness* means that a dishonest prover can almost never succeed in convincing an honest verifier of a false statement (there is a negligibly small probability of success). A ZKP must be complete, sound, and, additionally, it must satisfy the *zero-knowledge* property, which means that the proof does not reveal any information about the secret to the verifier, except for what is already revealed by the claim itself.

A typical ZKP is a proof of knowledge of a discrete logarithm. That is, given a publicly known number t, the prover wants to prove knowledge of a number x such that $g^x = t \,(mod\, p)$ where p is some prime number. This example of a ZKP was first proposed by Schnorr [43] in 1991. Schnorr-like protocols later became the basis for AABCs, introduced by Camenisch and Lysyanskaya in 2001 [11]. AABCs allow an identity provider (a trusted third party) to issue a credential to a user. This credential contains various claims, which describe different properties of the user, such as gender, age, address or date of birth, and also user's rights or roles, for example, access rights. Using the credential, the user can prove to some service provider the possession of a credential containing a given claim without revealing any other information stored in the credential.

A number of variants of the Camenisch-Lysyanskaya scheme have been proposed offering additional functionalities such as credential revocation [2,10,12,37] and improvements such as efficiently encoding binary attributes [9] or verifiably encrypting attributes under some third party's encryption key [7,13].

In 2012, a novel form of ZKPs was proposed [1], named *Zero-Knowledge Succinct Non-Interactive Argument of Knowledge* (zk-SNARK). A zk-SNARK is a non-interactive and succinct protocol where only one single message is sent from the prover to the verifier. It offers a very small proof that can be quickly verified even if a statement is very large. Contrary to the zk-SNARK protocol, a recently proposed new ZKP construct called *Bulletproofs* [5] requires no trusted setup. However, verifying a bulletproof is more time consuming than verifying a zk-SNARK proof.

Due to their efficiency, Bulletproofs and zk-SNARKs are highly suitable for blockchain-based applications and, in particular, cryptocurrencies. On the other hand, AABCs are specifically tailored for identity management and authentication processes. Emmy offers AABC primitives and currently does not contain any zk-SNARK or Bulletproof schemes. This does not mean it cannot be integrated with blockchains, the standards and recommendations of W3C Verifiable Claims

Working Group [50] which is heavily focused on the scenarios with distributed ledgers fully rely on the AABCs.

The widely used techniques for transferring user attributes, like SAML [44], OpenID [38], and WS-Federation [51], present considerable privacy concerns. Namely, identity providers can track activities of their users or can even impersonate them. In contrast, with AABCs, issuers are not involved in the authentication process. Additionally, users disclose only those attributes that are required by services in a way that makes linking the transactions highly difficult.

3 Existing Implementations

While a significant amount of theory on Privacy-Enhancing Technologies (PETs) exist, there is a clear lack of implementations. The library most frequently used for demonstrating research innovations related to AABCs is Identity Mixer (idemix) [26] from IBM, which has also been integrated into the IBM cloud platform BlueMIX [25]. An older version of idemix is publicly available but lacks some core functionalities like an integrated communication channel for prover-verifier interactions. Moreover, this open-source idemix repository is no longer maintained.

A newer version of idemix is available [27] as part of the the Hyperledger Fabric [24], a platform for distributed ledger solutions, but this version provides only a limited functionality and does not contain the ZKP machinery needed for fully-working AABCs (e.g., range proofs, proofs of partial knowledge). Besides Fabric, the Hyperledger consortium provides tools, libraries, and reusable software components for providing digital identities rooted on distributed ledgers so that they are interoperable across different administrative domains and applications. Hyperledger Indy is available as open-source [29] and also contains an implementation of the Camenisch-Lysyanskaya anonymous authentication scheme [11]. However, yet again, no machinery for ZKPs is provided. Microsoft released their version of a AABC system called U-Prove [49]. This library is complex and hardly usable for non-expert developers. It lacks functionalities such as range proofs and has not been maintained since 2014. U-Prove and idemix have been part of the ABC4Trust project [42] which addressed the federation and interchangeability of technologies that support ABCs.

