Blockchain: Trust Management for Authentication

ECE 8491: Blockchain Technology and Uses

Sean Carroll

Professor Hasshi Sudler

December 10, 2018

The objective of this research is to propose a novel method of optimizing identity and access management through the integration of blockchain technology.

Centralized Data Storage Versus Decentralized Data Storage

Centralized storage has several advantages, but also, several disadvantages. Centralized storage can provide greater data integrity. The data does not need multiple redundant copies, so less storage space is needed. Centralization of data can enable IT professionals to focus on more secure and resilient data protection scheme for that one, clearly defined data space. Decentralized storage could make security more difficult, and costly, because each data space may require a different methods of security protection.

Let us consider the basic concept that centralized storage is stored in one physical location. Data is ultimately accessed through one identifying address, even if multiple proxy or relay addresses are implemented, in the case of a database server. With advancements in technology, centralized storage can be very complex. Centralized servers often contain cache, virtual memory, RAID implementations for physical hard disks, and specialized hardware to allow quick access for multiple clients. Those clients, along with other clients, are often communicating, and possibly changing, the memory on the server. This creates heavy load on the server; high bandwidth, efficient access, and constant uptime are of utmost importance to support functionality and user demands. There is often a backup server to prevent downtime and support.

In the case of password management, whereby enterprise passwords are often stored in a remote, centralized database server, it is important to consider the case separate from normal data centralization. Remote centralized database servers used for password management hold

different client-server interaction. A secure server typically holds passwords that are hashed and salted. These servers are often used for remote attestation, and often inclusively, for user authentication. The client server requests hold more consistency—the handshake, although often of great security risk, holds more consistency in request type.

In consideration of personal password management, users often have many passwords across multiple devices. Those devices often do not have password synchronization, and this creates a burden for the end-user. The lack of password accessibility makes authentication difficult and can result users to use similar or repeated password for ease in authorization. This results in security concerns, for both the user and for the password management servers, as consistency in passwords increases risk. Securing passwords in a centralized password server puts reliance and trust in the security of both users and the centralized server of focus, but also in the security schemes of other password databases that cannot be controlled.

There are several password managers that provide encrypted storage vaults for user passwords. Password managers that rely on a master password to authenticate passwords on all devices provide ease-of-use for the end user for authentication. However, in doing so, the security of the encrypted vault is put into question. The encrypted vault could be implemented several ways, and it is common to have either local storage of the vault on client operating systems or in a cloud-based storage system that relies on either a third-party, or a self-hosted server. The operating system of the user, or the cloud-based storage security demands secure implementation for security of not just one password, but of all passwords. By placing many passwords in a centralized vault creates a central point of attack, provides a mechanism for greater ease-of-use by the user, and in result, dramatically increases the potential risk for the

security and privacy of the user. Table 1 includes password managers of various conceptual designs.

Name	License	Platform Support	2-FA	Delivery	Notes
1Password	Proprietary	Android, iOS, Linux, macOS, Windows, most browsers	Yes	Local with Cloud Sync	Payment, centralized
Dashlane	Proprietary	Android, iOS, Linux, macOS, Windows, most browsers	Yes	Local with Cloud Sync	Payment, centralized
Bitwarden	GPLv3	Android, iOS, Linux, macOS, Windows, most browsers	Yes	Cloud	Fully decentralized
KeePassXC	GPLv3	Android, iOS, Linux, macOS, Windows, most browsers	Yes	Local with Cloud Sync	Fully decentralized
LastPass	Proprietary	Linux, macOS, Windows, most browsers	Yes	Cloud	Payment, centralized
Zeropass	MIT	Android, iOS, Linux, macOS, Windows, most browsers	No	Cloud	Not in active development
B.lock	MIT	Chrome browser	No	Nebulas Blockchain	Uses Nebulas, Chrome only, no fees
Lesspass	GPLv3	Android, iOS, Linux, macOS, Windows, most browsers	No	Cloud	No support with server setup
Passwor-D-App	MIT	Linux, macOS, Windows, most browsers	No	Ethereum Blockchain	Gas on Ethereum is expensive

