Put Your Money Where Your Mouth Is – Towards Blockchain-based Consent Violation Detection

Jonathan Heiss, Max-R. Ulbricht, Jacob Eberhardt
Information System Engineering
TU Berlin
Berlin, Germany
{jh, mu, je}@ise.tu-berlin.de

***Abstract*—Faulty access control in API-based multi-service setups can lead to violations of consent declarations through unauthorized Third Parties. This threatens Service Providers to lose the trust of their Service Consumers and to be exposed to sensitive fines as defined by the GDPR.**

Addressing this problem, in this paper, we propose a novel, blockchain-based approach for enabling economically motivated and technically mediated detection of violations of consent declarations in multi-service setups and derive its legal viability from a thorough analysis of the GDPR. The herein introduced *Violation Detection* mechanism allows for a censorship-resistant and publicly verifiable detection of violations to registered Consent Policies based on off-chain computed violation claims utilizing non-interactive zero-knowledge proofs. The corresponding *System Design* specifies all required roles and artifacts to integrate the Violation Detection mechanism with standard procedures for consent-based access control. The integration of our system supports Service Providers to fulfill legal requirements and, therefore, paves the way towards automated policy violation detection within GDPR-compliant consent-based access control solutions.

***Index Terms*—Blockchain, Privacy, Consent Violation, Incentive, Detection Mechanism, GDPR, Compliance, Access Control**

I. Introduction

The significance of and awareness for lawful consent management has drastically increased since the European Union's General Data Protection Regulation (GDPR) came into force in May 2018. The GDPR clearly specifies legal obligations for the management of Personal Identifiable Information (PII) by Service Providers and defines sensitive fines of up to 20 million Euro or 4% of their annual revenue if these obligations are violated.

Since modern services heavily rely on the integration with Third Party Applications through Application Programming Interfaces (API), strict access control based on the consent of the Service Consumer is fundamental for the protection of PII. However, the implementation and operation of such mechanisms are exposed to human fallacy and, hence, can lead to undetected violations of consent declarations. If such violations remain undetected for long, extensive harm can be caused not only to the affected Application Users but also to the Service Provider which has failed its legal obligations.

The detection of violated consent declarations in operational APIs is, however, non-trivial. As Third Party Applications that can wrongously access PII not intended for them are not legally actionable, there is no incentive to report such violations. Consequently, hidden leaks present a lasting opportunity for malicious Third-Party Applications to abuse the illegitimately obtained PII for illegal or unethical purposes.

Furthermore, the inherent non-transparency and centralization of access control mechanisms present a problem for Service Providers as well as for Application Users. While users are unable to verify the accurate enforcement of their consent declarations and, therefore, have to trust the Service Providers, the providers themselves cannot effectively prove their compliance with legal obligations derived from the GDPR. Appropriate technical solutions to improve this situation are needed and can even represent a competitive advantage for Service Providers. However, approaches to effectively, trustlessly and transparently detect consent violations are currently missing.

To address this issue, we propose a system for economically motivated and publicly-verifiable detection and reporting of consent violations. It allows Service Providers to advertise an economic commitment to GDPR-compliant and consent-based access control. This commitment, deposited as cryptocurrency, serves as an incentive for Third-Party Applications and Application Users that report consent violations and, thereby, prove the Service Providers' non-compliance. To increase transparency and also allow auditability by supervisory authorities, the proposed technically mediated violation claims are verifiable by employing a public Blockchain.

While the design of this blockchain-based system is the primary subject of this paper, we, in particular, provide the following contributions:

- We propose a novel *Violation Detection* mechanism that allows for censorship-resistant and publicly verifiable detection of violations to registered Consent Policies based on off-chain computed violation claims utilizing non-interactive zero-knowledge proofs.

- We introduce a *System Design* that allows for economic commitments to GDPR compliance and specifies all required roles and artifacts to integrate the Violation Detection mechanism with standard procedures for consent-based access control.
- We support Service Providers to fulfill some of the legal obligations required by the GDPR. By integrating the proposed system they
  1) establish an *auditable Archive of Consent Policies* and are, therefore, "able to demonstrate that the data subject has consented to processing of his or her personal data" as required by Article 7(1),
  2) implement mechanisms to "become aware of [...]" personal data breaches and are able to "notify [it] to the supervisory authority" as required by Article 33(1),
  3) can prove the implementation of "appropriate technical and organizational measures to ensure and to be able to demonstrate that processing is performed in accordance with" the GDPR as required by Article 24(1).

The remainder of this paper is organized as follows: In Section II, we introduce consent-based access control and analyze the requirements for GDPR-compliant data processing based on given consent. In Section III, we derive a conceptual consent model from these legal requirements, present the technical representation thereof, and describe the process of detecting violations of consent policies. On this basis, we provide our design objectives, outline our system design, and propose suitable technologies for the system implementation in Section IV. Finally, we discuss some open issues (Section V), present related work (Section VI) and conclude (Section VII).

