Make users own their data: a decentralized personal data store prototype based on Ethereum and IPFS.

M.Alessi, A.Camillò, E.Giangreco, M.Matera, S.Pino, and D.Storelli
Engineering-Ingegneria informatica S.p.A.
Lecce, Italy
{marco.alessi, enza.giangreco}@eng.it

***Abstract*—In the times we are living, data protection infringements, at local, national or international level, are a daily occurrence, highlighting how important is the problem of users' awareness and "consent" about what data should or not be shared. A vast number of service providers strives to have access to users' personal data. While users may be aware of sharing their data with services they receive, they may be still unaware if their data is passing in others' hands and unknown third parties. But the sharing of personal data remains unavoidable, in this always connected digital era, contextualized services are not only fancy desires, they could save money, time, and even lives. The problem becomes even more complicate if we try to consider the devices around us: how to share devices we own, so that we can receive pervasive services, based on our contexts and device functionalities. The European Authority has provided regulations about personal data protection, but there are still significant differences in the ways each EU member state would implement the protection of privacy and personal data in national laws, policies, and practices. The tool that should empower users with the personal data protection has to face two problems: data privacy and control. Due to the lack of central authorities, blockchain based technologies would seem fit for the challenge, but such solutions are not fully exploited. One possible reason could be that distributed architectures alone do not achieve privacy of data. In this paper we tackle the challenge of a novel Personal Data Store, by making use of a distributed architecture, based on the Ethereum framework, together with an ontology to model user profile and data/device sharing towards services. Such solution, The Decentralized Identity Manager, solves personal data protection by offering a unique endpoint, without any central authority, where users can manage their data/device access, their privacy levels, and grant or deny sharing consent, every time services ask for personal data.**

Keywords—Security and Privacy for Iot; Privacy Challenges; Personal Data Storage; Blockchain application; Profile management

I. INTRODUCTION

It would be impossible to imagine, in this very moment, the number of data which is being created, transferred or shared over the internet. Everyone would probably agree that this number is surely high enough, that an important degree of it would represent "personal data", data which would belong specifically to users, describing sensible information about those users. We can assume that the entity of this scenario is large enough to raise a question: is each one of those users really aware of to whom or what he/she is lending personal data? Breaks in data protection and security are a real problem, while data are shared everywhere at every minute, it becomes more and more unclear what exactly happens to these often personal and sensitive data. Most of the times, the issue is raised by people's fault, as we are all well aware, when users share their information or their pictures on social networks. Such actions often lead to loss of control over personal data, users may not be able to really delete their contents, even if the delete button is present. And even that "delete action" could be observed to learn the clicking behavior, as well as browsing behavior or history, from service providers, in order to sell personal information to advertising companies. Collections of these observed data often lead to highly valuable and huge data profiles, and the subjects of those profiles could have no clue these data exist, processed by some third party that the subject has never even heard of. Directives are already present, in order to ensure privacy and safety of data. The EU Data Protection Directive 95/46/EC1 provides explicit rules on legal grounds for data processing to make sure personal data are protected. In addition, from 25 May 2018 in all EU member states, the newest General Data Protection Regulation (GDPR) will be applied[1]. Yet people have little control over their personal data, once they have shared those information to services they receive or third parties. A Personal Data Management (PDM) is needed, to empower users with effective control over their personal data, a tool which provides insight in the processing of personal data sharing. The concept of a Personal Data Management system is not new, but it has to cope with the arise of the Internet of Things, and the great number of smart devices in our surroundings. Those make it possible to deliver personalized and contextualized services which we call "pervasive". Pervasive services need, in order to deliver their special outcomes, to have additional information about our devices: where to display the outcomes, from what device to retrieve important data needed to deduce contexts. Our contexts are not only limited to the places where we consume services we need, but also involve devices where those services are sent. Following Michael Levin and his 3C Framework[2]: a service must continuously adapt itself in order to be optimally used through the optimal device. For the reasons presented above, we believe that modern personal data store systems have to solve the issue of offering access to personal devices, shared for the purpose to receive the contextualized services. We tackle the problem of personal data management, including personal devices as personal data. The aim of the paper is to provide a prototype for personal data/device management, a Personal Data Store (**PDS**), which takes advantage of distributed technologies, which may cut out central authorities, thus overcoming trust issues, and at the