Table 1: Evaluation of password managers

Distributed Blockchain

The growing prevalence of server security breaches creates immense challenges for IT professionals. Not only are breaches a growing issue, but the growing number of DDOS attacks is also of very serious concern. If the centralized server is overloaded with requests, either from normal users, or from attackers, the centralized server cannot function in a manner that is intended. Although technologies to mitigate malicious requests can help to prevent server downtime, we must acknowledge that the problem is due to the fact that the data is centralized. Logically, storage decentralization would provide a solution to the problem of client inaccessibility (Zhangy et al., 2018). If data decentralization was implemented through security-by-design, it would increase reliability in data access. However, to migrate existing centralized databases to decentralized data storage systems should not be viewed as the panacea for solving existing problems in centralized databases. It is crucial to consider the Secure Software

Development Lifecycle (SSDLC) in this case, where centralized data storage is treated as a legacy design platform, and decentralized storage mechanisms are treated as a new and improved paradigm. By framing the situation in such a way, we can move forward with a much more feasible task. That task is to design some system where centralized data systems can leverage decentralized data storage platforms to maximizes their security.

Industry has long been searching for technologies that could possibly change the way the internet is fundamentally designed, and it appears that blockchain technology could be a potential technology to start the progression towards the "Web 3.0" movement. Centralized data and centralized database servers are often related to the client-server model. The Peer-to-Peer (P2P) data transfer model, has historically seen less utilization, but is still used in present. The latter model was recently reconsidered, and blockchain has leveraged its core principles to attempt to transform not only the internet, but, several other industries along with it.

In this paper we focus particularly on the Ethereum blockchain. We see Ethereum as a platform that could integrate current centralized data storage schemes with newer technologies to increase security, authenticity, accessibility, privacy, and trust, in the future. I want to point out that we can store centralized data on a distributed blockchain, such as Ethereum. However, one must truly ask the question: does blockchain provide added benefits over other, more implemented and proven technologies? Ultimately, industry and consumers will adopt the technology that provides reasonable ease-of-use, security, and reliability. Figure 1 displays a decision tree diagram that attempts to assist the average person in determining whether implementing a blockchain is suitable for their use. Figure 1 was adapted from Shyamasundar and Patil's 2018 paper "Blockchain: Revolution in Trust." I want to add that Figure 1 only refers to the state of blockchain during the time of this research publication.

