

**University of Victoria
Faculty of Engineering
Summer 2018 ENGR 446 Report**

Creating transparent and efficient transactions through smart contracts and blockchain

**University of Victoria
Faculty of Engineering
Victoria, British Columbia**

**David Li
V00818631
ENGR 446
Engineering
lidavid@uvic.ca**

Sunday, 21st July

**In partial fulfillment of the requirements of the
B.Eng. Degree**

Insert Title of Report Here
ENGR 446
4A Computer Engineering
Dr Helen Bailey
Faculty of Engineering
University of Victoria
P.O. Box 1700
Victoria, B.C.
V8W 2Y2

Dear Helen,

Please accept the accompanying ENGR 446 Final Technical Report entitled "Creating transparent and efficient transactions through smart contracts and blockchain."

This report is the result of interest and exposure to blockchain development and technologies as part of my third co-op work term. Using the most popular smart contract programming language, Solidity, I created logic for a decentralized application. Some issues of implementing and determining the viability of smart contracts to simplify transactions include regulation, conflict resolution and ability to reverse fraudulent transactions. Technological advances including exponential increases in computational power that render modern cryptographical useless are not explored. Some problems faced in this report include rapid changes in technology, constantly changing standards, and legal uncertainty surrounding blockchain.

I would like to thank my manager, Dino Coletti, for his patience and good judgement, as well as the my coworkers who were always willing to help.

Sincerely,

Table of Contents

List of Figures	iii
List of Tables	iii
Summary	iv
Glossary	v
1 Introduction	1
1.1 Background	1
1.2 Objective	3
1.3 Aims	4
2 Discussion	5
2.1 Potential Solutions	5
2.2 Initial Assessment	6
2.3 Engineering Analysis	6
3 Conclusion	10
4 Recommendations	11
5 References	12
Appendix A Code Listings	14

List of Figures

1	An example of server-blockchain architecture in a DAPP.	1
2	Illustrating how a smart contract works	2
3	Architecture of Hyperledger composer	5
4	Difficulty of brute forcing a 256-bit key for different computers	7
5	Lifecycle of a smart creation [15]	8
6	Decentralized structure vs Centralized Structure	10

List of Tables

1	Timeline of Cryptocurrency	1
2	Sample Decision Matrix for designing a blockchain system	6

Summary

Growing usage on digital services and interconnectivity have revolutionized modern society and simplified purchase goods, but created a centralized web ecosystem. Web 3.0, or the decentralized web built on blockchain technologies allow connectivity and transactions between peers instead of trusting large centralized companies. The major types of blockchains are public, permissioned and private with varying amount of access. In addition, smart contracts are self executing code that directly controls transfer of assets under predefined conditions.

With access to significantly public resources, a open-source base, a public blockchain provides high processing speeds and support for smart contracts. Furthermore, smart contracts can be used to streamline and eliminate middlemen in transactions. Regulatory uncertainty, lack of legal acceptance for smart contracts and scaling issues with blockchain hinder adoption. Despite advantages of smart contracts to simplify transactions, writing bug-free code is virtual impossible and following best practices is essential to limit risk. Overall, for certain transactions with clearly defined conditions, smart contracts can greatly reduce cost while increasing transparency.

Glossary

blockchain A blockchain is a digitized, decentralized, public ledger of all cryptocurrency transactions. It constantly grows as the most recent transactions (blocks) are appended in chronological order. 1–4, 6–9, 11

Dapp Decentralized Application have backend code running on a decentralized peer-to-peer network and not controlled by a single entity. In a decentralized application transactions are verified through consensus of multiple users. 1, 2

Ethereum an open software platform based on blockchain technology that enables developers to build and deploy decentralized applications. The main unit of currency is ether. 1, 3, 7

HyperLedger group of open-source blockchain technologies started by the Linux Foundation 1

HyperLedger Composer permissioned blockchain that defines assets, business rules, and participants, and access controls for existing roles and types of transactions. 11

side chains are emerging mechanisms enable secure transfer of digital assets and tokens between blockchains. Attached to the parent blockchain using a two-way peg, sidechains are a separate blockchain and enables interchangeability of assets at a predetermined rate [1]. 11

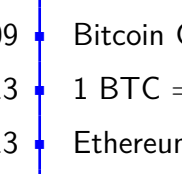
smart contract computer program that directly controls transfer of digital currencies or assets under predefined conditions and used to automatic transactions on the blockchain. These transactions are trackable and irrevocable. 1, 2, 4, 7–9

1. Introduction

1.1. Background

History of Cryptocurrency

TABLE 1 Timeline of Cryptocurrency



A vertical timeline with a blue line and square markers. The years are listed on the left, and the corresponding events are listed on the right.