IRMA [45] is another platform for privacy-friendly authentication, also based on the famous Camenisch-Lysyanskaya scheme [11]. IRMA provides attribute-based credentials integrated into the Android application and is thus not offered as a library which could be easily reused.

Recently, a group of researchers presented their implementation of an AABC system called CLARC [32]. The library implements a series of ZKP schemes but lacks a communication layer.

In 2003, the Trusted Computing Group [48] adopted and standardized the *Direct Anonymous Attestation* (DAA) scheme [4] as the method for remote authentication of a hardware module, called Trusted Platform Module (TPM), while preserving privacy of the user of the platform that contains the module.

Over the years, DAA implementations improved [8,6], but they are still tightly bound to the TPM chip, which makes these constructions often too prohibitive for the use with low-resource devices (e.g., with smartcards). Additionally, the implementation is not offered as a library.

The Intel Enhanced Privacy ID (Intel EPID) technology [30] is being used in various applications that need a guarantee that involved devices are authentic, have not been hijacked or replicated into a non-genuine piece of hardware. The source code for Intel EPID is publicly available [46], but the library does not offer a modular architecture that could be used for combining cryptographic primitives into schemes other than Intel EPID. Also, the communication layer is not included.

As discussed above, several implementations of PETs that support the development of anonymous authentication solutions exist, however, there is no fully-fledged AABC library that could be easily integrated into cloud services to offer privacy-aware and trust-enhancing authentication. In the remainder of the paper, we address this problem by presenting our library that integrates the cryptographic primitives and protocols, the entire ZKP machinery, along with a robust communication layer required to enable anonymous authentication to cloud services.

4 Emmy Building Blocks

Emmy [19] is a cryptographic library written in the Go language. It offers various schemes enabling the development of privacy-aware and trust-enhancing authentication systems such as AABCs. It is a fully-fledged software library that contains low-level cryptographic primitives and protocols as well as high-level communication procedures to facilitate simple integration with existing cloud services. Emmy implements the following layers:

- **Utilities layer** provides various randomness related functions, like concurrent generation of *safe primes* (primes of the form $2p + 1$, where p is also a prime). It provides mathematical functions, for example, for the decomposition of positive integers into a sum of squares (Lipmaa decomposition [33]), which is needed for the implementation of range ZKPs (to prove, in zero-knowledge, that some secret value is an element of an interval). Furthermore, to turn interactive ZKPs into non-interactive ZKPs (three-move protocol into one-move protocol), the layer implements Fiat-Shamir heuristics [21].
- **Groups layer** provides various modular arithmetic and elliptic curve groups. A common API is provided to access functions for multiplying the elements, computing an inverse, retrieving the random element, and generating the group parameters for a given security parameter (for modular arithmetic groups). Four elliptic curves are offered (NIST recommended curves [18] over prime fields: P-192, P-224, P-384, P-521) as wrappers around Go implementation of elliptic curves. Besides the basic modular arithmetic group of all integers smaller than n and coprime with n, denoted as $\mathbb{Z}_n^*$ (if n is a prime, we use notation $\mathbb{Z}_p^*$), emmy implements the *RSA group*, which is

a group $\mathbb{Z}_n^*$, where n is a product of two distinct large primes. RSA group where only square elements are considered is named *QR RSA group*. Where n is a product of two safe primes, a QR RSA group $\mathbb{Z}_n^*$ is called a *QR special RSA group*. The final modular arithmetic group implemented by emmy is the *Schnorr group*, which is a cyclic subgroup of $\mathbb{Z}_p^*$, such that for its order q and some r it holds $p = qr+1$ (p, q are primes). The order of a Schnorr group is smaller than the order of a group $\mathbb{Z}_p^*$, which means faster computations.