same time, leveraging those pervasive properties that distributed technologies offer. The rest of the paper is organized as follows. Section II gives insights about the problem addressed. Section III presents the concept of the proposed solution and how the technological platform is realized. In section IV an evaluation of the proposed prototype is given in its basic scenarios developed to test the functionalities. Finally, section V concludes with final outcomes and future work.

II. MOTIVATION

A Personal Data Store (PDS), or called personal data vault or locker, is a service allowing an individual store, manage, and deploy their key personal data in a highly secure and structured way[3][4][5]. Each user has not only control over his/her data: users have ownership, thus they can decide what services may access personal data store, and eventually what kind of data can or cannot be retrieved by those services. A general description of Personal Data Management systems has been given by Bus and Ngyuen[6]: they divide the objectives of a general PDS into three levels: infrastructure, data management, and user interaction.

• The infrastructure level has two main objectives: ensure integrity and confidentiality of data. That includes supporting all appropriate techniques like encryption, logging, monitoring, authentication and identification of protocols. For example, reliability and acceptability of an infrastructure can be verified through mechanisms of marketing; regular certified checks or forms of supervision over part of the infrastructure

• Data management level ensures safe and effective data control including permissions management mechanisms, communication policies, data auditing capabilities, etc. The most common approach to data reliability management is to create a contract between the user and the data controller by "giving responsibility" to the latter and making him/her aware of the fact if the required permissions are ignored.

• User Interaction is defined as "the element that enables end users to have a significant interaction with service providers, regarding permissions and policies associated with the use of their personal data".

The previous issues suggest that each PDS should offer simple and intuitive tools to control context-dependent data sharing, which all rely on trustworthy underlying data management and infrastructure layers. Many Personal Data Stores have been developed in order to solve the issues of personal data management, thus proving the importance of the matter.

Various data sharing approaches are possible: a user could decide to share some raw data, some aggregate data, or just a representative model of behavior. In addition, the data shared could be strictly related to the user or could be anonymized in order to assure privacy. Within the same PDS, users may be eligible not only to share their personal data to a particular subset of users, but also to access other users' data. Large scale development of PDSs and data control systems, is a long discussed subject in literature, and nowadays, more than ever, it is an unsolved and important issue.

We may divide personal data stores into three main categories: centralized, decentralized and hybrid. The centralized PDS often make use of a central authority which is entitled to manage not only the service, but also the trust between users and services, and eventually acting as an intermediary if trust or legal issues arise. On the contrary, decentralized PDSs lack of a central authority, thus they have to implement mechanisms for regulating trust and data exchange. Hybrid approaches are also possible, where the management and trust is divided amongst users and a small number of reliable authorities. In the following section we describe various examples of PDSs which have been analyzed for both categories.

