Problem Domain Analysis of IoT-Driven Secure Data Markets

Máté Horváth(✉) and Levente Buttyán

Department of Networked Systems and Services,
Laboratory of Cryptography and Systems Security (CrySyS),
Budapest University of Technology and Economics, Budapest, Hungary
{mhorvath,buttyan}@crysys.hu

Abstract. The Internet of Things (IoT) provides us with a vast amount of new data day by day, however, currently, most of these are only stored without utilizing their full potential. The attractive concept of data markets can change this situation in the near future and thus we initiate the study of security aspects of such systems. In this work, as a first step, we analyse the data markets based on the possible security requirements of the different participants. We identify more than 30 possible scenarios and connect these to the relevant areas of cryptography. Our analysis also highlights several open problems motivating further research on certain cryptographic primitives.

Keywords: Cybersecurity · IoT · Data markets · Market mechanisms

1 Introduction

Current technological trends, as the proliferation of smart devices and the Internet of Things (IoT), made the rapidly increasing amount of data a new raw material waiting for utilization. The main barrier of this is that in most cases the collected data is only available for the user and manufacturer of a sensor or smart device. One possible way of exploiting the full potential of this information is to build an ecosystem around it. This is exactly the idea of Data Markets [5], where data brokers (DB) buy data from the owners and resell the collected data (possibly together with computing resources) to a third party that provides some value-added services (VAS) to its users. These services can typically help predictions or support optimization via analysing a wide range of data.

This work was partially performed in the frame of the FIEK_16-1-2016-0007 project, implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the FIEK_16 funding scheme, and it was also partially supported by the National Research, Development and Innovation Office – NKFIH, under grant contract no. 116675 (K).

E. Gelenbe et al. (Eds.): Euro-CYBERSEC 2018, CCIS 821, pp. 57–67, 2018.
https://doi.org/10.1007/978-3-319-95189-8_6

Our Contribution. The issue of security and privacy arises naturally whenever data get out of the control of its owner. In this work, we investigate the possible security issues related to data markets. Compared to a simple cloud computing scenario, where only two roles (user and cloud service provider) are present, data markets involve parties of three type (data owner, DB, and VAS provider) which can all trust or distrust the others. Based on the level of trust between the parties, we identify 31 different scenarios, some of which can be handled straightforwardly, some are closely related to different areas of cryptography, while others motivate further research on specific problems. We found that the two most interesting open questions are the following. *Is it possible to provide the secrecy of a computation and at the same time verify that it obeys certain restrictions? Can an encryption scheme allow for computation on certain hidden data but not on others such that the admissible data can be determined in a fine-grained manner?* Our work also motivates the adjustment of existing cryptographic primitives to the use-cases provided by data markets.

Related Works. While to the best of our knowledge, the concept of data markets has not been implemented yet, several pioneering projects are approaching towards this goal. Their common property is that only limited information is available about their design (in the form of white papers). These initiatives include the Data Market Austria Project [3]; the pilot study of IOTA Data Marketplace [9]; Ocean Protocol [12] that aims to connect data providers and consumers; the decentralized marketplace of Datum [4]; and the Enigma Protocol [15] based on secure multiparty computation (MPC). Market formation through automatic bidding [6,7] can also be considered in the context of data markets.

2 System Model for Data Markets

The resources of a data market are provided by owners of sensors and "IoT" devices (DO), who provide (sell) data directly to the data broker (DB) and indirectly to VAS providers. In our model, DB is not decentralized but rather a single entity, investing in infrastructure for data storage and executing computations. Naturally, different DBs might compete with each other in the future and most probably also with different distributed systems (improving the service quality), but our focus is the inner working of a single, centralized marketplace (with one DB).

DBs offer their services to different value-added service providers (VASPs) that can utilize data in order to fulfil the various needs of end users (either individuals or companies). It is important to note that we do not want to restrict the scope of offered services, but from statistical analysis to the training of AI (artificial intelligence) models, any service is included. Even if the final goal of such ecosystems is to serve the end users, they are less important in our study as we are interested in the security aspects of data markets. More precisely, all the available information for an end user is a subset of the information handled