- **Commitments layer** provides several commitment schemes. Commitments enable one party of a protocol to choose some value and commit to it while keeping it hidden, with the ability to reveal the committed value later. Commitments are important to a variety of cryptographic protocols, including ZKPs. Emmy implements several commitment schemes. Pedersen commitment [40] is to be used in the *Schnorr group*, Damgard-Fujisaki [17] commitment in the *QR special RSA group*, and Q-One-Way-based commitment [15] in the *RSA group*.
- **ZKP layer** provides the core building blocks for anonymous authentication schemes. For historical reasons proofs for quadratic residuosity and nonresiduosity are implemented which were the first known ZKPs presented in a seminal paper [22]. For proving statements about discrete logarithms, emmy offers several different schemes. For proving knowledge of a discrete logarithm modulo prime, emmy implements Schnorr proof [43]. For proving a knowledge of equality of two discrete logarithms emmy offers proof introduced by Chaum and Pedersen [14]. A non-interactive version of proof of equality of discrete logarithms is also implemented by emmy, following the construction proposed by Lysyanskaya et al. [34]. For proving a partial knowledge of a discrete algorithm, emmy follows construction from [16]. Schnorr proof systems can be generalized to all one-way homomorphisms (one-way meaning that for a homomorphism f, one can easily compute $y = f(x)$ whereas computation of its preimage $x = f^{-1}(y)$ is practically infeasible; in Schnorr proof a homomorphism f is given by $f(x) = g^x$) [43]. Within emmy, proofs of homomorphism preimage knowledge and partial proofs of homomorphism preimage knowledge are implemented. Additionally, Schnorr can be generalized for proving the knowledge of multiple values $a_1, a_2, \ldots, a_k$ for a given number h such that $h = g_1^{a_1} \circ g_2^{a_2} \circ \cdots \circ g_k^{a_k}$. These are called representation proofs [3]. Emmy implements such proofs for Schnorr group and RSA groups.
- **Communication and portability layer** for client-server interaction via gRPC (for all messages exchanged between the prover and the verifier). It consists of a client, server, and communication layer supporting the execution of the client-server protocols. Communication between clients and the server is based on Google's Protocol Buffers [41] and gRPC [23]. Protocol buffers are a language- and platform-neutral flexible, efficient, and automated mechanism for serializing structured data. The developer needs to define the preferred structure of the data once, and then special generated source code is used to easily write and read the structured data to and from a variety of data streams, using different programming languages. In a client-

server model, Remote Procedure Call (RPC) is a protocol that one program (client) can use to request a service from another program (server) located in a remote computer without having to understand network details. A gRPC is a high-performance, open-source communication system. Although emmy is written in Go, it comes with compatibility package providing client wrappers and types (based on gRPC) that can be used for quickly generating bindings for other languages.

The key ingredient of the Camenisch-Lysyanskaya scheme [11] is a representation proof in *QR special RSA group*. Heavily simplified, the scheme outputs a credential in the form of a triplet of large integers (A, e, v). The user proves to the verifier that he knows the attribute values $a_1, a_2, \ldots, a_k$ such that $A^e = g_1^{a_1} \circ g_2^{a_2} \circ \cdots \circ g_k^{a_k} \circ S^v$. If a user does not want to reveal a_1 (for example name, address or some other attribute which might reveal the user identity), he proves the knowledge of values $a_2, \ldots, a_k$ such that $A^e \circ g_1^{-a_1} = g_2^{a_2} \circ \cdots \circ g_k^{a_k} \circ S^v$

When the credential is issued, for some attribute values only commitments can be revealed to the issuer. The user can later still prove to the verifier that the committed value lies in some specific interval (for example age). This can be done by Damgard-Fujisaki scheme [17] which enables proving various properties of committed values: proofs that you can open a commitment, that two commitments hide the same value, that a commitment contains a multiplication of two committed values, that the committed value is positive, that the committed value is a square, that the committed value lies in some interval range. The latter is based on the Lipmaa scheme [33].

