Enabling Data Markets using Smart Contracts and Multi-Party Computation

Dumitru Roman and Kien Vu

SINTEF, Norway
dumitru.roman@sintef.no, kienv91@gmail.com

Abstract. With the emergence of data markets, data have become an asset that is used as part of transactions. Current data markets rely on trusted third parties to manage the data, creating single points of failure with possibly disastrous consequences on data privacy and security. The lack of technical solutions to enforce strong privacy and security guarantees leaves the data markets' stakeholders (e.g., buyers and sellers of data) vulnerable when they transact data. Smart Contracts and Multi-Party Computation represent examples of emerging technologies that have the potential to guarantee the desired levels of data privacy and security. In this paper, we propose an architecture for data markets based on Smart Contracts and Multi-Party Computation and present a proof of concept prototype developed to demonstrate the feasibility of the proposed architecture.

Keywords: Multi-Party Computation, Smart Contracts, Data Markets.

1 Introduction

The online behavior of users (e.g., online shopping, interacting on social media) leaves a trail of digital footprints. Nowadays, these digital footprints are in most cases accumulated and traded by various online organizations, offering "free" services to the users. Companies, such as Facebook and Google, are using individuals' personal data to profile users and provide them so called "targeted ads". As a result, they are basically feeding on the users' data to gain tremendous profits, leaving the user without any possibilities to actually monetize their own digital footprints in a transparent data market.

Besides the delicate legal and ethical aspects of exploiting user data (see for example the recent Cambridge Analytica scandal), a major reason for the difficulty of users to monetize their data in a data market is the lack of technical solutions to enforce the desired levels of privacy. When it comes to selling individuals' personal data, one of the main issues is trustworthiness. Generally, individuals do not want to sell their data to a counter-party and have the risk that that party will sell the data further to others for a higher price, or misuse the data for another purpose than the individual initially agreed to.

Two key technologies are emerging as essential to allow such a data market to be established. The first one is *secure multi-party computation* (hereby referred to as

MPC)[1]. MPC is a technology that allows data to be shared in a secret form (i.e., encrypted), while at the same time allowing meaningful computations to be performed on the data, the results of which could be shared with a third-party willing to pay for them. At no point in this process are data exposed in clear form, meaning that no parties involved in sharing the data and computing on the data can have access to the data, other than the owner of the data. By enabling calculation without removing encryption, MPC is the first cryptography-based technology that permits multi-owner data sharing and processing, ensuring that only data owners are allowed to access their data.

The second technology is *smart contracts* (hereby referred to as SC)[2]. A smart contract is a software program that typically runs in a decentralized infrastructure and once started it cannot be changed. A SC can facilitate, verify, and automatically enforce transactions giving it the potential to preserve data provenance in a data market context, where data ownership is essential. By design, a SC is being executed on a Blockchain which is an open distributed ledger that can record transactions between parties efficiently and in a verifiable and permanent way.

This paper explores the combination of MPC and SC technologies in the context of data markets. It proposes an architecture for combining MPC and SC as baseline technologies for data markets, and reports on a successful proof of concept prototype developed to demonstrate the feasibility of the proposed architecture.

The remainder of the paper is organized as follows. Section 2 provides an overview of the proposed data markets architecture combining MPC and SC. Section 3 describes the implemented proof of concept prototype. Finally, Section 4 discusses relevant related works, summarizes the paper, and outlines potential future work.

2 Architecture for Data Markets using Smart Contracts and Multi-Party Computation

Figure 1 shows the proposed architecture for data markets based on combining MPC and SC technologies. The key stakeholders in the market would be companies or individuals that subscribe either as a *buyer* or *seller* depending on the terms of the smart contract.

[1] https://en.wikipedia.org/wiki/Secure_multi-party_computation
[2] https://en.wikipedia.org/wiki/Smart_contract