Year	Event
2008	Bitcoin White Paper
2009	Bitcoin Genesis Block
2013	1 BTC = \$ 31 USD
2013	Ethereum White Paper
2015	Ethereum Genesis Block
2015	HyperLedger starts
2017	Over 1000 different cryptocurrencies
2018	AWS Blockchain Templates

In 2008 bitcoin white paper [2] described a way to solve the double spending problem without a centralized body using blockchain. Although, the value of bitcoin (BTC) has grown exponentially, high computational and energy consumption in mining and slow performance [3]. Released in July 30, 2015, Ethereum, an open-source platform based on blockchain technology, distinguishes itself from bitcoin through faster transactions, unlimited processing capability for smart contracts, and its network is optimized to support Decentralized Application(DApp) [4].

Decentralized Applications

Blockchain technology is revolutionizing the internet by establishing trust in shared data. [5]. Additionally, transactions recorded on the blockchain are practically impossible to remove or change.

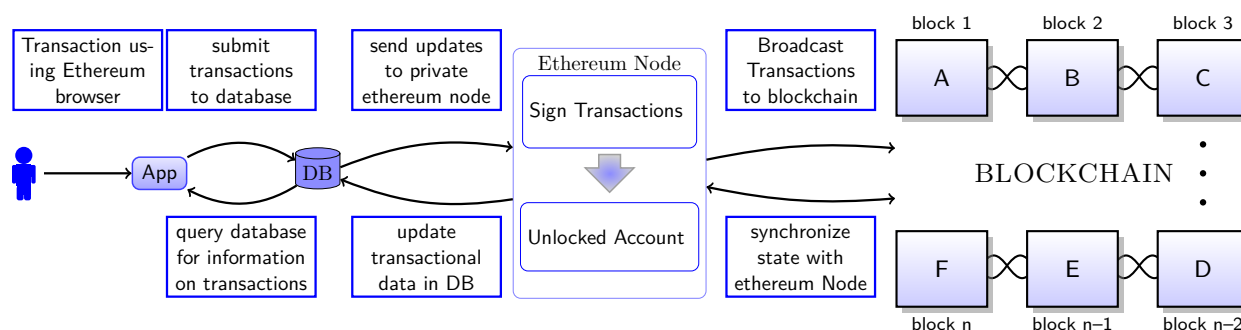


Figure 1: An example of server-blockchain architecture in a DAPP.

A decentralized application, or Dapp are deployed on peer to peer networks such as Ethereum or on the cloud ¹. A decentralized system (peer to peer) has many advantages over a conventional centralized network including no single points of failure, cheaper distribution (servers are expensive), faster upload speeds.

¹Amazon recently started offering blockchain templates on Amazon Web Services (AWS).

Despite, the obvious advantages of decentralized and blockchain technologies, a lack of resources for unpopular materials may result in prolonged downloads. An equivalent anecdote is a inadequately seeded torrent results in intolerable download speeds. Whereas a traditional database only checks, writes and stores data once, a decentralized blockchain system require thousands of operations to write and store data, therefore the costs of maintaining a blockchain are substantial higher and only justifiable with increased utility/security [6].

Public and Private Keys In a blockchain system, any key holder can use their private key to sign a piece of data. This results in a signature. In a Dapp, this can be used for:

1. Recovering the public key (ethereum account address) of the Author.
2. Verify if the raw data is identical using the Author's public key.

Blockchain technology has been proposed as a supplement to help consumers prove ownership of assets in digital systems. Blockchains are fantastic public recordkeeping systems because they also cant be backdated or changed without a recorded transaction. In theory, blockchains could transform untrustworthy transactions into distributed databases.