One might ask what is the value of such credentials. Why not simply use attributes in plaintext and signed by an issuer? Due to the randomization of a triplet and the usage of ZKPs, the verifier cannot determine whether any two credential proofs are coming from the same credential (if no unique attributes are revealed). This property is called unlinkability. Even the entity that issued a credential does not have any advantage in this sense.

5 Certificate Authority based on the Photographed Personal Documents

In this section, we demonstrate the integration of the cryptographic library emmy into a real-world scenario where verifiable credentials are issued by Certificate Authority (CA) based on the photographed documents.

Whether the service provider is in fintech, sharing economy or cryptocurrency economy, before an account is opened, the user needs to provide an acceptable proof of identity and proof of address. This is often done by sending photographed personal documents which are then manually verified at the service provider. By having CA which would issue digital credentials based on the photographed documents, the user would need to go through the cumbersome process of photographing and uploading the documents only once. The service providers would need no manual verification as this would be done already by the CA.

The service providers which do not necessarily need the proof of identity can benefit from using verifiable credentials too as they get an assurance of the authenticity and accuracy of data.

The scenario comprises three different entities:

1. CA: a certificate authority where credentials are issued based on the photographed personal documents and new claims (e.g. driving license, education certificates) can be quickly added to the existing credential (again by sending photographed documents).
2. Service provider: any type of service provider which needs proof of identity or wants to get an assurance of the accuracy of users' data.
3. Users of the service provider from point 2.

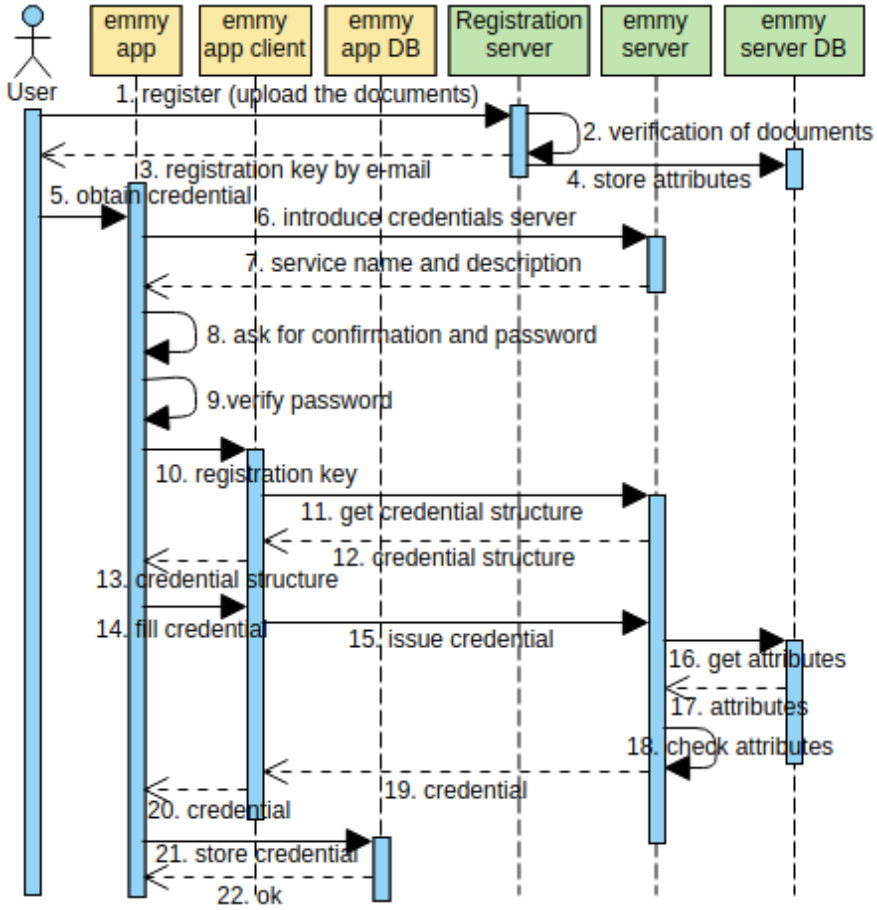

Fig. 1: Obtain credential.