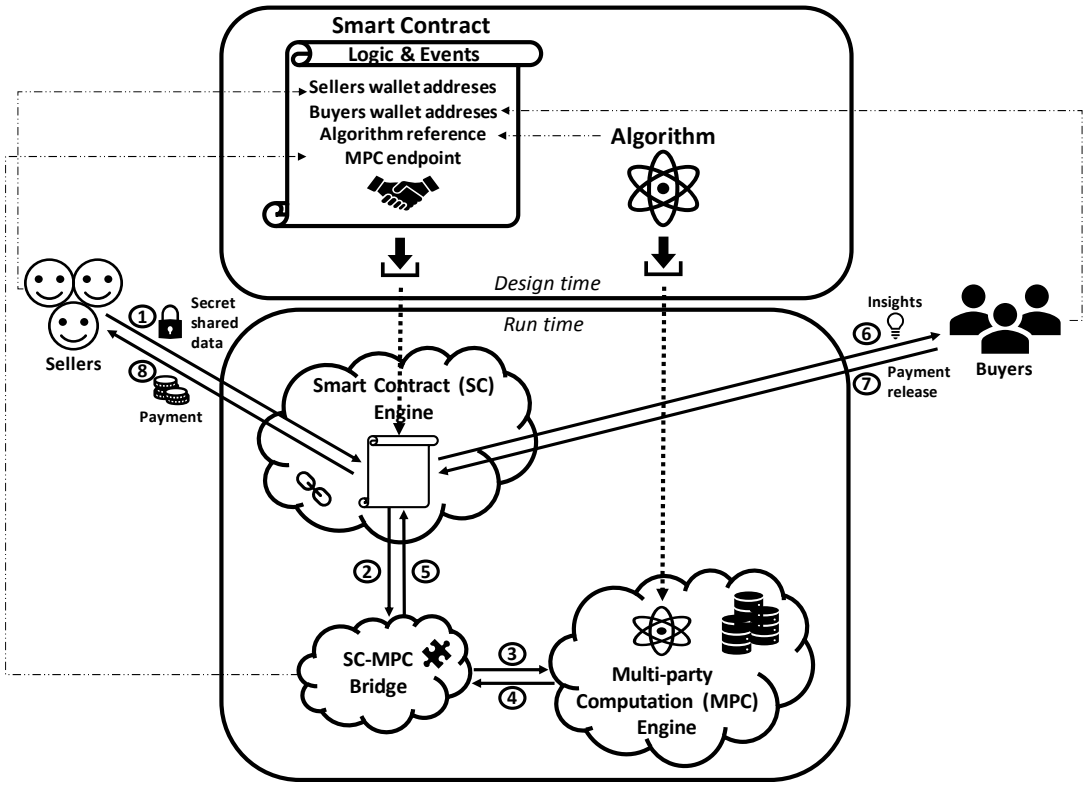

Figure 1. Architecture for data markets using MPC and SC

As depicted in Figure 1, the architecture is divided by design time and run time. The design time infrastructure includes the design of the smart contract and algorithm and their deployment in the run time infrastructure:

- *Smart contract*: A program containing information about the business logic and transactional events of the contract (e.g., how and when data will be exchanged, when and how much the sellers will be paid, etc.), payment information (buyers'/sellers' wallet addresses), the reference to the algorithm that will run on the data, and the address of the MPC Engine that will execute the algorithm.
- *Algorithm:* The MPC algorithm that will run on the (secret shared) data.

The run time infrastructure consists of three components:

- *SC Engine*: The engine where the smart contract will be deployed.
- *SC-MPC Bridge*: A component for ensuring communication between the SC Engine and the MPC Engine. This component is needed since MPC engines are currently not designed to communicate with SC engines.
- *MPC Engine*: The engine where the desired algorithm will be executed on (secret shared) data.

A typical process this architecture would support at runtime is as follows: the data sellers secret share their data as part of the SC execution (1); the SC execution passes the data to the SC-MPC bridge (2) which in turn supplies it to the MPC engine (3), returning the result of the computation to the SC-MPC bridge (4), which is then passed to the SC executing engine (5) triggering the message with the result in form of insights to the buyers (6). Upon receiving of the result, the buyers release the payment (7) and the SC execution enters the final state of executing the payment to the sellers (8).

3 Proof of Concept Prototype

The above architecture was implemented in a proof of concept prototype to test its feasibility. A set of concrete technologies have been selected for the implementation of the architecture as follows.

For the SC engine, Ethereum[3] [1] was chosen. The SC is written in *Solidity*[4] – a high-level, statically typed, and Turing complete programming language. Ethereum, compared to other SC frameworks such as NEM[5] and EOS[6], appears to be the most mature development framework for smart contracts to date.

By design, the SC executed on the Ethereum blockchain is isolated from the outside world. The reason is that Ethereum uses a peer-to-peer network protocol and it uses miners as peers to validate each block on the chain. Hence, each miner must get the same result to reach consensus. In other words, if the SC allows communication with external sources it would lead to the risk of never getting the same result and never achieving consensus. Based on this, a mechanism to allow the communication with the outside world and send/receive events outside of the Solidity SC is needed, in our particular case to receive secret shared data and pass it to the SC-MPC bridge. For this, we chose the Oraclize[7] library. Oraclize is an open-source library that works as a data carrier between the SC and the outside world. Oraclize ensures that data fetched from the data source is genuine and untampered. To guarantee this, Oraclize combines the returned data with a document called authenticity proof. In our case, this proof ensures that the secret shared data has been successfully sent and received by the SC-MPC bridge.

For the MPC engine, the Sharemind MPC[8] framework [2] was chosen. Sharemind, compared to all the other existing MPC frameworks, provides high-level libraries and tools to enable application development by non-cryptographers on a business-process implementation level, making it easier to write MPC algorithms. It provides a programming language called *SecreC* [3] that allows the users to write algorithms to be computed on the secret shared data. All the intermediate and final results are encrypted and only the allowed user making the query can decrypt the result. Sharemind consists of

[3] https://ethereum.org
[4] https://github.com/ethereum/solidity
[5] https://nem.io
[6] https://eos.io
[7] http://www.oraclize.it
[8] https://sharemind.cyber.ee

different nodes that act as independent hosts processing the specified algorithm. The algorithm is created outside the framework, however when it is deployed to the MPC engine, it is the Sharemind hosts that will evaluate and authorize the execution of the algorithm.