Smart Contracts

Traditional legal contracts are written to represent the contracting parties. In a smart contract, self-executing source code is used to automatic transactions that are publicly available on the blockchain [4]. Furthermore, smart contracts allow buyers and sellers exchange money, property, shares, or anything of value in a transparent, conflict-free way while avoiding the services of a middleman (See Figure 5). This allows validation of complex transactions swiftly while maintaining transparency. Although the benefits of using smart contracts are obvious, legal enforceability is difficult because "no central administering authority to decide a dispute" exists [7].

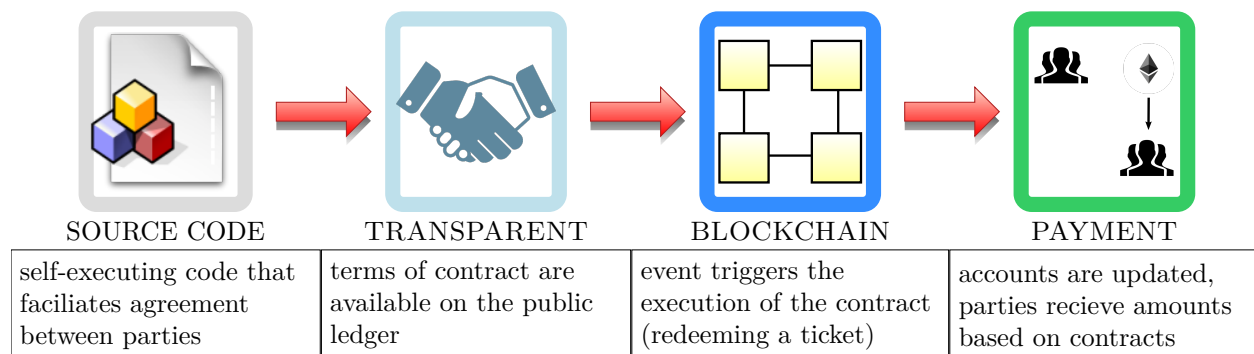


Figure 2: Illustrating how a smart contract works

In addition, irreversible and immutable transactions are a disadvantage that hackers can exploit. For example, an amateur coder killed the contract that allowed users to transfer Ether for the Parity Ethereum Wallet, rendering 150 to 300 million dollars completely useless [10]. Scrutinizing smart contracts and reducing bugs in production code is essential. An example of smart contract is available in Appendix A 1.

The desirable properties of a blockchain such as integrity and immutability of data, e.g., transactions and smart contracts, require an active mining pool. In a proof of work consensus (POW) mechanism, miners need to invest energy and resources, i.e. computational power, to participate in the consensus process and prevent tampering blocks. Additionally, miners are incentivized to run nodes that verify transactions through bitcoin or ether.

Advantages and Disadvantages of Decentralization

Currently, centralized IT systems are vulnerable to "malicious attacks, software and hardware faults, human mistakes (e.g., software and hardware misconfigurations)" [8]. Decentralized systems have no single point of failure, improved security and are more transparent, however, efficient code is more important in smart contracts. As illustrated in Figure 1 a blockchain-server architecture model allows for developers to implement decentralized applications with smart contracts while maintaining the flexibility and simplicity of retrieving and sending information.

1.2. Objective

The prominence of cryptocurrency and decentralized applications suggests usage of smart contracts will experience explosive growth.

Problem

Currently commonplace transactions require days to process and for parties verify correctness. For example to purchase houses, a plethora of steps are required, one must interactive with lawyers, real-estate agents, home inspector, buy insurance and shop for a mortgage.

Purpose

Leveraging existing blockchain technologies can automatic the majority of steps and cut out the middlemen, resulting in buyers conversing directing with sellers. Although smart contracts have immense potential to simplify transactions, issues such as limiting access to information, latency when updating (takes 10 minutes to write info to bitcoin blockchain), and immutability of translations (cannot undo fraudulent transfer of assets) must be addressed.

1.3. Aims

The aims of this project are to illustrate how a decentralized blockchain system can:

1. Reduce cost of transactions by at least 50% from removing middlemen.
2. Improve transparency in software systems through augmented accessibility and understandability.
3. Increased security and greater enforceability of contractual obligations.