II. Preliminaries

To provide context for our contributions, in this Section, we first illustrate the common procedure of consent-based access control and then derive legal requirements for consent-based data processing from the GDPR.

A. Consent-based Access Control

In multi-service setups, services can access user data stored at other services through integrated Application Programming Interfaces (API). In such setups, we address the procedure for consent-based access control as illustrated in Figure 1.

A *Service Consumer* is registered at her favorite Social Media Network, here the *Service Provider*. By using this service, she uploads *Personal Identifiable Information* (PII) to it, e.g. personal images. Besides the Social Media Network, the same Service Consumer is also registered at her favorite Image Editing Application, here the *Third Party Application*.

In her role as *Application User*, she now wants to edit one of her images that have been uploaded to the Social Media Network. Therefore, she equips the Image Editing Application with the right to access and process the respective image and sends this consent declaration to the Social Media Network. In return, the Social Media Network issues an access token that authorizes direct access to the image through its API. Utilizing this token, the Image Editing Application can now query the image and legitimately provide the editing functionality to the Application User.

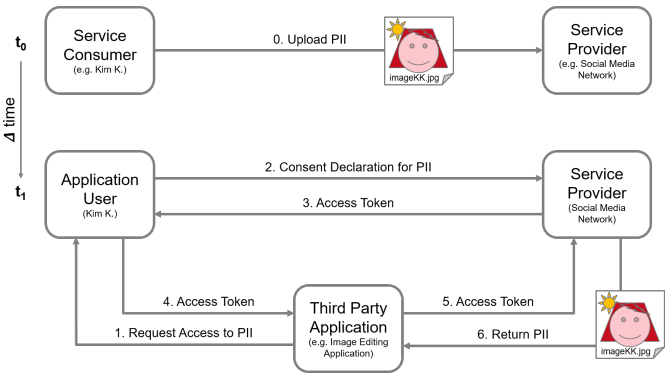

Fig. 1. Standard Procedure for Consent-based Access Control.

B. Legal Requirements for Consent-based Data Processing

Article 5 of the GDPR [1] provides the *Principles relating to processing of personal data*. According to these principles "personal data shall be [...] processed lawfully, fairly and in a transparent manner [...]". If "the data subject has given consent to the processing of his or her personal data for one or more specific purposes" *Lawfulness* can be assumed [1, Article 6 (1(a))]. Additional *Conditions for consent* that have to be fulfilled for consent to be legally sufficient are specified in Article 7. E.g., given consent has to be easily retractable and the controller that initially collects the personal data has to ensure to be able to demonstrate that a specific data subject has actually consented to the processing of his or her data [1, Article 7 (1)].

Besides the fact that legally sufficient consent must be freely given, the consent itself must be a "specific, informed and unambiguous indication of the data subject's wishes" and needs to be provided "by a statement or by a clear affirmative action" as defined in Article 4 of the GDPR [1]. Specificity, informedness, and the necessity of a clear affirmative action, therefore, deserve closer examination.

Specificity: Even though the GDPR itself does not explicitly define what requirements *"specific consent"* has to fulfill, it is, however, at least declared that consent has to be given for "one or more specific purposes" [1, Article 6 (1 a))]. Moreover, the Article 29 Data Protection Working Party (Art. 29 WP) advance the view that consent, to be specific, "should refer clearly and precisely to the scope [...] of the data processing. It cannot apply to an open-ended set of processing activities. [...] In other words, blanket consent without specifying the exact purpose of the processing is not acceptable" [2].

Informedness: Within the GDPR itself it is clearly explained what is meant by the term *"informed consent"*. In recital 42 of the regulation [1], it is defined that "[f]or consent to be informed, the data subject should be aware at least of the identity of the controller and the purposes of the processing for which the personal data are intended". Therefore, consent

to be constituted as legally sufficient has at least to ensure that the data subject is aware about who intends to processes their personal data for what reason.

Clear affirmative action: Eventually, a controller is responsible for ensuring that consent is obtained in an unquestionable manner. Acceptable forms, according to recital 32 of the GDPR [1], are oral or written statements, including by electronic means, but also "ticking a box when visiting an internet website" or "choosing technical settings for information society services". Contrary to such clear affirmative actions, "[s]ilence, pre-ticked boxes or inactivity should not therefore constitute consent" [1, Recital 32].

Besides the above mentioned fact that a data subject has to clearly indicate wishes regarding the data processing, it is of particular relevance for our system design to ensure a decent representation of the identity of the party that is going to process the respective PII as well as the purpose of the processing [3].

III. Violation Detection

In the following section, we derive our conceptual consent model from legal requirements set by the GDPR, present the technical representation thereof, and describe the process of detecting violations of consent policies.

A. Conceptual Model & Technical Representation

As shown in the last section, any technical implementation of the legal concept of consent has to feature suitable mechanisms to enable appropriate representations of the identity of the party that is going to perform the data processing as well as the purpose that makes the processing necessary. Thus, for every PII or sets thereof, a statement regarding the combination of an identity and a purpose bound to the respective data has to be obtained from the service user and stored within the system to be considered a legally sufficient consent representation.