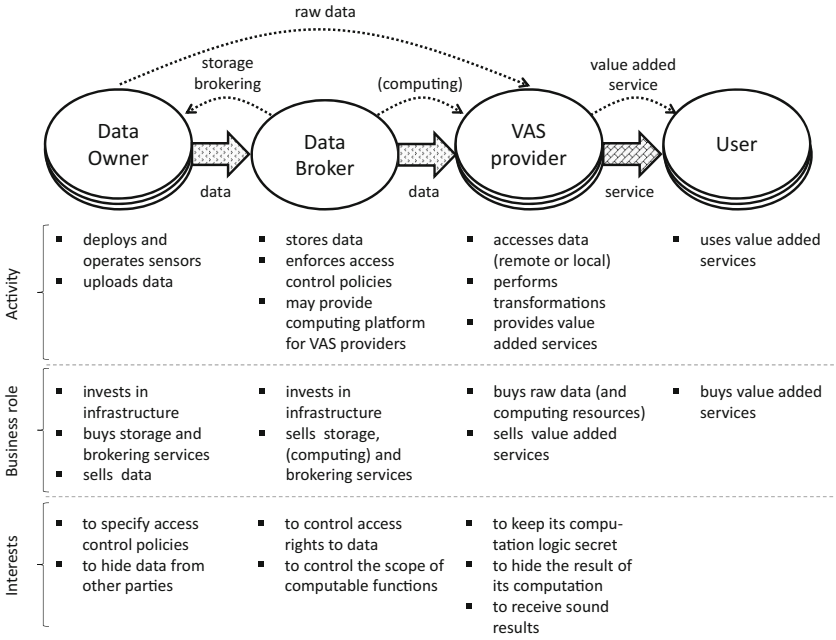

Fig. 1. System model of the envisioned data market.

by the corresponding VASP and vice versa, any information about the end user is exposed in the system through the VASP so for our purposes it is enough to consider VASPs.

The imagined model of the ecosystem is depicted in Fig. 1, containing also the rough business models of the different parties. At the same time, the economic aspects of such system are out of the scope of this work.

3 Related Tools

In this part, we provide a brief and informal description of the concepts appearing later in this work. These are the following.

Trusted Execution Environment (TEE) refers to an isolated processing environment in which computations can be securely executed irrespective of the rest of the system. In a high level, when considering information leakage, outsourced computation that is executed in TEE (by an untrusted party) is equivalent to local computation and thus in the rest of this work we do not differentiate between these two cases. For more details on TEE, see [14].

Differential Privacy is a mathematical framework for the privacy protection of data used for statistical analysis (see [11]). While in some very specific cases it can be useful for privacy protection also in Data Markets, we will not consider this solution because of two reasons. We do not restrict the scope

of computable functions to statistics (i.e., individual data values may be of interest instead of only cumulated data) and we are interested in enforcing access control policies belonging to data, while differential privacy does not restrict access to data.

Homomorphic Encryption (HE) enables to execute computations on encrypted data that results in an encrypted output. This capability can be used to make outsourced computation secure, by maintaining the secrecy of inputs and outputs (while the applied function is public). Depending on the supported functions, there exist additively, multiplicatively, somewhat and fully homomorphic encryption schemes. While the latter one supports an arbitrary number of multiplications and additions, in somewhat homomorphic schemes the number of computable multiplications is restricted. See details in [1].

Functional Encryption (FE) is a generalization of traditional (either private- or public-key) encryption schemes that integrates function evaluation into decryption. More precisely, so-called functional secret keys can be issued for a function f, that when used in the decryption of a ciphertext corresponding to a value x, results in the value $f(x)$ without leaking any more information about x, than exposed by $f(x)$. For details, we refer to [8].

Attribute-Based Encryption (ABE) is a subtype of FE, that realizes fine-grained access control. Ciphertexts and secret keys are associated with access control policies and "attributes" (or vice versa) and decryption is possible only if the attributes satisfy the policy.

Oblivious Transfer (OT) and Private Information Retrieval (PIR) are dealing with transmission of data between two parties, such that the receiver does not have to reveal the data, she wants to retrieve from a dataset (PIR). We call this OT when the receiver obtains no other information, besides the desired data, from running the protocol. In the sequel, prime is going to denote that the protocol is non-interactive. For more details about OT and PIR, see [13].

Secure Multi-party Computation (MPC) allow n parties to jointly compute an n variate function on their inputs, often through several rounds of interaction, without revealing their inputs to each other. Among those cryptographic primitives that aim to achieve some form of secure computation, realizations of MPC are the closest to practical usability [10].