MyDex[6] is a centralized Personal Data Store, managed by the owner company. The architecture is Cloud based and personal information is encrypted by default, so that personal data is not accessible for Mydex managers. The only data visible from the service provider are the metadata which describe data types contained within the profile and the relative permissions. Any profile data may be shared with third parties by signing a contract in which the terms of the sharing are specified. Organizations or services that want to access personal data must sign a contract with Mydex and pay a small fee to access their services. The presence of a contract allows Mydex to act legally in case of violation of the terms. In addition to the data entered explicitly by the user, dynamic users' information may be included in the data profile, such as the history of credit card transactions, phone call history, bank transactions, etc. **IRMA[8]**, literally I Reveal My Attributes, is a service developed by Radboud University. It is based on the idea of data minimization, i.e. the minimum strictly necessary dataset has to be shared with services, in order to make the latter operative. Furthermore, such data must not allow identification of the individual. The goal is therefore not to allow a service to extract the identity of a user avoiding the sharing of highly identifying data (such as telephone number). The profile is stored and encrypted on a smart card which is associated with a private key and a PIN, so that the only owner of the data is the user himself. The insertion of some data must take place through an authority that guarantees the truthfulness of these (e.g. the Registry Office can insert and certify the veracity of the date and place of birth of a user). Given the difficulty in updating data, this solution is not very suitable for the concept of social web profile, where the presence of variable data requires an easily accessible architecture. **OpenPDS[9]**, developed by MIT, it is similar to Mydex, with a focus on user's metadata. **SafeAnswers** is the technique that, applied to OpenPDS, allows to extract semi-anonymized profile data reducing the probability that the user could be identified. These "answers" can also be stored in the OpenPDS and reused by third party services in a later time. To do this, third-party services will have to create an ad-hoc module that processes raw data and generates a higher level response. This module must be installed on the user's PDS and this means that the third-party services will never come into possession of the raw profile data. Unlike the PDSs described above, this does not handle explicit profile data (i.e. explicitly entered by the user) but only observed data (e.g. sensor data, browsing history, etc.). The SafeAnswers modules and their applications will apply inferences and deductions on these data. User data is

encrypted and stored in a single location specified by the user, which can be a server, a cloud server or his/her computer. A good example for showing the important growth of decentralized blockchain-based technologies for Personal Data management in the recent period is **Onename[10]**. It is created by the startup BlockstackLabs and based on Bitcoin blockchain. Onename is first of all an identity management service that exploits the virtual Bitcoin registry to store user profiles ensuring security, privacy and data decentralization. The profile data is stored on a p2p network, encrypted or not according to the user's needs, and associated in a non-corruptible way to the identity registered on the blockchain. Being a recent application, it is constantly evolving and promises to add more features regarding sharing profile data with third parties. Similar to onename, **BITNATION[11]**is a decentralized, open-source system, powered by the ethereum technology, in an attempt to foster a peer-to-peer voluntary governance system, rather than the current 'top-down', 'one-size-fits-all' model, restrained by the current nation-state-engineered geographical apartheid, where quality of life is defined by where users are born.

In this paper we present the experimentation on Personal Data Stores which has been carried out in the Servify project. Servify is the first project of SI-Lab for the development of skills and technological tools in User – Driven ICT – based Service Innovation[12]. Such research took us to develop the current Decentralized Identity Manager(**DIM**). The architecture of the component is entirely based on the concept of decentralization, and it is based on Ethereum[13] IPFS[14].

III. THE PROPOSED SOLUTION

The Decentralized Identity Manager component is developed in order to manage a user's identity and profile. The most important feature of the component is to provide user profile data, both static and dynamic, for the contextual personalization of pervasive services. On the one hand, users can use the component to create, edit and delete profile information, indended as personal data or devices; on the other hand, the services take advantage of information within this component, on user's permission, to provide a better contextualized experience. Decentralized because the first design purpose of the component is to let the users own their data, without considering a centralized authority or third-party services that the users have to trust. In recent years, centralized organizations, public and private, accumulate large amounts of sensitive and personal information, sometimes this situation is cause for tremendous scandals[15]. With this in mind, we have intended to implement the user identity and its management with the main concepts of privacy-by-design and decentralization.

The user profile, on which the current component is based, derives from a multidimensional model. In order to conceptually represent the user model, it was decided to adopt schema.org[16] model. This choice is motivated by the fact that the ontology in question is one of the most widespread at this time. Among the ontology representation formats studied, it was decided to use the JSON-LD format, a choice conditioned by the fact that user information must also be exchanged between web services. Using this format, information is expressed more clearly and concisely and can also be human readable, which cannot be said for other markup languages. In addition there are numerous tools that allow to quickly verify the semantic correctness of the file.

A. *The user profile schema*

As stated earlier, the user profile model is derived by schema.org, the main schema.org entities considered are: *Person*, the starting point to describe the user and all its dimensions; *Product*, representation of the devices owned by the user; *Place* representation of places; *Action*, actions performed by a user (e.g. travel or driving).