Limitations

The regulatory uncertainty and impact of future regulations on blockchain technologies such as smart contracts will not be investigated. In addition, criminal usage of cryptocurrencies to avoid taxation and legal repercussions are beyond the scope of this report. In addition the impacts of quantum computing altering the validity of modern cryptography algorithms will not be investigated.

2. Discussion

2.1. Potential Solutions

- **Public blockchains** are large distributed networks that are run through a native token such as bitcoin or ether. Anyone can participate and the community maintains its open-source code. The two largest public blockchains are Ethereum and Bitcoin. They are open for anyone to participate at any level and have open-source code that their community maintains.
- **Permissioned blockchains** define role based access control for individuals in the network and uses native tokens. HyperLedger Composer, an open-source framework for permissioned blockchains, is used for smart contracts and for blockchain application development [9]. One use case is an accounting system that calculates payment, while hiding that information from unrelated organizations.

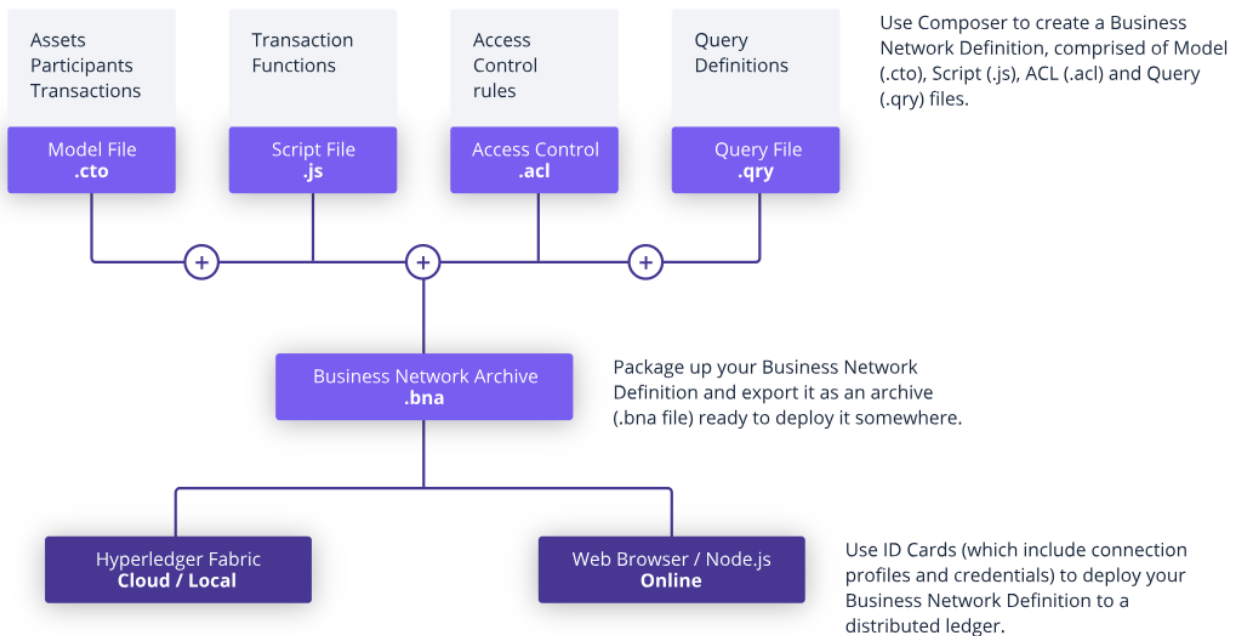


Figure 3: Architecture of Hyperledger composer

- **Private blockchains** membership is tightly controlled and lacks a native token. Useful for consortiums with trusted associates and exchanging confidential information, however, less powerful because it is supported by limited private resources. Large organizations such as governments will likely use these extensively.

2.2. Initial Assessment

Determining which platform is best for smart contracts should be done using a weighted decision matrix, based on the particular application. For internal processes such as supply chains, a private blockchain is sensible (data cannot be changed) and cryptographic auditing with known identities (public keys). For a trustless system that verifies every transaction, using a public blockchain is essential. In comparison, role-based access control is feasible by using a permissioned blockchain.