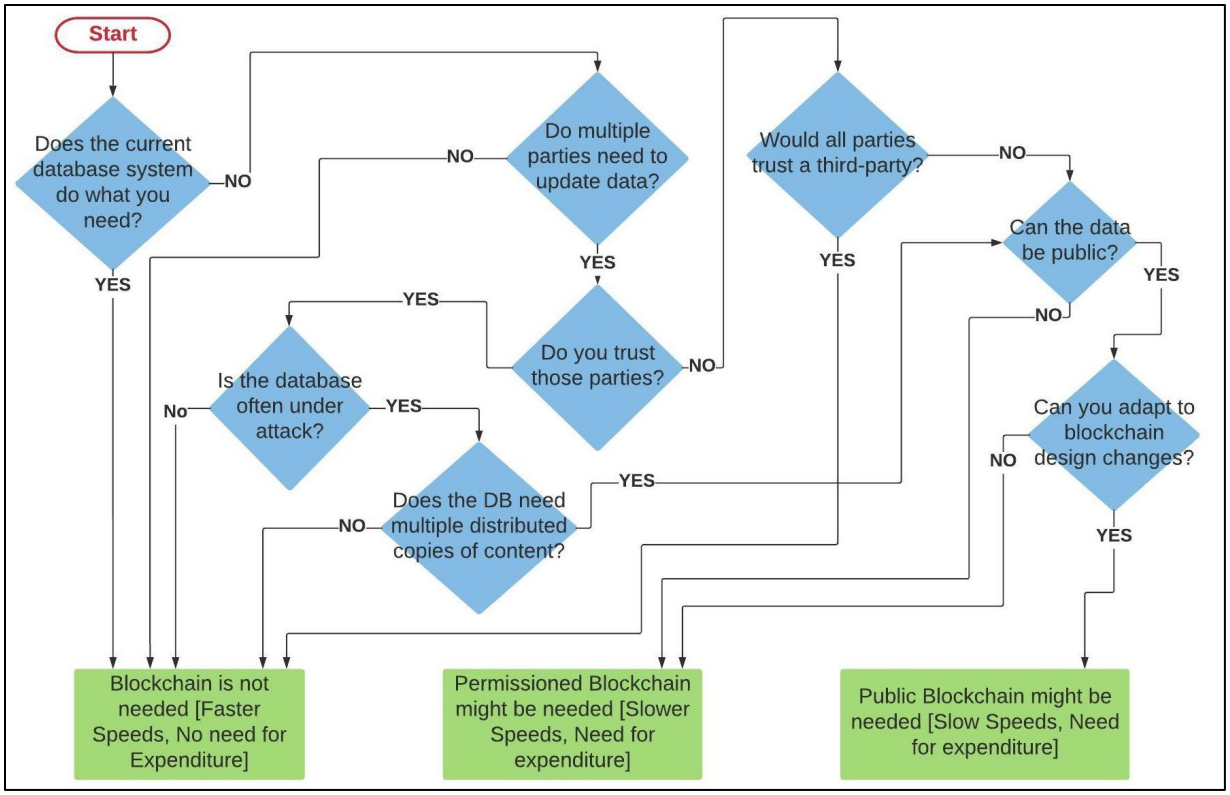

Figure 1: Decision tree diagram: should blockchain be used for data storage?

In Figure 1, the majority of decision paths lead to the conclusion that we should not use blockchain to store passwords on a blockchain. This is under the assumption that the purpose in using blockchain was to store hashed passwords on the Ethereum blockchain, coming from a centralized password database. Although storing hashed passwords on a securely encrypted blockchain would provide potential for greater accessibility of credentials for authentication, current mechanisms appear to have several advantages over using blockchain. Distributing password hashes on a distributed blockchain creates more risk than storing that data in a central location and also creates redundant information and inefficiency for the blockchain network. Although the password data is quite small, the gas costs of using Ethereum also raise cost concerns in large-scale implementations. This specific case outlines more risk and raises security and privacy concerns that are not necessary; the current centralized password management database is recommended over using blockchain.

In general, there's often a feeling of promised privacy with blockchains like Bitcoin. However, they're not actually private in full scope. In the case of bitcoin, a distributed ledger of all our transactions is tied to one private key and this allows anyone to audit the chain and find out your identity if there exists an incentive. Bitcoin provides a form of pseudo-anonymity, which can often be confused with privacy (Verbin, 2018). One could say that we are seeing a general trend in technology, towards less privacy focused schemes. We see technology giants positioning their revenue models towards data-driven models, because there is a clear monetization incentive to collect user's private data (Nabi, 2017).

Blockchains provide security and authenticity at the expense of privacy. However, if there was some way to increase privacy, there would be greater trust in the blockchain, more social trust amongst users, and that would logically lead to greater incentives to share and collaborate among users and would also result in more social capital on the blockchain (Eek and Rothstein, 2005). I want to propose an argument for using blockchain and how centralized databases could be linked to the Ethereum blockchain in a way that I think will transform the way the internet works. This proposal is rooted in the new theories within private computation that are distinct from the older ideas implemented in private communications.

Private Computation and Private Communication

Private computation mechanisms generally provide more causal trust mechanisms to ensure communication handshakes between entities. Private communication mechanisms are considered as older mechanisms to ensure communication handshakes between entities, in comparison (Verbin, 2018). Each holds a different philosophy on how handshakes should be performed. Private computation removes the need for a trust third-party (TTP) from the communication between two parties. The concept of removing a TTP from communication has the ability to transform the fundamental structure of trusted computing as whole. Trust is a very

unique concept in its own right, for, to give trust to another party, demands a base of trust that is very definitive and difficult to revoke. Trustworthiness differs from trust in the sense that trustworthiness is given but can be revoked if the other entity's behavior is not within defined expectations, or policy. TTP's are critically important to current technological network infrastructure designs, and their importance should not be dismissed. However, their trusted designs of raise security concerns due to their lack of transparency and verifiability. If we are to remove trust entirely from the handshake required for communication, through implementing concepts in private computation, we provide an environment where each party can be confirmed to have more privacy and security. By removing the need for horizontal trust in communication, we only need to rely on vertical trust, that is in this case, trust in the Ethereum blockchain platform. Private communications often rely on a TTP for verification and authentication but are known to focus more on verified encryption mechanisms to resolve the handshake of communication between parties. Before proposing an approach to applying private computation mechanisms to integrate centralized data password management on the Ethereum blockchain, we will define important terms for clarification in Table 2. Table 2 was inspired by Verbin's 2018 research into the topic of privacy.