Such entities are the starting point to describe the user context. Since there is no specific schema.org entity that allows describing the context, it was decided to provide context information by using the *Person*, *Action*, *Place*, *Product* entities respectively. Action identifies a user's action, referenced through a relationship with the Person entity (for example, moving from one place to another or physical activity in a specific time frame). This action is characterized by one or more execution places, described by the Place entity, and may involve one or more devices, described by the Product entity. Even in the case of a user's interests, schema.org does not provide supporting entities and attributes. A new interests attribute of the Person entity is therefore defined and a new InterestTag entity, derived from the Thing entity, which includes two attributes: name and weight. The first identifies the name of the object of interest to the user. The second attribute identifies a weight, i.e. how much the specific interest is strong for the user. A partial schematization of the proposed model is shown in Fig. 1. The entities and attributes in green are the extensions proposed to schema.org.

The Context attribute specifies the chosen vocabulary. In the specific case this is equivalent to http://schema.org. This information allows services and applications that receive such a file, to identify the ontology used for the description of the entity. The structure is divided into two **public** and **private** blocks, useful for the service that will implement the user identity to manage the privacy of information. In the public section are present all the attributes with a level of public privacy. In the private section, there is the list of all attributes with a private privacy level. The level of privacy is of fundamental importance because it defines whether to make information visible to external services (public) or visible only to the owner (private). The attributes in the private section will be encrypted and visible only to the user who owns the profile. The *shares* attribute, not present in schema.org and introduced in this model, indicates all the applications with which the user has decided to share their profile information, thus making the model ready to build up the history of services to which data have been shared. The application name is specified thanks to the *service* attribute. The *owns* attribute is suitable to include the devices of a user (eg smartphones, computers, tablets) coded with the Product type. This entity includes information about owned devices (device APIs, unique IDs, etc...), useful to optimally address devices by services. *homeLocation* and *workLocation* respectively identify the geographical location of the user's home and work site.

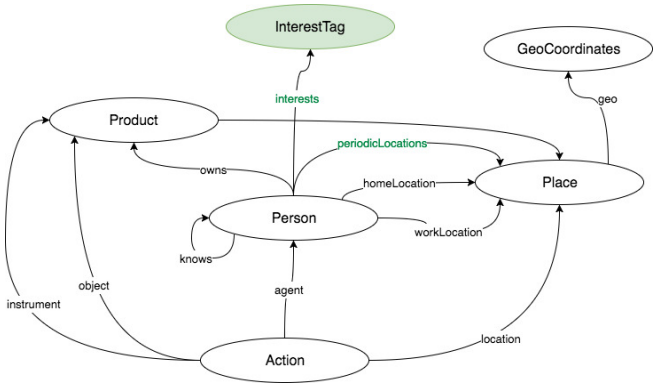

Fig. 1 Ontology schema defined for Decentralized Identity Manager

A new attribute has been introduced, *periodicLocations*, which has the purpose to describe the places frequently visited by the user (e.g. gym, supermarket, etc.). The social relations of the user are described by the attribute *knows*, and their preferences, as previously stated, by the *interests* attribute.

B. *Technological Requirements*

The technological solution is designed to take advantage of distributed technologies like Ethereum and IPFS. Let's see such technologies in detail.