If we are going to formalize these requirements, we have to further take into account that any obtained consent statement represents a decision of data subjects for or against the processing of their PIIs for a given combination of processing party and specified purpose. We, therefore, represent a consent statement as a mapping of a 3-tuple containing an identity (id), a processing purpose (p), and the corresponding data (d) to a boolean value, which codifies the decision.

$$\langle id, p, d \rangle \rightarrow bool$$

Since application users should be enabled to consent as precisely as possible, the granularity of possible consent provision schemes has to be taken into account [3]. To represent that there can be multiple processing parties and processing purposes or the data consists of multiple PIIs within a specific set, the data subject defines a function f that encodes its consent:

$$f \colon ID \times P \times D \rightarrow bool$$

We apply f to the combinations of data items and processing purposes for every processing party to derive a codified representation of consent.

$$\begin{pmatrix} (id_1, p_1, d_1) \\ (id_1, p_1, d_2) \\ \vdots \\ (id_2, p_1, d_2) \\ (id_2, p_2, d_2) \\ \vdots \\ (id_3, p_1, d_1) \\ (id_3, p_3, d_3) \\ \vdots \\ (id_i, p_j, d_k) \end{pmatrix} \xrightarrow{(id,p,d,f(id,p,d))} \begin{pmatrix} (id_1, p_1, d_1, true) \\ (id_1, p_1, d_2, true) \\ \vdots \\ (id_2, p_1, d_2, false) \\ (id_2, p_2, d_2, true) \\ \vdots \\ (id_3, p_1, d_1, false) \\ (id_3, p_3, d_3, false) \\ \vdots \\ (id_i, p_j, d_k, true) \end{pmatrix}$$

Fig. 2. Formal Consent Certificate.

As described in Section II-A, an access token is issued by a Service Provider after a Service Consumer consented to the processing of their data through a Third-Party Application for a specific purpose. Formally, this means that the Service Provider implicitly maps identity id and purpose p to an access token AT. Hence, we can simplify our codified consent representation:

$$\begin{pmatrix} (AT_1, d_1, true) \\ (AT_1, d_2, true) \\ \vdots \\ (AT_2, d_2, false) \\ (AT_2, d_2, true) \\ \vdots \\ (AT_3, d_1, false) \\ (AT_3, d_3, false) \\ \vdots \\ (AT_i, d_k, true) \end{pmatrix}$$

Fig. 3. Formal Consent Policy.

This herein derived representation of consent allows for a technically mediated automated check of violations of consent policies. Therefore, we refrain from using more complex consent models in conjunction with specialized privacy preference languages such as YaPPL [4], since the model presented seems to be more than sufficient for our system design. Details about the theoretical process and the foundations for a trustless implementation of the above mentioned automated check of violations of consent policies are sketched in the following section.

B. Detection of Consent Violations

As a first step in the process of detecting and then publicly proving consent violations by a Service Provider in a non-

repudiable way, we introduce a naive Proof-of-Access. In a second step, we improve on this approach by making it privacy-preserving by adding a zero-knowledge property. Finally, we combine this zero-knowledge Proof-of-Access with Consent Policies to detect the occurrence of consent violations.

1) Proof-of-Access: In the context of our work, a Proof-of-Access is an attestation that unambiguously confirms that a specific data item could be retrieved by a party who claims access. Without loss of generality, we assume Service Provider APIs deliver responses in a way that can be mapped to the following format derived from our formal consent policy specification, where we omit processing party and purpose, as they are known to the caller:

$$\begin{pmatrix} d_1 & : & value \\ d_2 & : & value \\ \vdots & & \vdots \\ d_{n-1} & : & value \\ d_n & : & value \end{pmatrix}$$

Fig. 4. Abstract Structure of Service Provider API Responses.

Such a response can directly be turned into a simple Proof-of-Access by requiring the Service Provider to add a digital signature. Then, the signed response can be shown to any Third-Party to prove that the data contained was actually made available through the Service Provider, i.e., could be accessed. However, providing such a proof would expose all data contained in the response.

2) Zero-knowledge Proof-of-Access: The naive Proof-of-Access introduced before would force a prover to reveal potentially sensitive PII with the proof. Clearly, this is undesirable. Thus, we expand on this proposal by replacing concrete values with boolean flags, indicating accessibility. This makes the resulting proof zero-knowledge in the sense that it reveals no information but the fact that a data item's value was accessible in the input data set.

$$\begin{pmatrix} d_1 & : & value \\ d_2 & : & value \\ \vdots & & \vdots \\ d_{n-1} & : & value \\ d_n & : & value \end{pmatrix} \overset{\text{Zero-Knowledge Proof}}{\rightarrow} \begin{pmatrix} d_1 & : & bool \\ d_2 & : & bool \\ \vdots & & \vdots \\ d_{n-1} & : & bool \\ d_n & : & bool \end{pmatrix}$$

Fig. 5. Hiding Transformation of Service Provider Response.