Table 2: Sample Decision Matrix for designing a blockchain system

Criteria	Existing Systems	BlockChain Systems		
	Centralized	Public	Permissioned	Private
speed and latency	5	7	7	6
scalability	5	9.8	7	4
security and immutability	3	7	8.5	9
storage capacity	4	9	9	6
transparency	3	9	7	5
Total	21	41.6	38.5	30

Despite the slow speed of the public blockchain, innovations such as side chains enable quick transactions and are used in decentralized game development [12]. A permissioned blockchain allows role based access control which is essential in business applications. One example is to prevent unrelated parties from viewing other's data. Furthermore, smart contracts allow buyers and sellers exchange money, property, shares, or anything of value in a transparent, conflict-free way while avoiding the services of a middleman. This allows validation of complex transactions swiftly while maintaining transparency. Usage private and permissioned blockchain cannot be ruled out, but generally, a public blockchain is advantageous.

2.3. Engineering Analysis

Security of Smart Contracts

Blockchain transactions are secure because that are immutable and decentralized. However, exploiting bugs in smart contracts are financially devastating [10] as fraudulent transactions cannot be reverted. Disconnects between software developers and security experts has resulted software vulnerabilities which are the root cause of computer security issues [11]. Although

blockchain technologies increase underlying security and reliability, exploiting poorly coded and insecure smart contract remains a major risk, and releasing open-source code allows hackers exploit flaws in the codebase before corrective processes are applied.

Brute force cracking of private keys

A Ethereum key, which is randomly selected 256 digits [4], is very difficult to hack. A simple calculation illustrates the impracticalities of brute forcing for a 256 bit key. Assuming that a 1 exaflop (10^{18} calculations per second) 15 megawatts supercomputer [13] is used, electricity costs are 0.1326 per kWh [14].

$$\text{Possible number of private keys} = 2^{256} = 1.1569 \times 10^{77} \text{decryptions} \quad (1)$$

$$10^{18} \frac{\text{decryptions}}{\text{second}} \times \frac{3.154 \times 10^7 \text{second}}{1 \text{year}} = 3.154 \times 10^{25} \frac{\text{decryptions}}{\text{year}} \quad (2)$$

$$\text{Time to decrypt a 256 bit key} = \frac{1.1569 \times 10^{77}}{3.154 \times 10^{25}} \text{years} = 3.66804 \times 10^{51} \text{years} \quad (3)$$

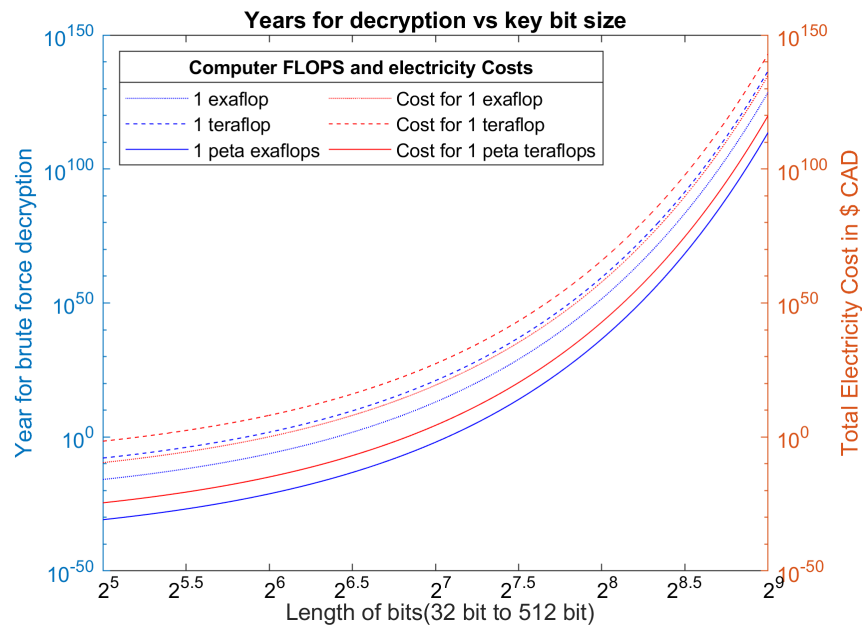


Figure 4: Difficulty of brute forcing a 256-bit key for different computers

The plots in Figure 4 illustrate how brute forcing hacking is economical unfeasable, unless computational power grows exponentially.