Obfuscation (Obf.) is a program transformation that preserves the functionality but alters the description and operation such that the inner workings of the program remains hidden at the expense of some efficiency loss. In other words, the inputs and outputs of an obfuscated program are plaintexts but the computed function is hidden. In spite of several breakthrough results in the past years, obfuscation is still an exotic area within cryptography with several open questions (see [8]).

We emphasize that MPC and FHE are about to protect the input(s) of a given public computation. However, as it is possible "to turn computation into

data", with the help of so-called universal circuits[1], these primitives can also be used to conceal the executed computation besides its inputs and outputs (although its overhead is significant). In the sequel, we denote with an asterisk if MPC or FHE is complemented with this tool.

4 Problem Domain Analysis

Our analysis is started with the investigation of the relations between the participants of the system. After identifying their possible goals regarding security (see also Fig. 1), we organize the emerging use-cases and identify the relevant branches of cryptography and also pointing out open problems.

4.1 Trust and Distrust Between the Parties

The main goal of this study is to identify the different scenarios that emerge when some of the actors in a data market does not trust all the other parties. In order to go through all possible cases and identify already solved and still open problems, we investigate the possible forms of trust between data owners, data broker and VAS providers.

DO → DB If the DB is trusted by the DO, then it is straightforward to store the data as a plaintext. In this case, access control policy enforcement can be outsourced to the DB. On the other hand, an untrusted DB should not store its clients' data in the clear but in *encrypted* form together with the corresponding *meta-data in cleartex*. Note that the latter one is necessary for providing the data brokering service. Access control of data becomes more challenging that can be resolved e.g., by solving the key-exchange between DOs and VASPs.

DO → VASP While a trusted VASP may access plaintext data, access control might still be important as most probably only smaller sets of data are sold to the VASP and thus even in this case VASP should have only *conditional access* to data. When the VASP is not even trusted by the DO, it is natural to expect that it has no direct access to any data and it is only allowed to get results of non-trivial computations on some specific data.

DB → DO Trusting DOs from the viewpoint of a DB is equivalent to *trusting the data source* i.e. the DB assumes that the received data is not fake, its source is the claimed sensor and the result of the measurement was not modified. The lack of this confidence can only be remedied partly using algorithmic measures (e.g., by checking the consistency of the data) and thus this type of distrust is out of the scope of this work. At the same time, this problem can be addressed e.g., using pricing based on game theoretic considerations.

[1] A universal circuit can be considered as a programmable function, taking as input a program description (that is going to be executed) and the ordinary input.

DB → VASP As the DB is selling information to the VAS provider, pricing should scale with the amount of provided information that can be measured using two different metrics as well. The first is the available amount of data for the VAS provider that can be controlled using some access control policy for the data. Note that in this regard, the interests of the DO and the DB coincides and thus we assume that either of them enforces a common access policy. The second possible way of measuring the sold information is through the scope of computable functions on the data that is available for a VASP. One natural way of restricting the computing capabilities of the VASP is via providing it with a restricted interface that naturally determines restrictions. However, we are interested in a more general setting, where such limitation is not present, especially because it leaves no room for the interests of the VAS provider (see the next part). Accordingly, we assume that arbitrary function descriptions are forwarded to the DB that are evaluated if they are in an allowed function class (for which the VASP has paid). Alternatively, if the computation of the functions is not outsourced to the DB, it should be solved that the data, sent to the VAS provider, is only useful for the computation of "allowed functions" and cannot be used as input for any other functions.

VASP → DO When purchasing data from DB, the VAS providers naturally rely on the honesty of the DOs, however, the enforcement of honest behaviour is the duty of the DB according to the system model of the data market. Accordingly, this trust relationship is indirect and thus not investigated.

VASP → DB The business model of a VAS provider is built around the functions, that are evaluated on the data, bought from the DB. This highlights the importance of this asset and shows that VAS providers are motivated to hide the computation logic they use even if the computation is executed by the DB as a part of its services. In case of so-called learnable functions, that can be reconstructed merely from input-output pairs, hiding these values is an important pillar of keeping the function private. Moreover, the output alone has also business value as the end user pays for this to the VAS provider. When talking about the input data, it is important to differentiate the data value from the meta-data. If the DB stores plaintext data, VAS providers can only obscure the accessed data if both the accessed values and meta-data remain hidden from the DB. When only encrypted data is available to the DB, typically meta-data can help to feed the proper input to the function of the VASP but hiding both information can be the goal of a deliberate VAS provider.