Unlike as in the case of simple Proof-of-Access, where a digital signature was sufficient, the realization of a zero-knowledge Proof-of-Access as described is not trivial. We need to perform a transformation on signed data in a way that the result is trustworthy.

For this task, we leverage ZoKrates [5], a language and toolbox for zero-knowledge verifiable off-chain computations [6], [7]. ZoKrates allows the execution of programs on blockchain-external resources but retains blockchain's trustlessness through enabling the on-chain verification of the program execution's correctness. It does not reveal information used in the processing, e.g., program inputs, with this proof of correctness unless desired.

We employ ZoKrates to prove that a Third-Party Application had access to specific data as part of a response from a Service Provider without having to expose this information. Technically, we use a ZoKrates program to perform the following three steps:

1) **Calculate Hiding Mapping:** First, the ZoKrates program calculates the transformation displayed in Figure 5 to hide sensible values for data items.
2) **Confirm Signature:** Additionally, the ZoKrates program checks the validity of the signature provided with the Service Provider's response. This convinces the recipient of the ZoKrates proof that an actual response from the Service Provider was used as processing input.
3) **Commit to Access Token:** The Access Token AT needs to be protected so that the party who receives a Proof-of-Access does not get access itself. However, since AT encodes identity id and purpose p, this information must still be encoded in the proof. We solve this through committing to an Access Token through a hash function: We compute hash(AT) within the ZoKrates program and publish it with the proof.

As a result, we obtain a zero-knowledge Proof-of-Access that proves which data items a Third-Party Application had access to without revealing the associated values. Formally, the Proof-of-Access consists of the following tuple:

$$(\pi, hash(AT), \begin{pmatrix} d_1 & : & bool \\ d_2 & : & bool \\ \vdots & & \vdots \\ d_{n-1} & : & bool \\ d_n & : & bool \end{pmatrix})$$

Hereby, π is the proof of execution correctness for the ZoKrates program.

3) Determining Consent Violations: We define a consent violation as the situation where a Third-Party Application has access to personal data for which no consent exists. While representing an essential building block, the zero-knowledge Proof-of-Access previously introduced is not sufficient to detect such situations. A Proof-of-Access always needs to be evaluated in the context of an applicable policy with which accessible data may or may not be aligned.

Employing our formalism, we detect violations of a specific consent policy in two steps. In a first step, the consent policy for the Access Token AT committed to in the Proof-of-Access needs to be retrieved. Then, in a second step, the access permissions for the contained data items are compared with the accessible items provided in the Proof-of-Access as depicted in Figure 6:

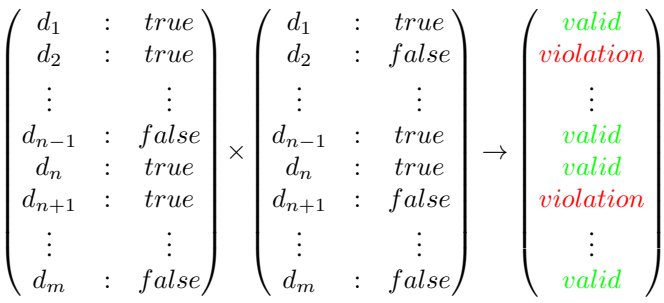

Fig. 6. Violation Check by comparing Proof-of-Access and Consent Policy.

A violation check passes if consent exists for all accessible data items and else fails.

IV. System Design

Given a consent violation detection mechanism that allows for third-party verification while preventing the disclosure of PII, we now describe our system design that integrates this mechanism into an appropriate procedure for consent-based access control. In this Section, we first introduce our design objectives, then outline our system design, and finally propose technologies that we apply for the system implementation.

A. Design Objectives

We design our system around three main objectives: First, the violation detection mechanism must be publicly verifiable and the Service Provider must be able to trustworthily provision a monetary reward as incentive for violation reporting. Therefore, we decide to implement our main system logic on a public, smart contract-enabled Blockchain. The Blockchain should allow for depositing the monetary reward in its native crypto-currency and provide a trustworthy environment for executing smart contracts. This would guarantee transparency, high availability, and censorship-resistance.

Second, the procedure should integrate well with existing setups for consent-based access control. To meet this objective, we design our system procedure with the OAuth 2.0 Implicit Grant protocol [8] in mind that represents an established standard for authorization and access management [9].

Third, the Service Provider should be supported with fulfilling its legal obligations as specified by the GDPR. Of these obligations, the following are specifically addressed by our system design directly or implicitly:

- *Demonstration of Consent:* According to the GDPR [1, Article 7 (1)], the Service Provider must be able of legally proving that consent of Application Users to process PII has been given. For this requirement to be fulfilled an auditable archive of consent policies should be maintained by the Service Provider.
- *Prevention of Consent Violation:* The Service Provider must implement "appropriate technical and organisational measures to ensure and to be able to demonstrate that processing is performed in accordance with this Regulation" [1, Article 24 (1)]. These data protection measures must be integrated into the system "by design and by default" [1, Article 25].
- *Reporting of Consent Violation:* If a consent violation is detected, the Service Provider is obliged to notify the supervisory authority within 72 hours. This notification must at least include a description of the nature of the violation, information about the contact person, likely consequences of the violation, and measures taken to mitigate its possible adverse effects [1, Article 33 (2)].