Ethereum[13] is a decentralized platform for the execution of smartcontracts and Distributed Applications, which are run without any possibility of censorship, fraud, or interference from third parties. Ethereum is not only a platform, but also a Turing complete programming language, running on a blockchain, which helps the developer to build and publish distributed applications. To understand the Ethereum technology, we need to deepen the concepts of blockchain and smartcontract[18]. A smartcontract is a contract that is able to go into execution and enforce its clauses without external intervention. Unlike a traditional contract, a "smartcontract" is written in a language that can be executed by a computer. As a contract on paper, it can provide for obligations, benefits and penalties that are borne or for the benefit of the contracting parties in different circumstances. However, it can also receive information as input, process it on the basis of defined rules and perform actions as output. The blockchain is made of blocks that store blocks of transactions, correlated by a timestamp. Each block includes the hash of the previous block forming a chain, where each additional block reinforces the previous ones. This is why every Ethereum node, before being able to write on the blockchain, must first download all the blocks already present on the blockchain, to be synchronized with the last transaction performed. The blockchain can therefore be considered, like a ledger, replicated on a network of equal nodes, where the proprietary information of one or more assets is recorded indelibly. The immutability of the records is guaranteed by cryptographic technologies and by the fact that, a change, to be accepted (for example a transfer of ownership of an asset), must be confirmed by the majority of nodes in the network. Writing on the Ethereum blockchain involves a cost in terms of ether (the cryptocurrency of Ethereum).

Fig. 2 Components of the Decentralized Identity Manager

For this reason, writing thousands of transactions on the blockchain can be very expensive.

Distributed storage refers to a network where data is stored on different nodes, which belong to the same storage system. The nodes are "peers" between them, i.e. there is no one more important than others, and the information is replicated, so as to be always available.

IPFS (InterPlanetary File System) is a distributed storage that aims at building a global file system and p2p. In IPFS, when a file is added to the network, it becomes accessible to all nodes via a hash. The hash is unique, and it gets modified every time the file changes.

C. *Deployment of the Solution*

As mentioned earlier, writing on the blockchain involves a cost in terms of ether. The greater the amount of information that needs to be saved, the greater the number of ether required. Hence the need to use IPFS as a "distributed storage", where to save the profile information of each user. In this way, the information that will be saved on the blockchain will be relatively minimal (low impact on the monetary ether cost), Only a limited amount of information is stored on Ethereum for each user: Username, Ethereum address, Hash of the profile file. The username is the unique username in the whole system chosen by the user during registration. This username is associated with an Ethereum (personal) address also generated when the Decentralized Identity Manager is used for the first time. The hash of the profile file, is a string generated dynamically by IPFS that allows to reach the file and access the information content. Username, Ethereum address and hash of the profile file are stored in a smartcontract that has the role of "identity log" of end-users. For prototype validation, a desktop and a smartphone version of DIM have been created. The first architecture involves the use of any user PC, on which the Ethereum blockchain and the IPFS daemon have to be executed. In this case, the user interface of the application is accessible via a browser. Ultimately, the PC becomes an IPFS node, with the ability to read or write user profile information. The second architecture, seen as an evolution of the first, is designed as a mobile application, able to read or modify the user profile, integrate the user's personal data with the data coming from the most important social networks and share personal information with external services which make explicit request.

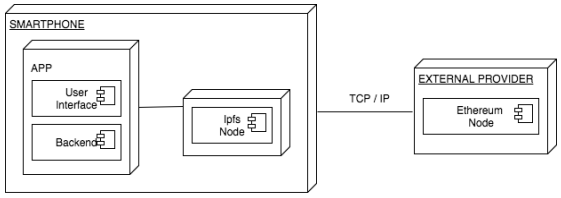

Fig. 3 Smartphone deployment of the DIM prototype

This type of architecture involves the use of an external Ethereum public node for reading and writing on the blockchain and a local node that connects to the IPFS network. The deployment diagram in Fig. 3 describes how the mobile application works. In this context, the user interface and the backend service are all included in the mobile application. The choice of Android operating system has been made for many reasons: its large use, and because there are libraries allowing to use both IPFS and Ethereum. Android IpfsDroid[19] library is meant to store and share decentralized files, and the Ethereum Lightwallet library[20] allows to send transactions to a remote node, and create an ethereum account by providing a passphrase.

