Privacy-preserving Blockchain-based Systems for Car Sharing Leveraging Zero-Knowledge Protocols

Ivan Gudymenko*, Asadullah Khalid†, Hira Siddiqui‡, Mujtaba Idrees‡, Sebastian Clauß*, André Luckow§, Manuel Bolsinger§ and Daniel Miehle§
*T-Systems Multimedia Solutions GmbH {Ivan.Gudymenko, Sebastian.Clauss}@t-systems.com,
†Hochschule Mittweida, University of Applied Sciences {Akhalid}@hs-mittweida.de,
‡Technische Universität Dresden {Mujtaba.Idrees, Hira.Siddiqui}@mailbox.tu-dresden.de,
§BMW Group {Manuel.Bolsinger, Daniel.Miehle}@bmw.de, {Andre.Luckow}@bmwgroup.com

***Abstract*—Blockchain-based systems open new opportunities for the IT business and society if designed and implemented properly. One of the essential challenges on their way to production readiness are severe privacy concerns and hence stringent compliance regulations. In this paper, we investigate how different privacy-preserving techniques leveraging zero-knowledge proof protocols and anonymous credentials can be used to tackle this issue on the example of a car sharing use case. Our approach considers a blockchain ecosystem enabling different parties including governmental authorities and automobile industry partners to cooperate ensuring end user privacy protection. The implemented proof-of-concept is leveraging Ethereum ZoKrates and Indy technologies respectively. This project is a joint initiative of T-Systems Multimedia Solutions GmbH and BMW Group.**

***Index Terms*—anonymous credentials, zero knowledge proofs, Hyperledger Indy, blockchains, Ethereum ZoKrates, privacy protection**

I. Introduction

New mobility services such as car sharing are rapidly gaining popularity, especially in large cities. For example, ShareNow is a car sharing company belonging to BMW Group and Daimler. As of 2019, ShareNow had 4.2 million registered customers, 0.5 to 1 million new customers every year, and more than 100.000 car rentals per day. Within the registration process, every new customer has to undergo an online video identification during which the validity of the driving license is actively checked. Depending on the country regulations, this must be repeated every 36 months. During this process, the photos of the national ID card and the driving license are transmitted. In this context, quite often more information pieces are disclosed than would be necessary for the registration process itself, since selective disclosure of the respective attributes is not possible in the classic scenario. Furthermore, customers are asked to confirm the validity of their driving license on each rental. Therefore, the classic privacy compliant processing of the received data pieces incurs organizational, technical and hence financial overhead. The situation gets even more complex when different car sharing companies cooperate to form a mobility ecosystem and possibly a network of value-added services. In order to tackle these aspects, BMW Group and T-Systems Multimedia Solutions GmbH conducted a technical evaluation of the newest privacy-preserving technologies in the context of consortial, blockchain-based car sharing systems. The focus was made on the application of selective attribute disclosure to the car sharing domain, which is expected to provide for privacy-friendly process management.

II. Privacy preserving driving license disclosure: a general system setup

The generic car sharing use case is defined in Figure 1.

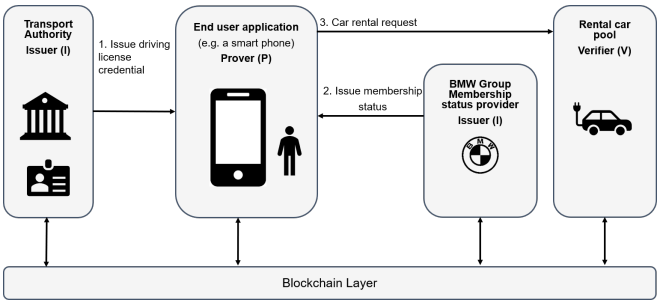

Fig. 1. The generic car sharing use case considered in the paper.

The main actors considered in the target use case essentially represent the core entities of a digital credential system, namely an issuer, a prover and a verifier, see Table I.