While infringements to the GDPR can be punished with fines of up to 20 million Euro or 4% of the Service Provider's annual revenue, the GDPR defines specific criteria that are taken into consideration to determine the actual degree of penalty. These include the gravity of the violation (e.g. number of affected users and their suffered damage), actions taken to mitigate the damage of affected users, preventative measures against non-compliance, cooperation with the supervisory authority and the proactive reporting of the violation. Our system design aims at satisfying these criteria and, hence, mitigating potential penalties of the Service Provider.

B. System Design

In the following, we describe our system design as depicted in Figure 7 by guiding through the contained processes step by step.

1) Depositing an Economic Commitment: The Service Provider initiates the violation detection mechanism by depositing an economic commitment to the Blockchain that serves as reward for detectors of consent violations. Therefore, it first deploys the *Commitment Contract* to the Blockchain which locks the reward and implements the computational condition for paying it out, i.e., the successful validation of a violation claim. The Commitment Contract also provides functionality for the registration and management of Consent Policies.

2) Consent Declaration and Authorization: Next, Consent Policies must be registered to the Commitment Contract to allow for the determination of their violation. This is done by the Service Provider who registers a Consent Policy on every consent declaration of a Service Consumer.

As consent must be declared in accordance with the GDPR requirements described in Section II-B, the Third-Party Application first sends a request to the Application User that contains its identity, the processing purpose, the PII of interest, and the *Consent Template* (Step 2.1). This template is a key-value list of all PII stored by the Service Provider and the corresponding consent rule which is 'non-consented' as default for all entries (in Figure 8 depicted as an empty box). To grant consent, the Application User transforms this template into a Consent Certificate by actively changing the consent rules for the requested PII from 'non-consented' to 'consented' (ticked box). Then the Application User signs the certified list of PII-consent rules and sends the Consent Certificate to the Service Provider (Step 2.2).

On receiving the Consent Certificate, the Service Provider generates an *Access Token* (AT) that codifies the identity

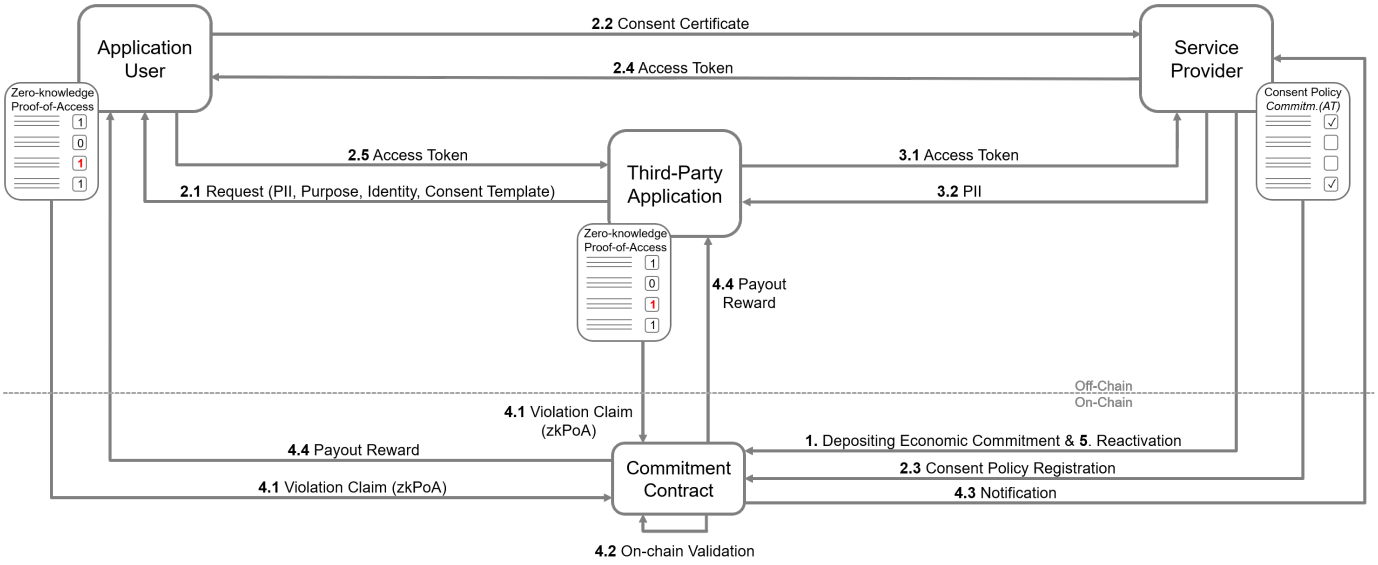

Fig. 7. Overview of the Proposed System Design.

of the requesting Third-Party Application and the purpose for which the Third-Party Application intends to use the requested PII. As this Access Token is required for the on-chain check of consent violations but must also be hidden from unauthorized third-parties, a commitment to the Access Token, e.g., its hash value, is attached to the *Consent Policy* (compare Section III-B3). Both, Access Token commitment and Consent Certificate, complement the Consent Policy which is then submitted to the Commitment Contract (Step 2.3). Once the Consent Policy is registered, the Access Token is issued to the Application User which forwards it to the Third-Party Application (Steps 2.4 and 2.5).