Depending on which of these four assets are intended to be hidden from the DB, 2^4 different scenarios can occur. For the ease of exposition, we denote a VAS provider's confidentiality goals with 4-tuple (F, I, I', O), where each variable correspond to a binary value, 0 meaning public and 1 denoting confidential. The variables represent the function to be computed (F), its input value(s) (I), meta-data for the input(s) (I') and the output (O) of the computation (e.g., $(1, 0, 0, 0)$ represents that the VASP only wants to hide the computational logic form the DB). Some of the resulting scenarios are contradictory so we ignore them in the rest of the work. These are the following.

- If a function is public then its domain is public as well. Practically, the function description must include the expected input that is described with the help of the meta-data, which then cannot be hidden (ruling out $(0,0,1,0),(0,0,1,1),(0,1,1,0),(0,1,1,1)$).
- Hiding the accessed meta-data from the DB implies that the used data values are hidden as well and accordingly $(1,0,1,0),(1,0,1,1)$ are also meaningless.
- $(0,0,0,1)$ is also contradictory as given a function and its input, the output can be obtained so it cannot be hidden.

In case of outsourced computation, the VASP might also want to verify the soundness of the received output. Variants of publicly verifiable computation can help to enable this (e.g., in case of MPC [15]), however, we assume the DB is only honest-but-curious and not malicious, especially as the DB is motivated to provide reliable service in order to keep its customers.

4.2 On the Used Notations

Having identified the possible requirements of the different parties in the system, we introduce a notation to refer to the possible scenarios that depend on the fulfilled requirements. From the viewpoint of the DOs, four main scenarios, called worlds, can be distinguished based on whether the DB and the VAS provider are trusted or not. Each world can be further subdivided based on the trust between DB and VAS providers. Accordingly, let $\mathcal{S}^{D(V)}_{V(F,I,I',O)}$ denote a scenario, which is specified by the values of the different variables. $D, V \in \{T, U\}$ refer to the DB and the VAS providers and their value shows thether they are considered to be "trusted" (T) or "untrusted" (U) by the DO; $V \in \{0,1\}$ indicates whether the DB intends to verify the functions, computed by VAS providers (1) or not(0); and finally $F, I, I', O \in \{0,1\}$ form the 4-tuple, introduced in the previous part, representing the goals of the VASP. When any of the above parameters are not specified in the notation, we always mean a general scenario in which any of the more specific sub-scenarios can occur (e.g., $\mathcal{S}^T$ denotes all the scenarios where the DB is Trusted).

One might miss from the description of the scenarios the specification of the enforcer of an access policy, which can be done either by a DO or by the DB. We find that this question is rather technology related and the effect of the options are the same, so we do not determine this in the use-cases.

In the subsequent parts, we are going to go through the scenarios, discussing their relevance, the related cryptographic solutions, and the open problems.

4.3 Trusted Data Broker

Our first observation is that whenever the DB is trusted $(\mathcal{S}^T)$, it is straightforward to delegate the enforcement of access control policies to the DB. We investigate the arising use-cases, while keeping this in mind. As a second remark, also note that those scenarios are contradictory, where the inputs (of a function from

Table 1. A rough summary of meaningful subscenarios of $\mathcal{S}^T_U$ with the related concepts and open questions.

		VASP cannot have access to plaintext data				
	function	0	1	1	1	1
	input value	0	0	0	1	1
	input metadata	0	0	0	1	1
	output	0	0	1	0	1
DB stores palaintexts (0)	No restriction on the computed function (0)	Outsourced computation or local computation + FE	Obf. / TEE / MPC*	(Obf. / TEE) + output Enc.	(TEE + OT) or (Obf. + OT') or MPC*	(TEE + OT) or (Obf. + OT') + output Enc.
	Limited function queries (1)	Straightforward verification in both cases	Function verification is challenging when the computed function is hidden			

a VASP) are intended to be hidden, while the meta-data of the same input is accessible for the DB ($\mathcal{S}^T_{(0,1,0,0)}, \mathcal{S}^T_{(0,1,0,1)}, \mathcal{S}^T_{(1,1,0,0)}$ and $\mathcal{S}^T_{(1,1,0,1)}$). The reason is that a trusted DB stores data as plaintext and thus the meta-data of the input directly reveal the input value.