A Lightwallet library and an IPFS daemon are installed on each device. The Lightwallet library communicate with an Ethereum node which can be self-hosted by the user or provided by third party entities. This brings numerous advantages, above all in terms of performance, given that the device does not have to manage the memory required for the download of all the blocks which make up the blockchain. Encryption works on two levels: the first is at Ethereum level, where the users take advantage of a pair of asymmetric keys which identifies an Ethereum account, the second is at IPFS level with one symmetric key in order to encrypt profile data. Ethereum private key give authorization in modifying information registered on the Ethereum smart contract. Each time the profile file is modified and stored on IPFS, a new IPFS hash related to the profile is generated and then saved on the smart contract. Private profile data are encrypted on the same file with the symmetric key.

A native Android app has been realized. Here the user can delete the entire profile, add/edit information, or delete a subset of it. In the homepage of the profile application there are three buttons that allow to: share personal information with external services, integrate personal data with information from Facebook and Twitter social networks and finally a QR code scan function to manage user authentication from external applications. In the main interface, it is possible to enter the profile management section, by selecting the "About me" button. In this section, for each information field, using the button next to the textbox, users can specify the privacy level of the attribute that can be public or private. If the user decides to make certain information private, the latter are saved in the JSON file and encrypted using a personal symmetric encryption key. The user can decide which information import from Facebook and Twitter, by selecting the respective checkbox along with the privacy level (private/public). It is possible to choose new data, that users desire to import from social networks, by clicking the "Integrate" button. This modifies the JSON file associated with that particular profile, entering the new selected information.

IV. VALIDATION

The validation phase aims at proving the main functionalities of the prototype: (i) explicit personal data insertion within the profile, (ii) intrinsic personal data extraction from social networks towards the profile, and (iii) personal data sharing with a requesting service.

- ***Explicit personal data insertion*** is carried out, by making users interact with a specific interface exposed by the component: the user fills the profile with his/her personal data (such as gender, name, age, devices owned, interests, etc...).
- ***Personal data sharing*** takes place when a service sends a requests for personal data to the decentralized identity manager: the user will receive such request notification and then he/she can accept or refuse the data sharing. If the data sharing is accepted, the user application will send requested data to the service. The insertion of personal data in the profile file can optionally be performed on a second moment, after service request. In fact, when a service requests certain missing data in order to provide its outcomes, the user receive not only sharing request, but he/she will be also asked to insert such missing information.
- ***Implicit personal data extraction*** is carried out in order to automatically retrieve personal data. Such personal data can consist in: location extracted from GPS device position, prototype usage (obtained by observing the user's interactions with the services, with other users or with devices.) or from an interest mining component to which the prototype interfaces. Thanks to the Interest Mining component, the users' interests and preferences are processed automatically from those contents users release on social networks (Facebook or Twitter).

In **Fig. 4** the profile sharing information section is illustrated, with an external service, which makes explicit request for personal data. In particular, the user is obliged to provide the mandatory attributes requested by the service, but may decide not to share some optional attributes, going to select/deselect the respective checkboxes. When that user accepts to share his/her - at least mandatory - personal data, then a special URL (provided by the requesting service) is invoked. Towards this URL, the DIM will provide the required data to the service. **Fig. 5** shows the user's interests extracted from Twitter through an "Interests Mining" procedure developed in a dedicated module.

In order to prove the usability and effectiveness of the solution the *mobile prototype* (see Deployment of the Solution) has been installed, together with the IPFS daemon, on a Android Smartphone. As previously stated, the mobile prototype is equipped with a Lightwallet, which is interfaced with an Ethereum node on a remote virtual machine, reachable through http calls., Two scenarios have been developed: a basic ***identification*** scenario meant to give access to the SERVIFY service eco-system, a ***"device sharing"*** scenario, where a

service requests user's devices in order to offer its outcomes following a pervasive metaphor.

The identification scenario exploits the solution in order to give users' devices access to services registered in the SERVIFY service eco-system: the service asks the user's DIM to share his/her unique ID within the eco-system.

In order to test the solution and its data sharing capabilities, it has been developed a ***"device sharing"*** scenario, where a service requests access to user's devices in order to offer its outcomes following a pervasive metaphor. The services which have been identified in order to take advantage of this scenario are:

Hotel room booking. The user has previously booked a hotel room. Then when the same user arrives close to the booked hotel room, the system checks if the user has the rights to access such hotel room. In order to do so, the service asks for the devices owned by the user and then, if a correct device - owned by the correct user - is close (via bluetooth) to the door, that door gets unlocked.