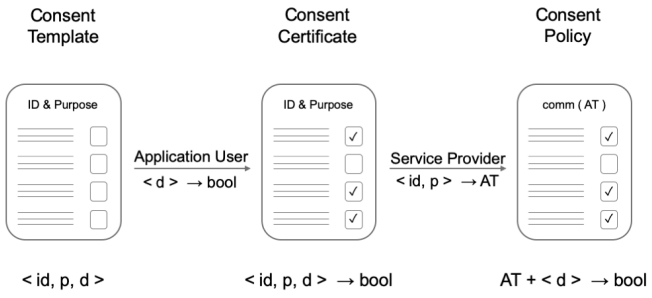

Fig. 8. Stages in Application Users' Consent Representation.

3) Accessing PII: Since the Third-Party Application and the Application User are in possession of the Access Token, they are both able to detect consent violations by reviewing if non-consented PII are returned from API calls. On an authorized API call, the Service Provider returns the requested PII as a list of key-value pairs (Step 3.1 and 3.2) where keys are PII types, e.g., address, name, or personal images, and reference the corresponding PII values. This representation maps to our formal Consent Certificate specification as described in Section III-B1.

4) Violation Detection and Rewarding: If a violation is detected, the detector must submit a valid violation claim to the Commitment Contract to obtain the reward (Step 4.1). This claim is technically represented as *Zero-knowledge Proof-of-Access* and computed off-chain as zero-knowledge proof based on the returned PII list, the Service Provider's signature of this list, and the Access Token (compare Section III-B2).

After the detector has submitted the violation claim, the Commitment Contract now determines its validity (Step 4.2) by comparing the provided Zero-knowledge Proof-of-Access and the corresponding registered Consent Policy. For this check to pass, the zero-knowledge proof within the Proof-of-Access needs to be valid, the provided Access Token commitment equals the one stored in the Consent Policy, and no PII without consent was illegitimately returned (compare Section III-B3).

After the detector has submitted the violation claim, the Commitment Contract now computationally determines its validity as Blockchain state transition (Step 4.2) based on the provided Zero-knowledge Proof-of-Access and the corresponding registered Consent Policy (compare Section III-B3). The violation claim is determined as valid if it is correctly computed, the provided Access Token commitment maps the

one in the Consent Policy, and at least one of the non-consented PII is illegitimately returned.

In case of consent violations, the Service Provider is immediately notified (Step 4.3) and the financial reward is released to the submitter of the violation claim (4.4). Being informed about consent violation, the Service Provider can fix the responsible API leak and notify the supervisory authority about the consent violation.

While the supervisory authority can now audit the violation in hindsight, our system design also allows a blockchain-capable supervisory authority to automatically generate a violation report from the available information which includes the nature, the point of time, and the estimated scope of the violation.

5) Reactivation: After the reward is paid out, the Commitment Contract provides no incentive for the detection of consent violations anymore. To reinitialize the violation detection mechanism, the Service Provider must renew its economic commitment by depositing another reward to the Commitment Contract. The already registered Consent Policies are retained.

C. Implementation Options

After introducing our system design conceptually in the previous Section, we now discuss appropriate technologies that we are taking into consideration for our ongoing system implementation.

As we designed our system towards straight-forward integration with existing consent-based access control setups, we propose to comply with the OAuth 2.0 Implicit Grant Protocol. We ground this decision on three reasons: (1) Most messages in the protocol are exchanged through the Application User which makes the access authorization more transparent to the Application User. (2) It uses a Bearer Token[1] that is first received by the Application User who is, thus, able to access her own personal data through the Service Provider's API. (3) It is one of the most established industry standard for authorization [9].

Further, we propose Ethereum [10] as suitable public Blockchain technology as it is an established standard public Blockchain technology and fulfills our requirements by providing an inherent crypto-currency, Ether, and censorship-resistant smart contract execution. ZoKrates [5], which we consider as critical technology for our consent violation detection mechanism as it facilitates the off-chain computations of zero knowledge proofs and their on-chain verification, also supports Ethereum natively.

The formal specification of consent certificates and policies, as well as the returned PII from the API can easily be realized with common structured file formats, such as JSON or XML.

To compute commitments to Access Tokens, we apply cryptographic hash functions.

V. Discussion

The previously presented system design allows for economical commitments to GDPR-adherence and integrates the Violation Detection mechanism in a common procedure for consent-based access control. In this Section, we discuss the remaining open issues with regards to the prevention of violation disclosure, consent updates and revocations, and system integration tasks.