Private Computation	
Zero-Knowledge Proofs (zk-Proofs)	A method to prove something to a party without teaching that party anything beyond the correctness of the claim.
Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs)	A zero-knowledge proof that satisfies conditions: 1. arguments are short and easy to verify, 2. zero or only little interaction between prover and verifier, 3. argument soundness only holds against a prover that has polynomially bounded computational power, 4. prover cannot construct an argument without having a certain piece of knowledge (witness).
Zero-Knowledge Scalable Transparent Arguments of Knowledge (zk-STARKs)	A proof technology superior to zk-SNARKs due to lack of need for "trusted setup." It implements much simpler cryptographic assumptions, avoiding the need for elliptic curves, pairings and the knowledge-of-exponent assumption and instead relying purely on hashes and information theory. The proof is secure against attackers with quantum computers.

Secure Multi-Party Computation (SMPC)	Methods for parties to jointly compute a function over their inputs while keeping those inputs private. Unlike traditional cryptographic tasks, where the adversary is outside the system of participants (an eavesdropper on the sender and receiver), the adversary in this model controls actual participants.
Homomorphic Encryption	An encryption method that allows computation on ciphertexts, generating an encrypted result which, when decrypted, matches the result of the operations as if they had been performed on the plaintext.
Trusted Execution Environment (TEE)	An execution space within the primary CPU that provides a higher level of security.
Private Communication	
Public Key Cryptography	Cryptography that uses separate keys for encryption and decryption; also known as asymmetric cryptography.
Hash Function	A function on bit strings in which the length of the output is fixed. The output often serves as a condensed representation of the input.
Key Distribution	A key-establishment procedure whereby one party (the sender) selects a value for the secret keying material and then securely distributes that value to another party (the receiver) using an asymmetric algorithm.
Digital Signature	The result of a cryptographic transformation of data which, when properly implemented, provides the services of: 1. origin authentication, 2. data integrity, and 3. signer non-repudiation.
Access Control System (ACS)	A set of procedures and/or processes, normally automated, which allows access to a controlled area or to information to be controlled, in accordance with pre-established policies and rules.
Authentication	A process that establishes the source of information, provides assurance of an entity's identity or provides assurance of the integrity of communications sessions, messages, documents or stored data.
Authorization	The process of verifying that a requested action or service is approved for a specific entity.
Audit	The independent examination of records and activities to ensure compliance with established controls, policy, and operational procedures and to recommend any indicated changes in controls, policy, or procedures.

Table 2: Definitions of privacy computation and privacy communication terms

In private computation, SMPC and Homomorphic Encryption show very promising futures, as they are substantially more provable privacy mechanisms. However, their algorithms need to become more efficient in order to implement them on a distributed blockchain (Verbin, 2018). Zk-STARKs also has greater benefits over zk-SNARKs, as outlined in the table, but more research needs to be conducted to implement efficiency requirements suitable for the Ethereum blockchain. I want to highlight the two remaining terms in the private computation section of the table: zk-SNARKs and TEE, as they are fully functional technologies, the former is implemented successfully on many blockchains, including Ethereum, the latter is commonly implemented in

the computer processors. My proposed design proposal for this research paper required immensely comprehensive research into two very technically complicated ideas: zk-SNARKs and Trusted Execution Environment.

Before discussing my design framework, I want to provide one small example of how zk-SNARKs might be implemented on the Ethereum blockchain. If we have a DApp that we are implementing on the Ethereum network and want to ensure the privacy of our users, we can configure our DApp to a set format (termed ideal scenario in Figure 2) to define the function *f*, and then implement one of the many zk-Proof or SMPC libraries that are actively development and updated. Many of these libraries are open-source and easily accessible to anyone: libsnark, libstark, snarky, fairplayMP, spdz, and bellman are all libraries written in various languages that are stable and functional (Verbin, 2018). If we can carefully define function *f* and design our DApp in a way that is securely written and provisioned to ensure resiliency over time, we have successfully optimized our DApp using private computation methods. The resulting DApp would then conceptually model the "real scenario" shown in Figure 2. The "real scenario" is a conceptual model in which communication between parties is verified, the data and user privacy of each party is ensured through definitive proof, and the handshake is handled by a verifiable, open-source framework. The complex mathematics involved in proving the zk-SNARKs process is not within the intended scope of this research. The concept behind zero-knowledge interactions demands further research. Furthermore, if a platform, such as Ethereum, was designed with greater privacy, and more importantly, greater vertical trust, there less conflict in collaboration between parties, because the social or horizontal trust level would also increase.