Scalability is the biggest issue facing blockchain technology, because storage is expensive, and public blockchains are trustless, every transaction is verified [6].

Enforcability of Smart Contracts

If contractual fulfillment are delegated to code, it becomes paramount to ensure that such code contains no errors. Oftentimes smart contracts run on top of a blockchain, so the code of the smart contract would be neither incorruptible nor secure. When self-enforcement guarantees performance and subsequent human intervention is disallowed, modification of a smart contract is impossible, logically, its code must be perfect.

Lastly, it must be acknowledged that self-enforcement may be limited to transactions where off-chain events are computationally verifiable. Just like not all contractual obligations can be represented in code, not all off-chain events can be captured as computer-readable data or measured with objective criteria.

Transparent and cheaper transactions

As in the analog world, the negotiation of the content and the clauses remains. Still, the involved parties need to agree on the contracts and the modalities considering the potentials of a decentralized execution infrastructure. Since the contract is represented as software code, the implementation effort (see Figure 5) must not be neglected.

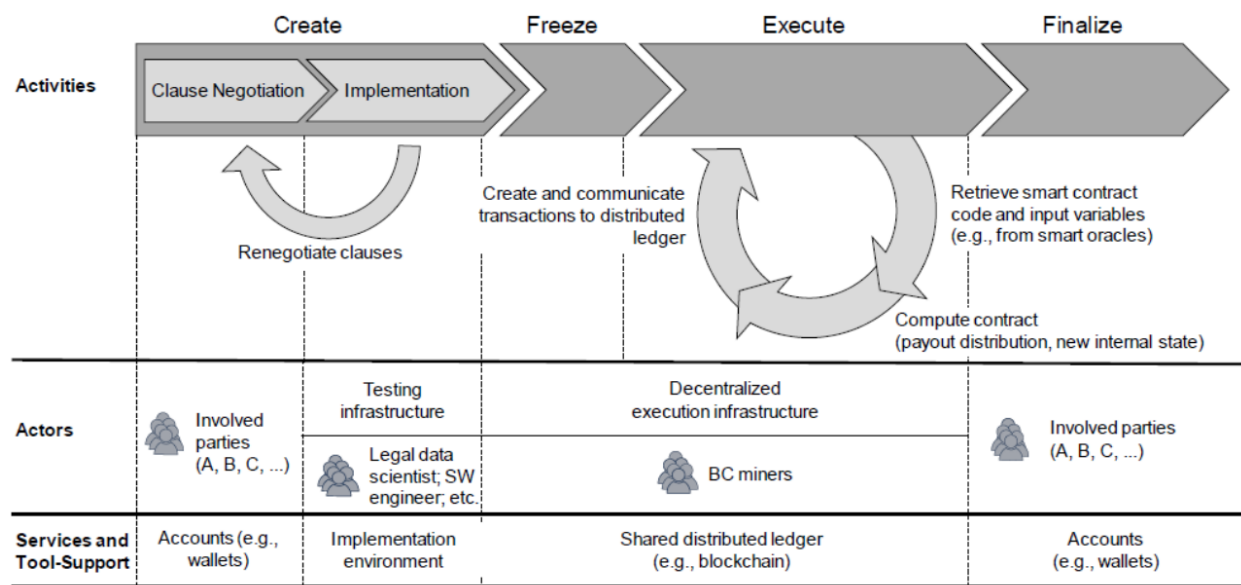


Figure 5: Lifecycle of a smart creation [15]

From an organizational point of the view, the question arises, which persons have the required skills to implement a smart contract, i.e., translation into machine readable code. In addition, the machine readable code, i.e., programming language, needs to be powerful enough to express the required norms with all necessary conditions.