TABLE I
Main actors

Actor	Description
Issuer (I)	Issues and revokes the credentials (transport authority KBA, and BMW Group)
Prover (P)	The entity being issued the credentials and proving them (represents the end user)
Verifier (V)	Checks the validity of the prover's credentials (service provider)

In our system, two type of issuers are considered: the governmental transport authority (Kraftfahrt-Bundesamt, KBA) and BMW Group. The former (KBA) issues digital credentials certifying the validity of a driving license in form of well-defined attributes (e.g. driving license class, etc.). BMW Group issues an attribute certifying the user's membership in the BMW Group mobility program which enables the usage of certain services such as car sharing from the BMW Group mobility pool. Prior to the issuing phase, an end user has to be authenticated in some way. For example, conventional means

of end user authentication including electronic identification based on government identity card could be leveraged. This process is, however, out of scope of this paper. Having identified and authenticated the end user and checked the existence of a driving license record, the transport authority (KBA) issues a credential set consisting of well-defined attributes. For the sake of example, assume the following categories of attributes within the driving license context: a driving license class (A, B or C) and the current driving experience (below 3 years, 3 to 5 years and over 5 years). Similarly, the BMW Group membership attributes could be something like partner status i.e. silver, golden, and platinum. The attribute categories are summarized in Table II. The issued attributes can then be selectively disclosed to different service providers according to the certain service consumption requirements (e.g. in form of a defined schema). For example, a person having a class B driving license being in its possession for the time period between 3 to 5 years and having a BMW Group membership status "gold" would maintain the following attributes set: $\mathcal{A} : [B, 3 \leqslant E \leqslant 5, gold]$.

This aforementioned use case has been validated and implemented leveraging Ethereum-based ZoKrates approach (see Section III) and Hyperledger Indy (see Section IV).

TABLE II
ATTRIBUTE CATEGORIES

Category	Options
Driving license class	A, B, C
Driving experience	$E < 3$, $3 \leqslant E \leqslant 5$, $E > 5$
BMW Group membership status	silver, gold, platinum

III. SYSTEM IMPLEMENTATION LEVERAGING ZOKRATES FRAMEWORK

In order to implement the target car sharing use case in the Ethereum context, a custom attribute scheme was devised and the zero-knowledge functionality of ZoKrates framework was leveraged.

A. The generic issuing process

The issuing process adheres to the following simple procedure.

1) The prover P generates a secret which will be a random number r (private input for further proofs)
2) The prover applies a secure hash function to r and delivers it to the issuer: $P \rightarrow I : h(r)$
3) Having previously identified the issuer, the issuer I can now label the delivered value $h(r)$ as a attribute certifying, for example, the driving license class (A, B or C), sign it and finally write the result to the blockchain. The outcome of the issuance phase, therefore, is a signed attribute having the following structure $A = Sign_{IssuerKey}\,(class = B, h(r))$. The Issuer's wallet address under which the attribute has been issued into the chain can be considered a signature in this context (in a loose sense).
4) The process is repeated for further attributes. Finally, the prover maintains a private array of secure random numbers $R : \{r\}$ that were used to generate the attribute together with the corresponding public attributes $\mathcal{A} : \{A\}$.

B. Attributes verification and service consumption

In order to consume a certain service such as to hire a car from the BMW Group car pool, an end user (acting as a prover) has to convince the car terminal (a verifier) that he is in possession of certain attributes. For example driving license class B, driving experience between 3 to 5 years and has a BMW Group membership status gold. To do this, three corresponding attributes are revealed to the verifier (a car terminal) and proved, namely:

- $A_1 = Sign_{IssuerKey}\,(Class = B, h(r_1))$
- $A_2 = Sign_{IssuerKey}\,(3 \leqslant E \leqslant 5, h(r_2))$
- $A_3 = Sign_{IssuerKey}\,(BMWstatus = gold, h(r_3))$

The proving process is essentially reduced to proving the knowledge of the corresponding hash preimage r encoded into the proof and known only to the prover.

C. ZoKrates Framework

ZoKrates [1] framework is a toolbox for zkSNARKS on Ethereum. It is used to generate zero knowledge proofs and enables verifiable computation. ZoKrates itself basically constitutes of off-chain computation steps.