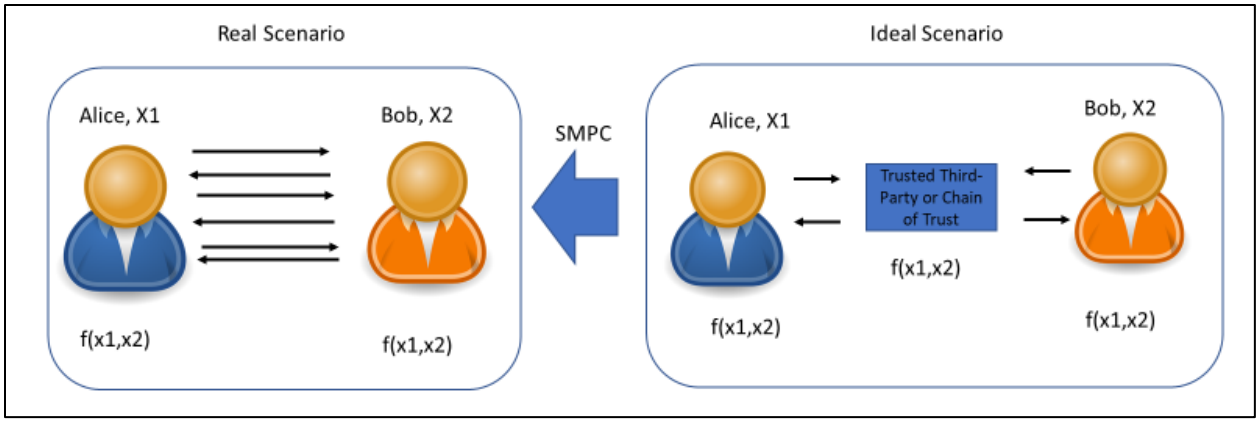

Figure 2: Modification of a communication platform using SMPC (in which the initial platform format is ideal)

It is important to clarify the blue box represented in Figure 2. Assuming an ideal scenario format for defining function *f*, Alice and Bob would presumably be communicating in a commonly secure manner through use of a cryptographic protocol, such as a symmetric or asymmetric-key protocol. The middle box could represent an intermediate certificate authority as part of public key infrastructure (PKI), or some similar mechanism. All these mechanisms still rely on a TTP, which can raise concern when attempting to achieve provable security. The DApp example given above replaced the TTP within the Dapp with a code library implementation of a zk-SNARK proof. In doing so, we increased the security of the Dapp by replacing TTP with a mathematical proof. Furthermore, the blue box could represent lower-level hardware, such as a firmware trusted platform module (TPM) implemented in a TEE.

Design Proposal

Homomorphic encryption and Secure Multi-Platform Computation do not currently have the functionality that are needed for pragmatic implementations. However, integration of these technologies with more efficient algorithms in the future would enable an even better option for trust management and authentication (Harnik, D., Ta-Shma, P., & Tsfadia, E.,2018)

Although blockchain may not make sense to store passwords on, it would make sense to store credentials in TEEs and employ zkSNARKs to securely authenticate credentials with other