Trusted VAS Provider ($\mathcal{S}^T_T$). When DOs trust both the DB and VASPs, the occurring problems are almost trivial to solve. $\mathcal{S}^T_{T(0,0,0,0)}$ corresponds to a scenario where the DB provides value-added services (DB = VASP) and it is trusted by the DO so challenges do not arise. All the other use-cases can be handled if the necessary computations are executed locally by the VASPs and for this plaintext data is accessible. $\mathcal{S}^T_{T(1,0,0,1)}$ represents exactly this case, while in $\mathcal{S}^T_{T(1,1,1,0)}$, $\mathcal{S}^T_{T(1,1,1,1)}$ the application of PIR can obscure the used inputs from the DB. $\mathcal{S}^T_{T(1,0,0,0)}$ and $\mathcal{S}^T_{T(1,1,1,0)}$ represents such situations where the VASP publishes the computed results (e.g., indirectly by buying specific stocks).

Untrusted VAS Provider ($\mathcal{S}^T_U$). The use-cases that can emerge in the world of trusted DBs and untrusted VASPs are summarized in Table 1. Outsourcing the computation to the DB simply solves $\mathcal{S}^T_{U(0,0,0,0)}$ even if function verification is required, as it is accessible to the DB. Local computation is also possible when DB encrypts the required data using FE and provides the VASP with functional secret keys for the required computation. In order to hide the computation logic, either the approach of TEE can be applied or techniques for obfuscation are necessary. Besides these direct solutions, the usage of MPC together with universal functions is also viable. At the same time, hiding the function endangers its verifiability. Indeed, the compatibility of these properties is a challenging open problem. When further information is intended to be concealed, besides the function, OT or the encryption of the output has to be integrated with a solution that hides the function.

4.4 Untrusted Data Broker

Moving along to the case of untrusted DB, access control to the data is not trivial anymore. The fact that the DB has only access to ciphertexts makes those scenarios meaningless in which the input values of the computed functions are revealed to the DB ($\mathcal{S}^U_{(0,0,0,0)}, \mathcal{S}^U_{(1,0,0,0)}$ and $\mathcal{S}^U_{(1,0,0,1)}$).

Table 2. A rough summary of meaningful subscenarios of $\mathcal{S}_U^U$ with the related concepts and open questions. (The computation, executed in TEE after finishing the OT protocol, is indicated in square brackets.)

		VASP cannot have access to plaintext data only to the results of computations					
	function	0	0	1	1	1	1
	input value	1	1	1	1	1	1
	input metadata	0	0	0	0	1	1
	output	0	1	0	1	0	1
DB owns ciphertexts (1)	No restriction on the computed function (0)	Function revealing FE	HE	Function hiding FE	Locally computed function hiding FE or FHE*	OT + TEE[Decr-Eval]	OT + TEE[Decr-Eval-Enc]or FHE*
	Limited function queries (1)	The verification of public functions is straightforward		Using the above function hiding FE, DO can verify functions		Open question	

Trusted VAS Provider ($\mathcal{S}_T^U$). According to the trust between DO and VASP, the latter one can have access to plaintext data e.g., by using public key encryption (e.g., ABE if fine-grained access control is required). In this case, the DB only stores and sells data but computation is executed locally by the VASPs. Therefore all the scenarios in $\mathcal{S}_T^{U(1)}$ are impossible because VASPs are allowed to access the plaintext data preventing the control of computable functions. Local computation also makes $\mathcal{S}_{T(0,1,0,0)}^{U(0)}$ and $\mathcal{S}_{T(0,1,0,1)}^{U(0)}$ unrealistic while the remaining scenarios (especially $\mathcal{S}_{T(1,1,0,1)}^{U(0)}$) are trivially solved by the separation of DB from the computation (in $\mathcal{S}_{T(1,1,1,0)}^{U(0)}, \mathcal{S}_{T(1,1,1,1)}^{U(0)}$ PIR can hide the accessed metadata from the DB). $\mathcal{S}_{T(1,1,0,0)}^{U(0)}$ and $\mathcal{S}_{T(1,1,1,0)}^{U(0)}$ again represents that the services of VASPs reveal their functions output to the public.