A. Violation Disclosure as Target for Attacks

In our system design, a Proof-of-Access published to the blockchain in order to prove a consent violation directly exposes the weakness in the Service Provider API: The concrete data item which could be retrieved without consent becomes visible publicly. While publishing this vulnerability may very well be desired as it puts pressure on the Service Provider to quickly deploy a fix, the information could also be used by malicious parties: The API's weak point is visible and targeted attacks trying to extract PII without consent could be launched.

Hence, to mitigate this risk and to give the Service Provider some time to patch its system, the following extension could be implemented: Instead of directly providing a Proof-of-Access on-chain, a Third-Party App subject to consent violation instead commits to that proof on-chain, which reserves its claim. Only after an adequate time period has passed, the Third-Party App publishes the proof to claim its reward.

B. Consent Update and Revocation

As noted in section II-B, Article 7(3) of the GDPR [1] obligates Service Providers to establish mechanisms to ensure that Service Consumers can withdraw their consents at any time. Furthermore, it requires that the withdrawal shall be as easy as to give consent. Since our system is designed in a way that enables fine-granular consent declarations we have to consider options to enable Consumers to not only withdraw consent as a whole but also to change their decision for every single PII represented in the *Consent Certificate*.

Given that within our design, the Service Provider is responsible for the management of consent declarations of its Consumers regarding the usage of their PIIs by Third-Party Applications it has to offer options to withdraw the whole Consent Certificate or alter options on single PIIs. Thus, in both cases, the corresponding Access Token has to be revoked to render the associated Consent Policy invalid as well as to disable API access to the respective PIIs. If a Service Consumer wants not to withdraw the complete Consent Certificate but change one or more options regarding specific PIIs, the old, now invalid Consent Policy acts as a blueprint for a new Consent Certificate with altered settings and a newly issued Access Token. The Service Provider has to register the recent generated Consent Policy to the Commitment Contract as well as transfer the new Access Token to Consumer and Third-Party Application. While the registration of new Policies a versioning mechanism ensures that all old, invalid Consent Policies are still available within the Commitment Contract to establish the aforementioned *auditable Archive of Consent Policies*.

A problem we have to address in this mode of operation is the possibility of Third-Party Applications to store PIIs locally.

[1] A Bearer Token equips any owner with all attached rights [8].

If a User revokes consent for a specific PII but the Third-Party Application has stored a copy thereof, every further processing of this stored PII is to be considered illegal and has to be omitted. Therefore, Third-Party Applications have to ensure to check if a valid Consent Policy exits for all PIIs they tend to process from their local storage.

C. System Integration

While our system is designed for easy integration with existing setups, some changes must be done by the Service Provider including API adjustment, PII determination, and system roll out.

The mapping of existing data schemes to our derived GDPR-compliant consent specification can be realized as simple API overlay that externally issues PII according to our key-value specification. However, the determination of the accessible PII types must be thoroughly planned since, once determined, changes would require all registered Consent Policies to be adjusted.

The violation detection system can then incrementally be rolled out. While new Consent Policies can directly be registered in the system, already existing Consent Policies can be integrated when they are updated by the Application User.

VI. Related Work

To our knowledge, this is the first system design for economically-driven detection of consent violations. With regards to our individual contributions, however, we intersect with and discuss here related research for Blockchain-based incentive mechanism and Blockchain-based and GDPR-compliant access control. While in each of these research fields a large body of work exist, none of the works combines contributions in both fields and only few system approaches share a similar spirit.

As crypto-currency-inherent, public Blockchains lend themselves for advertizing financial incentives, the idea of economically motivating desired behavior has been seized for various purposes including truthful data on-chaining [11], [12] and correct off-chain computations [13], [14]. The most related work in this field is the IKP approach [15] that shares a similar idea but applies it in a different context. It proposes an economically-driven mechanism for the detection of unauthorized TLS certificates issued by lazy or compromised Certificate Authorities in PKIs. However, its purpose, design, and violation detection fundamentally differ from our approach.

Related system approaches for Blockchain-based and GDPR-compliant access control [16]–[19] apply the Blockchain mainly to achieve immutability and transparency. However, they all face the challenge of preventing disclosure of the PII. The work in [16] is motivated by missing transparency in prevailing centralized access management setups. The authors propose a Blockchain-based solution for GDPR compliant data management that enhances transparency by shifting all mechanisms related to GDPR compliance, e.g. access control and authorization, to a permissioned Blockchain. In contrast to our work, here, the system is intended as a substitution not as a supplement of an existing system and also the privacy issue is very differently addressed. A different approach is Advocate [17] which proposes a public Blockchain-aided framework for user-centric and GDPR-compliant control of personal data in IoT environments. While in this approach consent policies are stored off-chain, the Blockchain is applied only for storing and managing their hash values.

VII. Conclusion

In this paper, we proposed a novel, blockchain-based approach that introduces an incentive mechanisms for the detection and reporting of violations of consent declarations in multi-service setups. The core of the herein introduced system design is a novel *Violation Detection* mechanism that allows for a publicly-verifiable and legally viable determination of violations to Consent Policies.