The benefits of using such CA are multifold: service providers get credentials with verified personal data and are thus able to develop more accurate profiling and analysis models, users can selectively reveal data and get discounts from service providers in exchange for providing verified data. CA builds business model based on the fees users pay to obtain credentials.

By using emmy as an underlying library, such a system can be built quickly and efficiently as emmy provides all the required cryptographic primitives as well as the communication layer – all communication between the entities is already built-in.

5.1 Integration of emmy into the Cloud Service

The user first needs to install the emmy app which is a smartphone app or browser extension (if emmy app is provided in a web browser, it has to be shipped as a browser extension which are installed in a secure way [28]). Emmy app offers a graphical user interface to the emmy client. The first time user uses emmy app a password needs to be set up. By using a password-based key derivation function PBKDF2 [39] emmy generates a key which is used to encrypt the secrets (credentials) that are stored internally in emmy app database.

The process to obtain a credential from the CA is depicted in Figure 1. The user goes to the CA website and uploads the photographed documents. An authorized person verifies the documents and triggers registration key generation. The registration key is sent to the user by e-mail. The registration key is then passed to the emmy app and the CA emmy server is contacted to obtain a credential. CA comprises three components: registration server (website), emmy server, and emmy server database.

Obtaining a credential is a multi-step process. First, emmy app establishes gRPC connection and then executes a call to obtain a credential structure (*regKey* is the key returned from the Registration server):

```
client, err := NewCLClient(grpcClientConn)
rc, err := client.GetCredentialStructure(regKey)
```

Emmy server returns the structure of credentials it can issue. The structure can contain known and committed attributes. Respectively, these represent attributes with values known to the issuer and the attributes for which only the committment of a value is known to the issuer. For each attribute the type of the value (type: string / int / bool) is specified and whether the attribute value is known to the issuer or not (known: true / false). Note that in our case only known attributes are used and emmy server could fill the credential with values by itself as it has an access to the database where attributes corresponding to the registration key are stored. However, these values could be then changed by the user and have to be thus validated again when the credential is being issued.

Emmy API offers UpdateValue function to set the attribute values of the credential:

```
name, _ := rc.GetAttr("Name")
err = name.UpdateValue("Jack")
```

Before the client can call IssueCredential method, a user's master secret key needs to be generated and *CredManager* needs to be instantiated.

```
masterSecret := pubKey.GenerateUserMasterSecret()
```

Public key *pubKey* needs to be provided by an issuer. Among other parameters it provides information about the *QR special RSA group* which is used for attributes and *Schnorr group* which is used for a master secret key. Secret key *masterSecret* is encoded into every user's credential as a sharing prevention mechanism. User can only share a credential if he shares a master secret key. Thus, if he shares one credential, all his credentials are shared.