parties and utilize blockchain as a method of validation and witness if needed. The high-level architectural hierarchy is shown in Figure 3. Centralized data storage, in the case of a password centralized server could use virtual encrypted enclaves that TEE technology provides, in the cloud if need for integration. The most optimal integration would be to use physical devices and provide remote attestation server in the cloud. The best option for a secure remote attestation appears to be a remote trusted environment (RTE), as reference in the top right of Figure 3. We must acknowledge that a closed source consortium would be needed to integrate the TEE with several technologies on the layer above, however, we will move more towards minimum platform design as these technologies are continually refined (The Confidential, 2017). QED-it, Quorum, Hyperledger Sawtooth, Corda, and Enigma, are examples of technologies that are working on compatibility with TEE through the use of zkSNARKs. This would enable a secure integration with DApp on the blockchain layer, would further incentivize third-parties to build on the Ethereum distributed chain because of the secure and trusted design of the layers below shown below in Figure 3. The security of TEE has been greatly researched and, secure optimizations have been considered in my previous work. A secure gateway between the blockchain would separate the blockchain from the higher layer above it and integration through blockchain nodes could be used, however, implementation may vary based on the development of technologies on the highest layer in Figure 3. Our architectural design would incentive centralized storage systems to integrate with the blockchain, because migration to a new technology would not necessarily be needed. Ideally a RTE is utilized for remote attestation which would could be in the form of a node on the blockchain, but not necessarily. Furthermore, other database blockchain storage technologies could be connected, such a IPFS, Sia, and Swarm. Other blockchains could potentially integrate this technologies with Ethereum. The

possibility of other integrating other blockchains is important to note, but also tangential to the focus of our design.

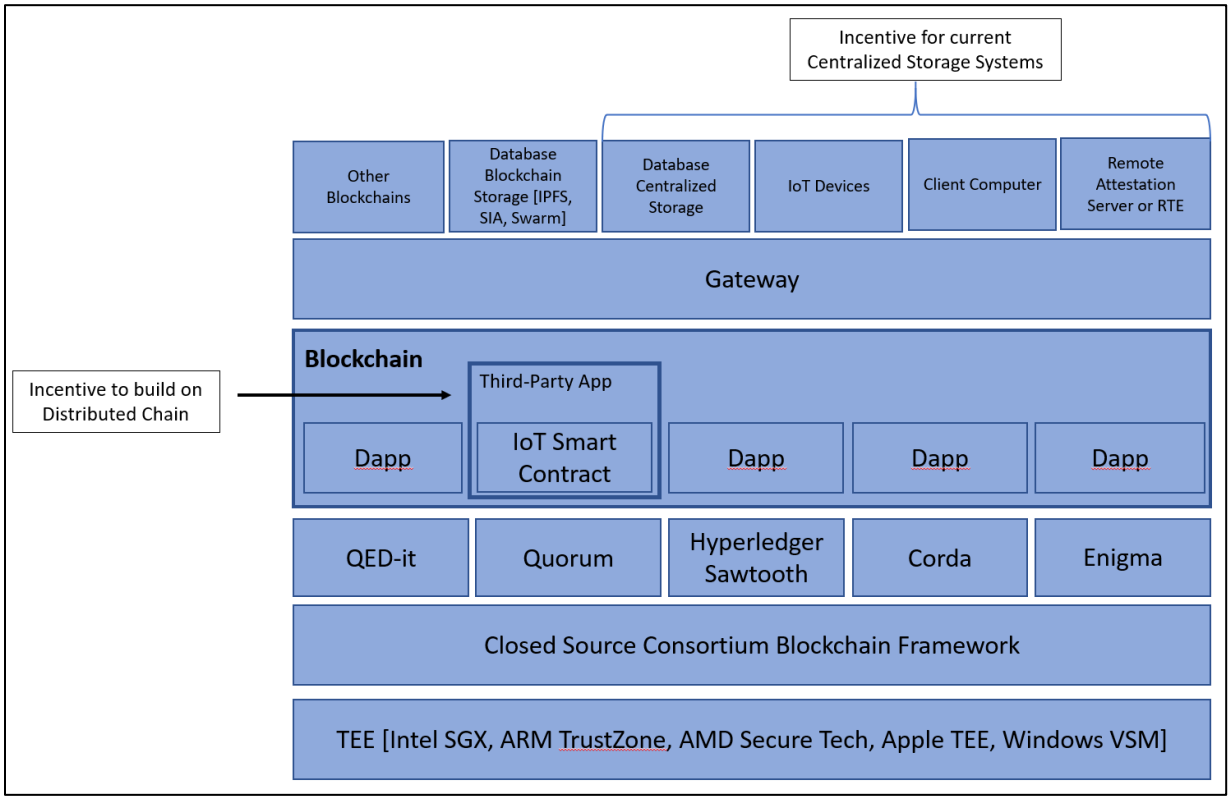

Figure 3: Proposed Architectural Design Overview

For the end-user, the design in Figure 3 enables less hassle and increased access across all their trusted devices, as their devices could potentially authenticate and provide access without the need for end-users to remember credentials and input them. If an unrecognized device is used by the end-user, two-factor authentication (2-FA) could be used and other recognized devices could verify access. Figure 4 provides an overview of how TEE is often implemented. Technically speaking, the encrypted enclave subsystem of TEE is a layer above the operating system, but the Operating System layer does not have access to it directly. In example, Intel Secure Guard Extensions (SGX), is a layer above the operating system, but is a subsystem of the entire TEE provided within the Intel vPro technology suite (Küçük et al., 2016).