By integrating the proposed *System Design* and, therefore, publicly committing to GDPR-compliant consent-based access control, the Service Providers can increase the trust of their Consumers in the responsible management of their PII. The deposit of an amount of a crypto-currency, at the same time, economically motivates Third-Party Applications as well as the Service Consumers themselves, to detect and report violations of Consent Policies. Beside these both advantages, Service Providers can effectively prove their compliance to legal obligations derived from the GDPR to supervisory authorities.

Our system supports Service Providers to fulfill the following three particular obligations: First, using our a system enables Service Providers to establish an auditable archive of Consent Policies. Supervisory authorities can effortlessly review Consent Policies of Service Consumers at any time by monitoring the public blockchain. This way, Service Providers satisfy the GDPR requirement formulated in Article 7(1).

Second, the system presents an appropriate technical measure to ensure and to be able to demonstrate the legally valid processing of PII under the GDPR, as required in Article 24(1). By early detecting consent violations, the harm caused to affected Service Consumers can be efficiently mitigated.

Third, the system facilitates the reporting of consent violations to supervisory authorities as required in Article 33. Due to the public verifiability of violations, supervisory authorities can immediately review relevant information about detected violations, including their nature, detection time, and estimated harm of affected users.

Given the nature of modern applications composed of different web services as well as the demand of legally sufficient API protection, our *System Design* and the introduced *Violation Detection* mechanism are major steps toward technically mediated violation detection within GDPR-compliant consent-based access control solutions in multi-service setups.

REFERENCES

[1] European Parliament & Council, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)," *Official Journal of the European Union*, vol. L 119/1, pp. 1–88, Apr. 2016.

[2] Article 29 Data Protection Working Party, "Opinion 15/2011 on the definition of consent," Tech. Rep. 01197/11/EN WP187, Jul. 2011.

[3] ——, "Guidelines on consent under Regulation 2016/679," Tech. Rep. 17/EN WP259 rev.01, 2018.

[4] M.-R. Ulbricht and F. Pallas, "YaPPL - A Lightweight Privacy Preference Language for Legally Sufficient and Automated Consent Provision in IoT Scenarios," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, 2018, pp. 329–344.

[5] J. Eberhardt and S. Tai, "Zokrates - scalable privacy-preserving off-chain computations," in *IEEE International Conference on Blockchain*, 2018.

[6] J.Eberhardt and S.Tai, "On or Off the Blockchain? Insights on Off-Chaining Computation and Data," in *ESOCC 2017: 6th European Conference on Service-Oriented and Cloud Computing*, 2017.

[7] J. Eberhardt and J. Heiss, "Off-chaining models and approaches to off-chain computations," in *Proceedings of the 2nd Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, 2018, pp. 7–12.

[8] D. H. (ed), "Rfc6749 – the oauth 2.0 authorization framework," Available at: https://tools.ietf.org/html/rfc6749, 2012.

[9] E. Y. Chen, Y. Pei, S. Chen, Y. Tian, R. Kotcher, and P. Tague, "Oauth demystified for mobile application developers," in *ACM Conference on Computer and Communications Security*, 2014.

[10] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, 2014.

[11] J. Adler, R. Berryhill, A. Veneris, Z. Poulos, N. Veira, and A. Kastania, "Astraea: A decentralized blockchain oracle," pp. 1145–1152, 2018.

[12] J. Heiss, J. Eberhardt, and S. Tai, "From oracles to trustworthy data on-chaining systems," in *IEEE International Conference on Blockchain*, 2019.

[13] R. Kumaresan and I. Bentov, "How to use bitcoin to incentivize correct computations," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 30–41, 2014.

[14] J. Teutsch and C. Reitwießner, "A scalable verification solution for blockchains," Available at: https://people.cs.uchicago.edu/~teutsch/papers/truebit.pdf, 2017.

[15] S. Matsumoto and R. M. Reischuk, "IKP: turning a PKI around with decentralized automated incentives," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2017, pp. 410–426.

[16] N. B. Truong, K. Sun, G. M. Lee, and Y. Guo, "Gdpr-compliant personal data management: A blockchain-based solution," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1746–1761, 2020.

[17] K. Rantos, G. Drosatos, K. Demertzis, C. Ilioudis, A. Papanikolaou, and A. Kritsas, "Advocate: A consent management platform for personal data processing in the iot using blockchain technology," in *Proceedings of the International Conference on Security for Information Technology and Communications*, 2018.

[18] A. Ouaddah, A. Elkalam, and A. Ouahman, "Fairaccess: a new blockchain-based access control framework for the internet of things," *Security and Communication Networks*, vol. 9, p. 5943–5964, 2016.

[19] S. Cha, T. Tsai, W. Peng, T. Huang, and T. Hsu, "Privacy-aware and blockchain connected gateways for users to access legacy iot devices," in *Proceedings of the IEEE 6th Global Conference on Consumer Electronics (GCCE)*, 2017, pp. 1–3.