Untrusted VAS Provider ($\mathcal{S}_U^U$). This is the most restricted scenario, where DOs are the only parties having access to (their own) plaintext data values. For a concise summary, see Table 2. $\mathcal{S}_{U(0,1,0,0)}^{U}$ and $\mathcal{S}_{U(0,1,0,1)}^{U}$ exactly match the problems that are considered by FE and (F)HE respectively and as the functions of VASPs are not concealed even their verification is straightforward. When the function is intended to kept secret as well ($\mathcal{S}_{U(1,1,0,0)}^{U}$ and $\mathcal{S}_{U(1,1,0,1)}^{U}$), a special form of FE, called function hiding FE, can be invoked either with decryption by DB or local decryption (actually computation) by the VASP after having received functional keys (from DO) and ciphertexts (from DB). However, in these cases function verification is possible, unfortunately, the verifier is not the DB but the issuer of functional keys. $\mathcal{S}_{U(1,1,0,1)}^{U(0)}$ can also be handled using FHE* (i.e., homomorphically evaluating a universal function). The strictest requirements can be satisfied by relying on TEE and OT. After running the OT protocol between DB and the TEE the resulting ciphertext is decrypted, the function can be evaluated on it and either the output is returned (in $\mathcal{S}_{U(1,1,1,0)}^{U(0)}$) or its encrypted form (in $\mathcal{S}_{U(1,1,1,1)}^{U(0)}$).

These use-cases highlight that the integration of computation on hidden data and fine-grained access control is an important challenge as in complex scenarios like these the two goals can arise together. While there are attempts to realize Attribute-Based FHE [2], to the best of our knowledge the integration of FE (for secure function evaluation) and ABE is an entirely open problem.

References

1. Armknecht, F., Boyd, C., Carr, C., Gjøsteen, K., Jäschke, A., Reuter, C.A., Strand, M.: A guide to fully homomorphic encryption. Cryptology ePrint Archive, Report 2015/1192 (2015). http://eprint.iacr.org/2015/1192
2. Brakerski, Z., Cash, D., Tsabary, R., Wee, H.: Targeted homomorphic attribute-based encryption. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 330–360. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_13
3. Data Market Austria Project. https://datamarket.at/en/. Accessed 5 Feb 2018
4. Datum Network - The decentralized data marketplace (White Paper V15). https://datum.org/assets/Datum-WhitePaper.pdf. Accessed 5 Feb 2018
5. Deichmann, J., Heineke, K., Reinbacher, T., Wee, D.: Creating a successful internet of things data marketplace (2016). https://mck.co/2lAIhjk. Accessed 5 Feb 2018
6. Gelenbe, E.: Analysis of single and networked auctions. ACM Trans. Internet Technol. (TOIT) **9**(2), 1–24 (2009)
7. Gelenbe, E., Györfi, L.: Performance of auctions and sealed bids. In: Bradley, J.T. (ed.) EPEW 2009. LNCS, vol. 5652, pp. 30–43. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02924-0_3
8. Horváth, M., Buttyán, L.: The birth of cryptographic obfuscation - A survey. Cryptology ePrint Archive, Report 2015/412 (2015). https://eprint.iacr.org/2015/412
9. IOTA Data Marketplace. https://data.iota.org/. Accessed 5 Feb 2018
10. Lindell, Y., Pinkas, B.: Secure multiparty computation for privacy-preserving data mining. J. Privacy Confidentiality **1**(1), 59–98 (2009)
11. Nissim, K., Steinke, T., Wood, A., Altman, M., Bembenek, A., Bun, M., Gaboardi, M., O'Brien, D., Vadhan, S.: Differential privacy: a primer for a non-technical audience (preliminary version). https://bit.ly/2HnYmXD. Accessed 5 Feb 2018
12. Ocean Protocol Foundation Ltd.: A decentralized data exchange protocol to unlock data for artificial intelligence (technical primer). https://oceanprotocol.com/. Accessed 5 Feb 2018
13. Ostrovsky, R., Skeith, W.E.: A survey of single-database private information retrieval: techniques and applications. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 393–411. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71677-8_26
14. Sabt, M., Achemlal, M., Bouabdallah, A.: Trusted execution environment: what it is, and what it is not. In: 2015 IEEE Trustcom/BigDataSE/ISPA, pp. 57–64. IEEE, Helsinki (2015). https://doi.org/10.1109/Trustcom.2015.357
15. Zyskind, G., Nathan, O., Pentland, A.: Enigma: Decentralized computation platform with guaranteed privacy. CoRR abs/1506.03471 (2015)