Performance of Smart Contracts

Statistically, every computer program coding error ("bugs"). It is practically impossible to ensure perfect performance of a smart contract. Immutable, self-executing smart contracts may protect the transaction from unexpected human interactions but introduce the risk of performance being affected by coding errors. Given that the smart contract may execute incorrectly due to a coding error, the practical difficulty of writing bug-free code, alleviating and reducing risk from vulnerabilities is essential.

Technical writings suggest that such direct coding of smart contracts would force lawyers to be more precise and structured in describing the rights and obligations of the parties [16]. Smart contracts could reduce ambiguity because it must be capable of a single interpretation by a code compiler. Since most lawyers are unlikely to become programmers and vice-versa, converting existing legal documents into smart contracts is undesirable. In order to create smart contracts, obligations must be reported in a detailed manner and provide objective criteria for execution.

Drawbacks of Blockchain Transactions

Disadvantages of blockchain data storage include difficult retrieving relevant information (without an abstraction layer, the entire blockchain or a single transaction is returned), users will experience latency before transactions are validated,² and writing to the blockchain is relatively expensive compared to traditional systems. Smart contracts are useful because they cannot be changed by the parties involved in the transactions, yet, in some cases such as criminal activity, the ability to reverse transactions or freeze accounts is extremely desirable. For example EOS, a centralized blockchain platform, was criticized for freezing accounts without due process and community backing. [17].

²For bitcoin, it takes 10 minutes before blocks of transactions are validated (mining process)

3. Conclusion

Decentralized systems are inherently more reliable, resilient against brute force attacks, and are more transparent [2, 4]. Fulfillment of smart contracts is limited due to off-chain information, logistical challenging obtaining impartial or truthful inputs from stakeholders, and verification of transaction completion involving real-world assets. This suggests effectiveness of smart contracts is currently limited to digital assets rather than for physical assets. In addition, translating legal-binding contracts to code is challenging because the majority of programmers lack a legal background.

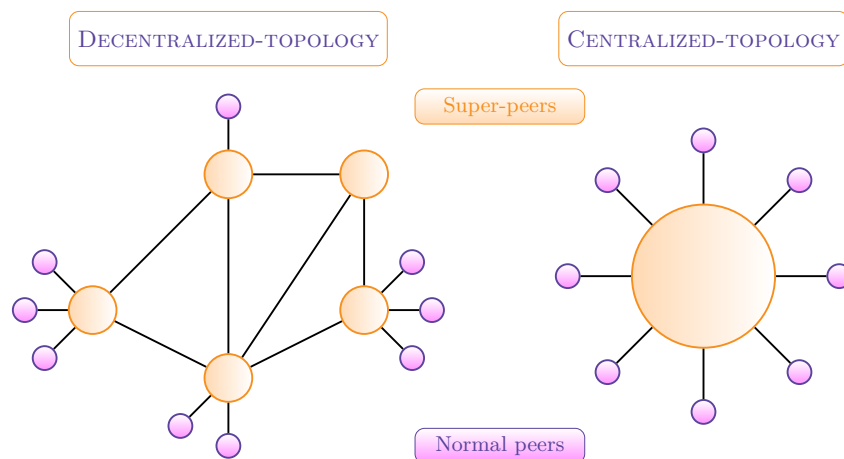


Figure 6: Decentralized structure vs Centralized Structure

Despite shortcomings that involve transactional steps that are computationally impossible to verify (package is delivered), performance of smart contracts is superior as transactions are tamper-proof. Additionally, reduction of ambiguity from usage of smart contracts is highly probable because only one interpretation is possible. Coding errors or bugs exist in every non-trivial application, therefore transactions through smart contracts intrinsically carry some risk. This implies protocols and prior real-world agreements are necessary to migrate risk when executing smart contracts. Furthermore, increased precision and detail for transactional inputs and outputs required for creating smart contracts would benefit all parties.

As shown in Figure 4 only computers with significant high processing power (quantum computers) can decrypt a 256 bit ethereum key cost-effectively. This indicates trying to hacking private keys is not a worthwhile venture. Overall, smart contracts in conjunction with blockchain technologies allow for users to have a unique digital presence, securely transfer ownership of assets and avoid key shortcomings of centralized IT systems.