Initialization phase. The first phase of our system is the initialization phase which comprises of 5 main steps:

1) *Program compilation*. Implemented program that proves the knowledge of our problem is compiled to ZoKrates arithmetic circuit to generate a witness.
2) *Trusted setup*. Generate a trusted setup[1] for the compiled arithmetic circuit. As a result of this trusted setup, a proving key and a verification key are generated. The proving key is sent to the prover.
3) *Verifier contract generation*. Generate a *verifier* smart contract, the proof is basically sent to the verifier contract for proof verification.
4) *Deployment.* The generated verification smart contract is deployed on the blockchain, to enable on-chain proof verification.
5) *Issuer Registration.* Finally this registers the issuers, that in our use case are BMW Group and KBA, on the blockchain (via a smart contract).

Attributes Issuance. The second phase of our system is the selection of characteristics/user data and use it to request membership issuance from respective authorities i.e. BMW Group

[1]In our prototype it is assumed to be executed by a trusted setup entity which safely deletes the auxiliary material after system setup. More on security assumptions of ZoKrates initialization process can be found here [2].

or KBA. The process has been described in Section III-A. For actual ZKP-relevant steps, the respective smart contract deployed during the initialization phase is called.

Attributes Disclosure Finally, in order to enable service consumption, a user selectively discloses the attributes against a car terminal to rent a vehicle.

1) *Attributes validity proof in zero knowledge.* First the car sharing service checks if the user is the real owner of the credentials presented in the form of the ZKP proof.
2) *Verify driver's attributes* The car sharing services compares the driver's attributes with the previously defined required attribute schema (e.g. driving license class, experience and BMW Group partner status).

D. ZoKrates-based Prototype Description

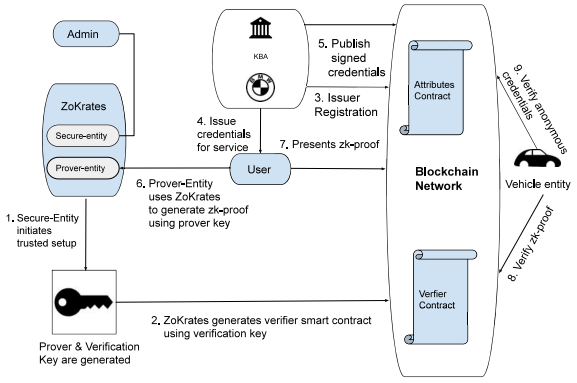

Fig. 2. Ethereum/ZoKrates-based system overview

Figure 2 represents the relevant system entities (*secure trusted entity, issuer, and prover*) together with an on-chain *verifier smart contract* which technically depict the use case.

1) *Secure entity.* This entity is run by the admin to:
   - compile the arithmetic circuit (the implemented program) to generate a preimage.
   - execute the trusted setup phase, using the preimage, in order to generate the prover and verification key.
   - share the prover key with the end user (prover).
   - generate the verifier smart contract using the verification key.

   The Secure entity runs only once at system initialization.
2) *Truffle entity.* Truffle is used to deploy the verifier smart contract for future verification sessions in zero-knowledge (done once per defined attribute schema).
3) *Prover entity.* This entity is run by the end user (prover) to generate the proof (executed on each verification session).
4) *Verifier smart contract.* The verifier smart contract convinces the verifier entity (a car terminal) to authorize the requested service consumption (called on each verification session).

IV. System implementation leveraging Indy Framework

Selective attribute disclosure required by the target use case can be implemented leveraging privacy-preserving identity management approaches which inherently support correlation-resistant attribute revelation functionality. In the blockchain context, this challenge has been efficiently addressed in Hyperledger Indy project [3] [4]. In traditional centralized identity management systems, the defined central organization is the root of trust and holds user's data whereas Indy puts end users in charge of their data using blockchain as an underlying trust provider. Various organizations participate in private blockchain networks and interact with each other to securely prove their identity and attribute claims in zero knowledge. The Indy ecosystem inherently provides for basic building blocks to implement the target use case in a generic way [5].

A. Indy-based System Setup

In order to implement the target use case, the endemic building blocks of Indy have been leveraged. The governmental authorities including the transport authority, which issues driving license credentials, form a governmental blockchain network. Automobile industry partners, service providers, and car sharing companies maintain their own blockchain network. Both blockchain networks are special-purpose blockchains targeted to distributed identity management (leveraging Plenum protocol [6]). The advantage of having domain specific blockchain networks, such as the ones for governmental purposes and service providers, lies in the fact that they can be designed based on potentially rather diverging assumptions, trust paradigms, etc. (e.g. different governance models, etc.). Each service provider can translate the respective business logic requirements into specific attribute schemes which generically define the attributes to be proven by the verifier for service consumption (e.g. renting a car). In our prototype, the obtained credentials are locally maintained on the user side. Figure 3 summarizes the high-level architecture of the designed Indy-based prototype.