The SC-MPC bridge is a component that was needed to connect the Ethereum environment with Sharemind. This component was implemented from scratch since no implementations for this previously existed (Sharemind does not provide REST APIs necessary for Oraclize to communicate with). The component was implemented as a Web service allowing the SC deployed on Ethereum to invoke Sharemind functions. For this particular architecture, Node.js was used to implement the SC-MPC bridge.

Regarding the components for the particular choice of technologies selected for the prototype, what needs to be deployed would be one SC for every algorithm. These SCs need to be deployed onto the main Ethereum blockchain network. The SC-MPC needs to be hosted as a Web service by a third party. The algorithms used to do computation on the secret shared data need to be written in SecreC and deployed to the Sharemind hosts.

4 Relevant Related Works, Discussions, and Outlook

The approach proposed in this paper differentiates from other emerging initiatives such as Datum[9] [4], Enigma[10] [5], and Insights Network[11] [6] in the choice and particular combination of existing technologies that enable secret sharing of data, computation on secret shared data, and a transparent transactional system that allows transactions to be realized within the same framework.

- The Enigma project aims to provide a privacy protocol that enables the creation of decentralized applications that guarantee privacy. The protocol Enigma is building also advertises the use of MPC. By building the protocol directly on top of the blockchain, it removes the need of a bridge as proposed in our architecture. However, it is a project under development and no public releases exist to date.
- Datum is a project for creating a decentralized marketplace for social and IoT data. Datum provides a working product in the form of an application that allows users to submit their personal data and get paid in the Datum currency. Datum encompasses a decentralized data storage running on a smart contract blockchain. However, they do not allow any form of external computation on personal data.
- Insights Network is a data exchange based on combining blockchain technology, smart contracts, and secure multi-party computation. It is based on the EOS blockchain and a custom MPC system. Like Enigma, it is a project under development, with little technical details about the emerging implementation available to date.

[9] https://datum.org
[10] https://enigma.co
[11] https://insights.network

The design of the architecture and its implementation in a prototype has come with different challenges. For example, one of the challenges is the process of writing smart contracts and deploying them onto the Ethereum ecosystem. The language and framework itself is easy to setup and use, but the "gas", a resource that is accumulated during deployment to the blockchain, is dependent on the amount of gas provided with the SC. The error handling mechanism for debugging smart contracts does not provide intuitive error messages, so one cannot be sure if too little or too much gas has been sent. That makes it important for users to be careful about everything they write in the SC.

In terms of the proposed architecture, the SC-MPC bridge component was needed, mainly due to the lack of mechanisms for communication between existing SC engines and MPC engines. For example, if future implementations of Etherum and Sharemind provide REST APIs, the need for such a bridge would potentially be eliminated, simplifying the architecture.

In this paper, we proposed an architecture for combining MPC and SC for the purpose of implementing data markets. Our proof of concept prototype demonstrates that combining MPC and SC is practically feasible with existing technologies. As part of future work, we plan to evaluate the prototype by testing its scalability. We want to check to what degree it can handle, for example, thousands of transactions at the same time. The challenges that may arise are related to cost efficiency and latency. Also, blockchain technology is undergoing rapid changes and new frameworks and technologies often emerge. It means that we have to be on the lookout for emerging technologies that may be suitable as mechanisms to be used in the development of future data markets. Finally, it is worth mentioning that the architecture and proof of concept reported in this paper represent a first concrete realization of the reference architecture for trusted data marketplaces introduced in [7].

Acknowledgements. This work is partly funded by the EC H2020 projects euBusinessGraph (Grant number: 732003), EW-Shopp (Grant number: 732590), and TheyBuyForYou (Grant number: 780247). We thank the Sharemind team for promptly answering questions regarding Sharemind.

References

1. *Ethereum White Paper* (2013), https://github.com/ethereum/wiki/wiki/White-Paper.
2. Bogdanov, Dan. *Sharemind: programmable secure computations with practical applications*. Diss. 2013.
3. Jagomägis, Roman. "Secrec: a privacy-aware programming language with applications in data mining." *Master's thesis, University of Tartu* (2010).
4. *Datum White Paper v15* (2017), https://datum.org/assets/Datum-WhitePaper.pdf.
5. Zyskind, Guy, Oz Nathan, and Alex Pentland. "Enigma: Decentralized computation platform with guaranteed privacy." *arXiv preprint arXiv:1506.03471* (2015).
6. *Insights Network - A Blockchain Data Exchange White Paper* (2017), https://s3.amazonaws.com/insightsnetwork/InsightsNetworkWhitepaperV0.5.pdf.
7. Roman, Dumitru, and Gatti Stefano. "Towards a Reference Architecture for Trusted Data Marketplaces: The Credit Scoring Perspective." *International Conference on Open and Big Data (OBD)*. IEEE, 2016.