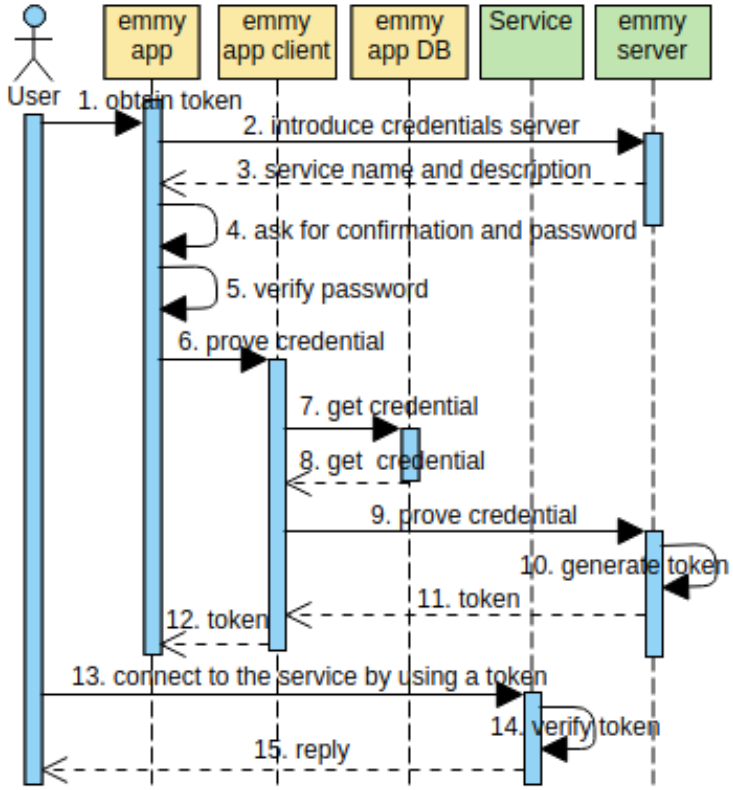

Fig. 2: Connect to the service.

Whenever a session is established with an issuer, the client provides a pseudonym which is a Pedersen commitment [40] to a *masterSecret*.

```
cm, err := cl.NewCredManager(params, pubKey, masterSecret, rc)
```

CredManager manages all ZKP interactions between the emmy client and emmy server. The IssueCredential method can now be triggered:

```
cred, err := client.IssueCredential(cm, regKey)
```

CA (emmy server) retrieves the attributes from the database and checks whether they are the same as attributes sent from the user which are to be encoded in the credential. If everything is ok, the credential is returned to the emmy app. It stores it in the internal database which is encrypted by the key generated from the user's password. The credential comes in the form of a triplet of large numbers (A, e, v). Whenever the possession of a valid credential needs to be proved, properties of the triplet are being proved using ZKPs (see 4). To provide unlinkability, a triplet is randomized by *CredManager* before the possession of a credential is proved.

Verifiable credentials can provide (partial) anonymity: user can reveal only partial information which do not reveal the identity. However, service provider could still observe the IP address and could thus identify the user. To provide a fully anonymity, credentials need to be used together with the privacy tools that reroute Internet traffic via public nodes such as VPN services or TOR [47] software.

Once the service provider has integrated the emmy server and is thus able to verify its credentials, there are different ways to integrate the verification of the credentials into the authentication process. Emmy server might return a session key which is then used for the authentication to the actual service. If unlinkability is required, each time the user connects to the service a new session key needs to be provided as otherwise, the server can easily link the sessions which have the same session key. If unlinkability is not required, the session key can be set to be valid for some period of time. Alternatively, if some claims need to be associated with a session, JSON Web Tokens [31] can be used (see Figure 2). They are self-contained and require only verification of the signature on the service side. Repeatedly using the same token breaks unlinkability but the service provider has an assurance of the authenticity and accuracy of the attributes which is in many cases the property of the verifiable credentials that service providers value the most. Thus, the attributes that are written to the token are verified and can be trusted.

When proving the possession of a valid credential, not all service providers might require all user's data. The user has control over which attributes are to be revealed to the service provider:

```
revealedAttrs = []string{"Gender", "BirthName"}
token, err := client.ProveCredential(cm, cred, revealedAttrs)
```

6 Conclusions and Future Work

Despite the internet reputation for anonymity, it is very difficult to perform any action online with true privacy. On the other hand, verifying identity is cumber-

some and often relies on replicated versions of offline systems which are usually imperfect analogues to showing a physical credential to a human verifier. In this paper, we presented a cryptographic library which provides a complete analogue to reliable offline systems and at the same time provide many advantages due to its digital nature. Most notably, the user can reveal only selected claims to the service provider. In our future work, we plan to open source the emmy smartphone app and the registration server. Additionally, we plan to extend the registration server to support other verification processes, in particular, physical verification, QR code verification, and verification by using anonymous payment using cryptocurrencies.