B. Indy Framework: a short overview

Hyperledger Indy specifies the terminology and design patterns for decentralized identity along with an implementation of these concepts which act as a building block for many applications of decentralized identity management including the car sharing use case presented in this paper.

Decentralized Identifiers: For every entity at least one Decentralized Identifier (DID) is created in the Hyperledger Indy ecosystem. There are 2 types of DIDs: (1) pairwise DIDs, which are not published on the ledger and used for 1:1 interaction of entities, making them correlation-resistant; and (2) public DIDs, which are published on the ledger and are accessible from any entity making them decentralized and resolvable via blockchain.

Roles: Every entity in Indy has a role which defines its capabilities in the network. Indy follows a hierarchical model of roles [7] and the ones relevant to our workflow are:

- *Trustee*: The trustee role represents the governing bodies in the network. The organizations with this role can (1) add or remove trustees and steward (2) allow an entity the get the role of a trust anchor.
- *Steward*: The steward role allows the organizations to run their nodes in the blockchain network and can grant an entity the role of trust anchor.
- *Trust anchor*: The trust anchor role is assigned to the member organizations of the consortium who issue credentials to the users.

Transactions: Indy ledger has various transaction types specifying the interaction on blockchain network. The transactions relevant to our workflow are NYM, SCHEMA and CLAIM_DEF. The details can be found here [8].

C. Indy Technology Stack

The technology stack for this solution uses Indy SDK, Plenum-based blockchains [9] deployed on docker containers and Java-based Indy clients.

D. System Design and workflow

This solution enables a user to get access to a car from the BMW Group mobility pool by selectively disclosing only the relevant pieces of information (in form of attributes) pertaining to the driving license and BMW Group membership status. As already mentioned above, the core workflow involves two blockchain networks, namely the governmental one (including the transport authority) and the one of automobile industry partners (including BMW Group). In our Indy-based system design, six interacting parties can be defined: Governmental Network Trustee, Automobile Industry Partners Network Trustee, Transport Authority (KBA) Trust Anchor, BMW Group Trust Anchor, a Car Terminal and an End User.

Involved Blockchain Networks

- *Governmental Network* essentially represents a consortium of governmental authorities. The trust anchors of all government agencies are onboarded on this network by the governmental trustee including the transport authority (KBA) trust anchor.
- *Automobile Association Network*: This network is a consortium for automobile companies. The trust anchors of all automobile agencies are onboarded on this network by the automobile trustee. BMW Group trust anchor is a part of this network.

Interacting parties

- *Governmental Network Trustee*. The governmental trustee is the highest role in the Governmental Network and is responsible for onboarding stewards and trust anchors into the network. Its public DID is published to the ledger at network creation.
- *Automobile Association Trustee*. The automobile association trustee is the highest role in the automobile association network and is responsible for onboarding stewards and trust anchors into the network. Its public DID is published to ledger at network creation.
- *KBA Trust Anchor*. KBA trust anchor issues driving license attributes to end users. It has to be initially onboarded onto the governmental network by requesting the trust anchor role from the respective trustee. After being granted the role, it publishes the schema and claim definition describing the generic structure of a driving license to be issued in future. Upon a user request to obtain driving license attributes, pairwise DIDs are created (see Section IV-B) and KBA issues a driving license to the user which in turn gets stored in a user's wallet.
- *BMW Group Trust Anchor*. BMW Group trust anchor issues the membership status to its users. It first onboards onto the automobile association network by requesting the trustee for trust anchor role. After being granted the role, it publishes the schema and claim definition to generate membership attributes. On membership issuance request, pairwise DIDs are created similarly to the case above and BMW Group issues a membership certificate to the user, which is also stored in the user's wallet.
- *A car terminal*. The car acts as a verifier for the proof submitted by user. It first announces the required attributes to be revealed/proved in zero-knowledge in order to obtain the service. In the current case, a proof should be of composite nature and include the statements about the status of driving license and BMW Group membership status attributes. The user app then reads the proof requirements, constructs a proof response using its credentials and sends it to the car. The car then verifies the submitted proof information by firstly consulting the respective revocation registries for the provided credentials. Revocation information can be fetched lazily enabling off-chain and offline verification. The implementation of revocation mechanism is out of scope of this paper. Secondly, integrity checks on the user's proof are performed using the public keys of credential issuers (KBA trust anchor and BMW Group trust anchor) and the corresponding claim definitions published on the ledgers.
- *User*. The end user gets credentials from both KBA and BMW Group to get access to the car. It contacts KBA and gets a driving license credential. Similarly, it gets a membership credential from BMW Group. Once the client has both the credentials, it reads the proof details that car has announced, constructs a composite proof response according to the requirements and presents it to the car. After verification, access will be granted or denied based on whether the proof verification was successful or not.