Continuous chat. The user is using a chat service, then the user moves from one device (pc desktop) to a smartphone device. The chat service shifts from one device to the other, requesting access to user's devices and then redirecting chat history and control to the active device (smartphone). The user now is able to continue his/her chat session on the smartphone.

In all the aforementioned scenarios (identification and device sharing) the services ask for user's personal information: in the identification scenario, it is required to share personal ID, in the device sharing (both hotel room booking and Continous chat), it is required to share device information (see The user profile schema). The requests for personal data is made at the same manner, with a special code, generated each time the service makes request. the service have to include a special Javascript lib: **servify_qrcode.js**. The service takes advantage of the library by generating a QR code with the following information taken as input: (i) the attributes that the service requires from the user (mandatory and optionals), (ii) the URI where the service needs to receive the corresponding data. The resulting QR code is shared with the user's DIM: the user has to frame such QR code with the QR code scanning functionality of the prototype. In this manner the DIM is able to decode required personal information requested and URI to which send such information.

Both identification and device sharing scenarios have been tested in a laboratory settings, and those will be more investigated during the SERVIFY project testbed, held by research professionals belonging to the project. During the testbed, users related to the academic and civic world will be involved as testers and will be asked to exploit the developed solution, give their feedback to assess usability and usefulness of each prototype and the project as a whole.

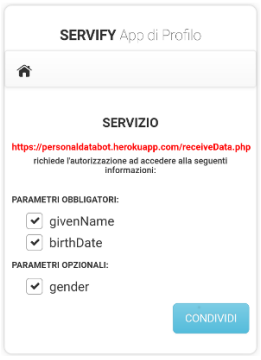

Fig. 4 Sharing Data with external services

Fig. 5 Interests Section

V. Conclusions and Future Work

In this paper we have presented a particular deployment of a Personal Data Store, in the sense of a technological framework which aims at solving the problem of personal data storing, protection and privacy.. The solution has extended the concept of PDS in different ways: using a dedicated ontology and a special distributed architecture. It takes advantage of a special profile schema, specially deployed to model personal information, made up not only by data about the user, by also by devices owned by him/her. In this way we provide a solution, ready to face the challenges of IoT era: such as device sharing. The prototype has also been developed making use of distributed technologies: IPFS and Ethereum, both assuring the absence of centralized authorities, thus avoiding the presence of a central entity, who could illegally exploit personal information. This choice has been vital in order to assure the ownership of personal data only to users who legally own them. Privacy has been enhanced by taking advantage of two different pairs of keys: a special public/private encryption keys offered by Ethereum platform, plus a symmetric key to encrypt IPFS profile, when users decide to set private some of their data. In particular it has been proposed an Android prototype to be tested during the validation phase. More functionalities of the prototype are:

• retrieve information from their Facebook or Twitter social profiles;

• share information with Servify included in the Servify eco-system. Then users are able to accept or deny sharing authorization for mandatory/optional data;

• view a tag cloud of their interests based on an Interest Mining process.

The validation of the solution has been carried out in a laboratory settings, and it has involved the use of the prototype together with services of the Servify eco-system. Such services make request for personal data and devices.

In order to validate the effectiveness of the solution, two main scenarios have been considered. The first one is the

identification scenario, where the user gets identified within the SERVIFY eco-system, sharing with the service his/her unique ID. The second scenario is the device sharing, where the user shares his/her devices in order to receive a pervasive service. While the validation has been carried out in a laboratory settings, we plan to test the prototype in a larger real settings during the SERVIFY project testbed, held by research professionals belonging to the project, with the support of users related to the academic and civic world.