Acknowledgements

The research was supported, in part, by grants H2020-DS-2017-780108 (FENTEC) and H2020-ICT-2016-730929 (mF2C).

References

1. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. pp. 326–349. ITCS'12, ACM, New York, NY, USA (2012)
2. Brands, S., Demuynck, L., Decker, B.D.: A practical system for globally revoking the unlinkable pseudonyms of unknown users. In: Proceedings of the 12th Australasian Conference on Information Security and Privacy. pp. 400–415. ACISP'07, Springer-Verlag, Berlin, Heidelberg (2007)
3. Brands, S.A.: An Efficient Off-line Electronic Cash System Based On The Representation Problem. Tech. rep., CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands (1993)
4. Brickell, E., Camenisch, J., Chen, L.: Direct Anonymous Attestation. In: Proceedings of the 11th ACM Conference on Computer and Communications Security. pp. 132–145. CCS'04, ACM, New York, NY, USA (2004)
5. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: Proceedings of the 39th IEEE Symposium on Security and Privacy 2018. pp. 315–334. SP'18, IEEE, San Francisco, CA, US (2018)
6. Camenisch, J., Chen, L., Drijvers, M., Lehmann, A., Novick, D., Urian, R.: One TPM to Bind Them All: Fixing TPM 2.0 for Provably Secure Anonymous Attestation. In: Proceedings of the 38th IEEE Symposium on Security and Privacy. pp. 901–920. SP'17, IEEE, NY, USA (2017)
7. Camenisch, J., Damgård, I.: Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In: Advances in Cryptology — ASIACRYPT 2000. pp. 331–345. ASIACRYPT'00, Springer Berlin Heidelberg, Berlin, Heidelberg (2000)
8. Camenisch, J., Drijvers, M., Lehmann, A.: Universally Composable Direct Anonymous Attestation. In: Public-Key Cryptography – PKC 2016. pp. 234–264. PKC'16, Springer Berlin Heidelberg, Berlin, Heidelberg (2016)

9. Camenisch, J., Groß, T.: Efficient attributes for anonymous credentials. In: Proceedings of the 15th ACM Conference on Computer and Communications Security. pp. 345–356. CCS'08, ACM, New York, NY, USA (2008)
10. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: Public Key Cryptography – PKC 2009. pp. 481–500. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
11. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology. pp. 93–118. EUROCRYPT'01, Springer-Verlag, London, UK, UK (2001)
12. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Advances in Cryptology – CRYPTO 2004. pp. 56–72. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
13. Camenisch, J., Shoup, V.: Practical verifiable encryption and decryption of discrete logarithms. In: Advances in Cryptology — CRYPTO 2003. pp. 126–144. CRYPTO'03, Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
14. Chaum, D., Pedersen, T.P.: Wallet Databases with Observers. In: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology. pp. 89–105. CRYPTO'92, Springer-Verlag, London, UK, UK (1993)
15. Cramer, R., Damgård, I.: Zero-Knowledge Proofs for Finite Field Arithmetic, or: Can Zero-Knowledge be for Free? In: Advances in Cryptology — CRYPTO 1997. pp. 424–441. Springer-Verlag, Berlin, Heidelberg (1997)
16. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Advances in Cryptology – CRYPT0 1994. pp. 174–187. CRYPT0'94, Springer-Verlag, Berlin, Heidelberg (1994)
17. Damgård, I., Fujisaki, E.: A Statistically-Hiding Integer Commitment Scheme based on Groups with Hidden Order. In: Advances on Cryptology – ASIACRYPT 2002. pp. 125–142. Springer-Verlag, Berlin, Heidelberg (2002)
18. Digital Signature Standard: https://csrc.nist.gov/csrc/media/publications/fips/186/3/archive/2009-06-25/documents/fips_186-3.pdf
19. emmy - Library for Zero-Knowledge Proofs: https://github.com/xlab-si/emmy
20. FENTEC Project Homepage: http://fentec.eu/
21. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Proceedings on Advances in cryptology – CRYPTO'86. pp. 186–194. Springer-Verlag, London, UK, UK (1987)
22. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems. In: Proceedings of the 17th Annual ACM Symposium on Theory of Computing. pp. 291–304. STOC'85, ACM, New York, NY, USA (1985)
23. gRPC: https://grpc.io/
24. Hyperledger Fabric: https://wiki.hyperledger.org/projects/Fabric
25. IBM Bluemix: https://console.bluemix.net/catalog/
26. IBM Identity Mixer (idemix): https://www.zurich.ibm.com/identity_mixer/
27. Idemix: https://github.com/hyperledger/fabric/tree/release-1.2/idemix
28. In-browser Cryptography: https://tonyarcieri.com/whats-wrong-with-webcrypto
29. Indy Crypto: https://github.com/hyperledger/indy-crypto
30. Intel Enhanced Privacy ID (EPID) Security Technology: https://software.intel.com/en-us/articles/intel-enhanced-privacy-id-epid-security-technology
31. JSON Web Tokens: https://jwt.io/