V. Attributes Revocation

Should certain attributes lose their validity, they have to be revoked. The revocation process has to be efficient and transparent for every participant in the system. The Ethereum-based prototype with ZoKrates and the Indy one handle revocation differently. In the implemented Ethereum/ZoKrates prototype, revocation is performed by the issuer in that the respective

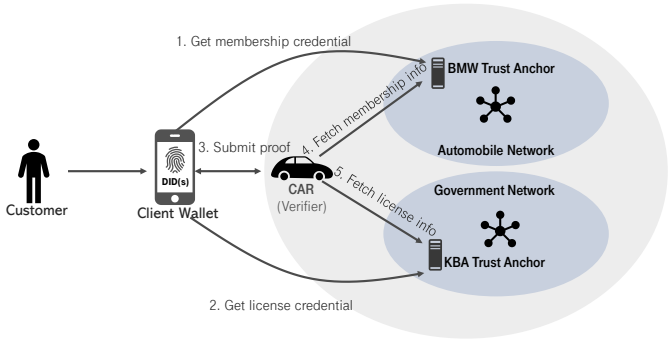

Fig. 3. A high-level overview of Indy-based system design

attributes are labeled as "revoked" and published on the blockchain. Whereas in Indy, cryptographic accumulators [10] are used to handle revocation and published on the blockchain as revocation registeries. A cryptographic accumulator is a one way membership function enabling membership checks without revealing the individual identities in this set.

VI. Results and discussion

Ethereum/ZoKrates and Indy both support the privacy-preserving car sharing ecosystem use case. Their methodologies, however, are widely different. Indy has strict hierarchical roles and the capabilities of every entity in the ecosystem is determined by the role they have. In Ethereum/ZoKrates, hierarchy is simpler and capabilities of different entities can be determined through dedicated smart contracts and are not a part of the ZoKrates ecosystem itself.

In our ZoKrates-based implementation, a simple protocol was devised leveraging the proof of knowledge of hash preimage encoded in the issued credential. The credential structure and the proof logic is in turn encoded into the respective smart contract which acts as an on-chain verifier. This approach is rather inflexible in comparison to Indy, since it does not directly provide for arbitrary composite proofs based on the already issued attributes. In contrast, Indy-based design allows for this and the obtained credentials can be flexibly reused to dynamically meet the verification schema.

Indy gives clients more flexibility to create composite proofs using credentials acquired from issuers possibly residing on an arbitrary number of different blockchain networks. In classic Ethereum-based systems leveraging ZoKrates, all actors need to be present on the same blockchain network for system to work.

Similarly, Indy supports off-chain proof verification and revocation checks (based on lazily fetched information) whereas in ZoKrates verification must be done on-chain using smart contracts. A comparison of the underlying construction of both technologies is given in Table III.

A. Performance assessment

Conceptually the interaction of the entities can be roughly divided into 3 categories: onboarding, credential/proof generation and proof verification. We benchmarked these workflows

TABLE III
ZoKrates vs. Indy: a conceptual evaluation

Criteria	ZoKrates	Indy
Blockchain tech.	Ethereum	Indy Plenum
ZKP engine	SNARKS	Idemix (as in Sovrin)
Credentials mgmt	Off-chain	Off-chain (Wallet)
Proof Verification	Smart Contract, On-chain	Claims Ver., Off-chain
Proof generation	Off-chain	Off-chain
Proof arguments	Defined by a Smart Contract	Schema and Claims
Identity mgmt.	Ethereum Addr. or Custom	DID, inherent impl.

on Intel Core i7-4600U CPU @2.10GHz machine with 8GB RAM, Windows 10 (Enterprise Edition) for both Indy and ZoKrates, see Figure 4.