We envision many possible future works, due to the vital important of data protection. The first important prototype evolution is the possibility to make it work in a bigger scenario, external to the Servify ecosystem, thus allowing interfacing the prototype to more third-party services in a real setting. For the same reason we envision the importance of the transition from Ethereum testnet (the blockchain we use at the moment) towards Ethereum mainnet.

Other possible future features involve:

• The possibility to manage one's own Ethereum identity through a user friendly interface, directly from the mobile application.

• the creation of a list of safe user's addresses connected with the distributed profile, where users can insert and expose different layers of personal data towards services.

An issue which is left to be solved in a future release, is the Ethereum dependence on RSA cryptography with asymmetric keying. This introduces a vulnerability problem due to the fact that a quantum computer, thanks to the Shor factoring algorithm[21] can be used to violate key encryption. To solve the aforementioned problem, a group of Guardtime scientists[22] have realized the KSI Technology Stack standard able to solve the aforementioned problem and, consequently, our prototype could be made safer by adopting it.

A major issue we envision, is to address the compatibility between blockchain technology and the European General Data Protection Regulation (GDPR) which will become a law in the Spring of 2018. The main difficulty is about the “right to erasure”, which means that if an individual no longer wishes for his data to be processed, this could ask the subject controlling his/her data to erase it from the blockchain. This is, at this moment, impossible because of how the blockchain works. In fact, transaction records stored in the blockchain cannot be changed or deleted after the first insertion. Should the meaning of the term “erasure” be considered differently in the blockchain context? We will explore solutions and try to provide a way to solve this incompatibility without compromising the benefits of the technology

REFERENCES

[1] https://www.agendadigitale.eu/cittadinanza-digitale/gdpr-tutto-cio-che-ce-da-sapere-per-essere-preparati/ Accessed 19 March 2018

[2] M. Levin, "Designing Multi Device Experiences". O'Reilly Media. February 2014

[3] M. Mun, S. Hao, N. Mishra et al., “Personal data vaults: a locus of control for personal data streams,” in *Proceeding* of the 6th International Conference on Emerging Networking Experiments and Technologies (Co-NEXT '10), New York, NY, USA, December 2010.

[4] T. Kirkham, S. Ravet, S.Winfield, and S. Kellom¨aki, “A personal data store for an Internet of Subjects,” in *Proceedings of the* International Conference on Information Society, i-Society 2011, pp. 92–97, June 2011.

[5] I. Drago, M. Mellia, M. M. Munaf ̀o, A. Sperotto, R. Sadre, and A. Pras, “Inside dropbox: understanding personal cloud storage services,” in Proceedings of the ACM Internet Measurement Conference (IMC '12), pp 481-494, Novembrer 2012.

[6] Bus, J. and Nguyen, M.-H. C. Personal data management a structured discussion. pages 270-288. 2013.

[7] https://mydex.org/ Accessed 19 March 2018

[8] https://www.irmacard.org/ Accessed 19 March 2018

[9] http://openpds.media.mit.edu/ Accessed 19 March 2018

[10] http://onename.com Accessed 19 March 2018

[11] https://bitnation.co/join-the-team/ Accessed 19 March 2018

[12] http://www.silab-sicilia.it/ Accessed 19 March 2018 Accessed 19 March 2018

[13] http://ethereum-project.org Accessed 19 March 2018

[14] http://ipfs.io/ Accessed 19 March 2018

[15] https://www.theguardian.com/world/2013/jun/08/nsa-prism-server-collection-facebook-google Accessed 19 March 2018

[16] https://schema.org/docs/schemas.html Accessed 19 March 2018

[17] https://www.ethereum.org/ Accessed 19 March 2018

[18] https://github.com/ethereum/go-ethereum/wiki Accessed 19 March 2018

[19] https://github.com/ligi/IPFSDroid Accessed 19 March 2018

[20] http://lightwallet.io Accessed 19 March 2018

[21] Post-Quantum Cryptography, DJ Bernstein, J Buchmann, and E Dahmen, eds. Springer Berlin Heidelberg, 2009. doi:10.1007/978-3-540-88702-7

[22] https://guardtime.com/technology Accessed 19 March 2018