32. K. Bemmann and J. Blömer and J. Bobolz and H. Bröcher and D. Diemert and F. Eidens and L. Eilers and J. Haltermann and J. Juhnke and B. Otour and L. Porzenheim and S. Pukrop and E. Schilling and M. Schlichtig and M. Stienemeier: Fully-featured anonymous credentials with reputation system. In: Proceedings of the 13th International Conference on Availability, Reliability and Security. pp. 42:1–42:10. ARES'18, ACM, New York, NY, USA (2018)
33. Lipmaa, H.: On Diophantine Complexity and Statistical Zero-Knowledge Arguments. In: Advances on Cryptology – ASIACRYPT 2003. pp. 398–415. ASIACRYPT'03, Springer-Verlag, Berlin, Heidelberg (2003)
34. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym Systems. In: Proceedings of the 6th Annual International Workshop on Selected Areas in Cryptography. pp. 184–199. SAC'99, Springer-Verlag, London, UK, UK (2000)
35. Marketing Week: Consumers are dirtying databases with false details. https://www.marketingweek.com/2015/07/08/consumers-are-dirtying-databases-with-false-details/
36. mF2C Project Homepage: http://www.mf2c-project.eu/
37. Nakanishi, T., Fujii, H., Hira, Y., Funabiki, N.: Revocable group signature schemes with constant costs for signing and verifying. In: Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography – PKC 2009. pp. 463–480. PKC'09, Springer-Verlag, Berlin, Heidelberg (2009)
38. OpenID Specifications: https://openid.net/developers/specs/
39. PBKDF2: https://en.wikipedia.org/wiki/PBKDF2
40. Pedersen, T.P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Proceedings on Advances in cryptology – CRYPTO'91. pp. 129–140. Springer-Verlag, London, UK, UK (1992)
41. Protocol Buffers: https://developers.google.com/protocol-buffers/
42. Sabouri, A., Krontiris, I., Rannenberg, K.: Attribute-based credentials for trust (abc4trust). In: International Conference on Trust, Privacy and Security in Digital Business. pp. 218–219. Springer (2012)
43. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology. pp. 239–252. CRYPTO'89, Springer-Verlag, Berlin, Heidelberg (1990)
44. Security Assertion Markup Language (SAML) V2.0 Technical Overview: https://wiki.oasis-open.org/security/Saml2TechOverview
45. Technical introduction to IRMA: https://credentials.github.io/
46. The Intel(R) Enhanced Privacy ID Software Development Kit: https://github.com/Intel-EPID-SDK/epid-sdk
47. TOR Project: https://www.torproject.org/
48. Trusted Computing Group: https://trustedcomputinggroup.org/
49. U-Prove: https://www.microsoft.com/en-us/research/project/u-prove/
50. Verifiable Claims Working Group: https://www.w3.org/2017/vc/WG/
51. WS-Federation: https://msdn.microsoft.com/en-us/library/bb498017.aspx