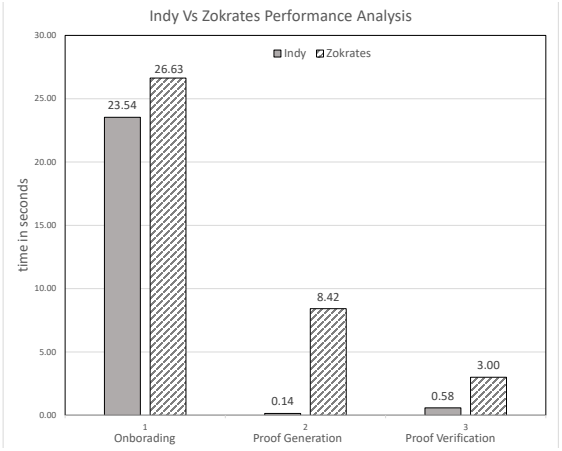

Fig. 4. A comparative performance analysis of system implementation using Ethereum/ZoKrates and Indy

Onboarding takes nearly the same time for both technologies. Proof/credential generation in Indy is roughly 60x faster and proof verification is 5x faster compared to ZoKrates. Proof verification is the most recurring operation in the system. In Indy, it is computed off-chain hence its positive impact on performance would be significant in (near) real time applications. Therefore, Indy-based implementations are inherently more efficient and performant if similar use cases are considered.

VII. Related Work

Hyperledger Indy and ZoKrates both provide alternatives to manage information without revealing private attributes which is beneficial for government and corporate trusts. Various projects have been proposed and developed in this domain. In [11], a privacy-preserving KYC scheme leveraging zero knowledge proofs in Ethereum context for financial services is presented. The approach is similar to our implementation except that it handles revocation differently. The paper presents two use cases based on this model, namely a KYC-compliant exchange and a KYC-compliant token. The authors of [12] provide a solution for a classic use case of transparent and secure electronic voting system. The approach presented in

this paper also involves a protocol developed on blockchain technology, except that it offers anonymity of voter transactions by leveraging an anonymous payment scheme called Zcash which relies on zk-SNARKs used to generate zero-knowledge proofs. The authors of [13] present an Indy-based approach to streamline the operations of states of British Columbia and Ontario by managing decentralized identities and trusted credentials. The motivation for this lies in the fact that the Canadian government spends an estimated CAD 10 Billion a year for unnecessary operations because of red tape. In [14], privacy-preserving grid balancing is presented based on weather forecasts leveraging Indy ecosystem to predict energy production and then to offer incentives to electric vehicle users to charge when energy is in surplus. Zero knowledge proofs can also be beneficial in the so-called cash transfer programs (CTPs) for humanitarian aids at times of natural disasters. Privacy and data-protection of the most vulnerable is an essential requirement of CTPs, whereas not many of the humanitarian information management systems incorporate privacy-by-design. Due to this, Hyperledger Indy was proposed as a solution in [15].

VIII. Conclusion and Future Work

Blockchain-based systems are rapidly gaining attention. However, one of the main challenges on their way to production readiness are severe privacy concerns and hence stringent compliance regulations. In this paper, two different proof-of-concept systems have been presented demonstrating how the privacy issues in such context can be tackled. As a target use case, a car sharing domain within a blockchain-based ecosystem incorporating governmental authorities such as a transport authority (Kraftfahrt-Bundesamt/KBA) and automobile industry partners was chosen. In order to enforce the privacy requirements, the cryptographic techniques based on anonymous credentials and zero-knowledge proofs were used. The proof-of-concepts are implemented leveraging two technologies, namely Ethereum with ZoKrates and Hyperledger Indy. The first solution based on ZoKrates uses zkSnarks for zero-knowledge proof generation and Ethereum blockchain for proof verification. In ZoKrates, the private information which needs to be proved in zero-knowledge is specified in a domain specific language that generates an arithmetic circuit of a proof. It provides an independent environment for off-chain proof generation. Furthermore, it also generates a verification smart contract which is then deployed on the blockchain for on-chain proof verification. The smart contract functionality can be utilized by various applications and it is interoperable with other blockchains based on Ethereum.