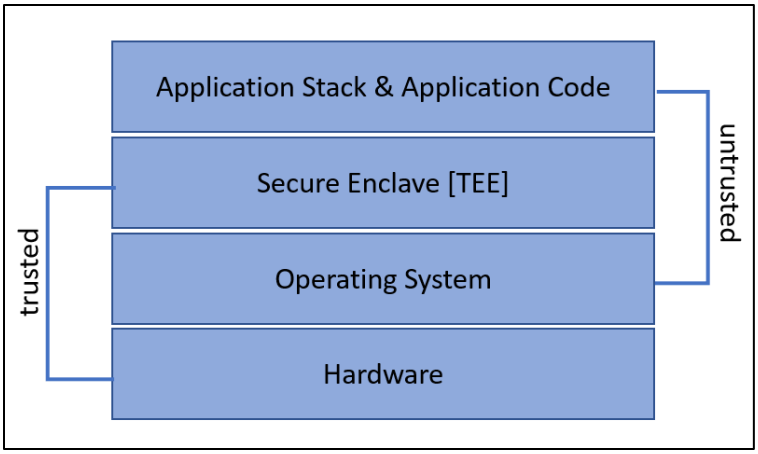

Figure 4: Traditional TEE architectural design

Conclusion

In this paper, we initially discussed the advantages and disadvantages of centralized data storage and decentralized data storage. We proceeded to evaluate several centralized password management options. Furthermore, we considered whether distributing passwords on a blockchain would provide advantages over other implementation options. This led us into a discussion of private computation and private communication technologies in the interest of promoting privacy on the blockchain. Lastly, we proposed a design that leveraged zkSNARKs and TEE to provide greater privacy, trust and security, for user-authentication, which argued for integrating existing centralized data management into the Ethereum distributed blockchain. Further research is needed into this design, and it is the hope that this research will provide a basis for greater research and technological development.

References

Eek, D., & Rothstein, B. (2005). Exploring a causal relationship between vertical and horizontal trust.

Filippi, P. D. (2016). The interplay between decentralization and privacy: the case of blockchain technologies, 19.

Harnik, D., Ta-Shma, P., & Tsfadia, E. (2018). It Takes Two to #MeToo - Using Enclaves to Build Autonomous Trusted Systems. ArXiv:1808.02708 [Cs]. Retrieved from http://arxiv.org/abs/1808.02708

Küçük, K. A., Paverd, A., Martin, A., Asokan, N., Simpson, A., & Ankele, R. (2016). Exploring the use of Intel SGX for Secure Many-Party Applications. In Proceedings of the 1st Workshop on System Software for Trusted Execution - SysTEX '16 (pp. 1–6). Trento, Italy: ACM Press. https://doi.org/10.1145/3007788.3007793

Nabi, A. G. (2017). Comparative Study on Identity Management Methods Using Blockchain.

Shyamasundar, R. K., & Patil, V. T. (2018). Blockchain: Revolution in TRUST. Proceedings of the Indian National Science Academy, 96(0). https://doi.org/10.16943/ptinsa/2018/49340

The Confidential Consortium Blockchain Framework: Technical Overview. (2017). Microsoft Azure. Retrieved from https://github.com/Azure/coco-framework

Verbin, E. (2018, October). Zero Knowledge Proofs and Their Future Applications. Web 3 Summit. Retrieved from https://2018.web3summit.com/video/zero-knowledge-proofs-and-their-future-applications/

Zhangy, S., Kim, A., Liu, D., Nuckchadyy, S. C., Huangy, L., Masurkary, A., … Zhang, Z. (2018). Genie: A Secure, Transparent Sharing and Services Platform for Genetic and Health Data. ArXiv:1811.01431 [Cs]. Retrieved from http://arxiv.org/abs/1811.01431