The second solution is based on Hyperledger Indy, a dedicated ecosystem for decentralized identity management leveraging anonymous credentials. Indy replicates real-world organizational structure and holistically addresses challenges imposed by decentralized cooperation and privacy-preserving distributed identity management.

For example, governmental organizations and automobile industry partners do not have share a single blockchain network and can maintain their own ecosystems with possibly different trust assumptions and governance models. Moreover, private data pieces/private attributes reside on the end user side with the end user controlling the granularity of data revelation on per case basis. Indy provides efficient revocation strategies using cryptographic accumulators and the possibility to lazily fetch cryptographic material for verification. As a result, a user and a car could perform the validation process locally by e.g. communicating via Bluetooth or Near Field Communication (NFC) by this decoupling the on-chain processes for which a stable Internet connection is required from the time-critical ones.

Summing up, Indy-based solutions are likely to be more flexible and efficient in a wide variety of use cases not requiring compatibility with the classic blockchain technologies such as Ethereum. In case the latter is important (e.g. Ethereum-based ecosystem already exists), the suggested approach leveraging ZoKrates can be applied.

Acknowledgement

We thank our T-Systems colleagues Tobias Wohland, Nikolaos Saklampanakis and Alexander Ebeling who provided insight and expertise that greatly assisted the research.

References

[1] Zokrates. a toolbox for zksnarks on ethereum. [Online]. Available: https://zokrates.github.io/

[2] B.-S. et al., "Succinct non-interactive zero knowledge for a von neumann architecture," in *Proceedings of the 23rd USENIX Conference on Security Symposium*, ser. SEC'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 781–796.

[3] The official sdk for hyperledger indy. [Online]. Available: https://github.com/hyperledger/indy-sdk

[4] An introduction to hyperledger. [Online]. Available: https://www.hyperledger.org/wp-content/uploads/2018/08/HL_Whitepaper_IntroductiontoHyperledger.pdf

[5] Hyperledger indy node documentation. [Online]. Available: https://hyperledger-indy.readthedocs.io/en/latest/index.html

[6] Introduction to hyperledger business blockchain design philosophy and consensus. [Online]. Available: https://www.hyperledger.org/wp-content/uploads/2017/08/Hyperledger_Arch_WG_Paper_1_Consensus.pdf

[7] Hyperledger indy network roles and permissions. [Online]. Available: https://github.com/hyperledger/indy-node/blob/master/docs/source/auth_rules.md

[8] Hyperledger indy transaction types. [Online]. Available: https://github.com/hyperledger/indy-node/blob/master/docs/source/requests.md

[9] Plenum byzantine fault tolerant protocol. [Online]. Available: https://github.com/hyperledger/indy-plenum

[10] Hyperledger indy credential revocation using cryptographic accumulators. [Online]. Available: https://github.com/hyperledger/indy-sdk/blob/master/docs/concepts/revocation/cred-revocation.md

[11] S. T. Alex Biryukov, Dmitry Khovratovich, "Privacy-preserving kyc on ethereum," in *ERCIM-Blockchain 2018*, 2018.

[12] P. Tarasov and H. Tewari. (2017, 10) Internet voting using zcash. [Online]. Available: https://eprint.iacr.org/2017/585.pdf

[13] Bc aims to cut government red tape with hyperledger indy. [Online]. Available: https://www.hyperledger.org/resources/publications/orgbook-case-study

[14] A privacy-preserving approach to grid balancing using scheduled electric vehicle charging. [Online]. Available: https://aaltodoc.aalto.fi/bitstream/handle/123456789/40877/master_Antonino_Antonio_2019.pdf

[15] Self-sovereign identities for scaling up cash transfer projects. [Online]. Available: https://www.121.global/wp-content/uploads/2019/09/Stevens_TUDELFT_Self-sovereign-identities-for-scaling-up-cash-transfer-projects.pdf