4. Recommendations

Continued research into blockchain technologies is necessary as innovations such as side chains, permissioned blockchains such as HyperLedger Composer, widespread adoption of cryptocurrency and technologies evolutions address the shortcomings of blockchain. In trustless systems tampering or modification of transactional history is extremely difficult. This suggests that smart contracts can greatly improve transactions, however, issues including scalability, reversing fraudulent activity and reduction of contract deployment hinder mass adoption.

All computer programs have bugs, by limiting complexity of smart contracts, using reliable standards highly audited packages, and using software security analyze tools risk can be greatly reduced. Continuing to monitor technology innovations such ever-increasing computational power is important to ensure cracking private keys remain uneconomical see Figure 4.

Provided a deterministic set of inputs and outputs, with computational verifiable conditions, using smart contracts is beneficial. Increasing awareness about blockchain technologies, so that the average person understands the limitations of smart contracts and suitable applications is a key aspect in increasing adoption and widespread acceptance of blockchain technologies.

5. References

Cited References

- [1] Adam Back Matt Corallo. *Enabling Blockchain Innovations with Pegged Sidechains*. [Online] Available: <https://blockstream.com/sidechains.pdf>. Accessed May 31, 2018.
- [2] *Bitcoin White Paper*. [Online] Available: <https://bitcoin.org/bitcoin.pdf>. Accessed April 25, 2018.
- [3] Saeed Elnaj. *The Problems With Bitcoin And The Future Of Blockchain*. [Online] Available: <https://www.forbes.com/sites/forbestechcouncil/2018/03/29/the-problems-with-bitcoin-and-the-future-of-blockchain>. Accessed May 06, 2018.
- [4] *Ethereum White Paper*. [Online] Available: <https://github.com/ethereum/wiki/wiki/White-Paper>. Accessed April 25, 2018.
- [5] Tiana Laurence. *Blockchain For Dummies*. 1st ed. For Dummies: Computers. For Dummies, 2017. ISBN: 1119365597,9781119365594.
- [6] Vitalik Buterin. *Ethereum scalability research and development subsidy programs*. [Online] Available: <https://blog.ethereum.org/2018/01/02/ethereum-scalability-research-development-subsidy-programs/>. Accessed June 28, 2018.
- [7] Norton Rose Fulbright and R3. *Can smart contracts be legally binding contracts?* [Online] Available: <http://www.nortonrosefulbright.com/files/norton-rose-fulbright-r3-smart-contracts-white-paper-key-findings-nov-2016-144554.pdf>. Accessed May 29, 2018.
- [8] P. Smith et al. "Network resilience: a systematic approach". In: *IEEE Communications Magazine* 49.7 (2011), pp. 88–97. ISSN: 0163-6804. DOI: 10.1109/MCOM.2011.5936160.
- [9] *Hyperledger Composer Overview*. [Online] Available: <https://www.hyperledger.org/wp-content/uploads/2017/05/Hyperledger-Composer-Overview.pdf>. Accessed May 27, 2018.
- [10] Thijs Maas. *Yes, this kid really just deleted 300 MILLION by messing around with Etheurems smart contracts*. [Online] Available: <https://hackernoon.com/yes-this-kid-really-just-deleted-150-million-dollar-by-messing-around-with-ethereums-smart-2d6bb6750bb9>. Accessed May 29, 2018.
- [11] Bingchang Liu et al. "Software Vulnerability Discovery Techniques: A Survey". In: *Proceedings of the 2012 Fourth International Conference on Multimedia Information Networking and Security*. MINES '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 152–156. ISBN: 978-0-7695-4852-4. DOI: 10.1109/MINES.2012.202. URL: <http://dx.doi.org/10.1109/MINES.2012.202>.
- [12] Adam Back Matt Corallo. *Loom Network sdk for developers*. [Online] Available: <https://medium.com/loom-network/loom-sdk-for-developers-using-an-indexing-layer-for-lightning-fast-dapp-performance-b17f8ba25a3c>. Accessed May 31, 2018.

- [13] Robert F. Service. “Design for U.S. exascale computer takes shape”. In: *Science* 359.6376 (2018), pp. 617–618. ISSN: 0036-8075. DOI: [10.1126/science.359.6376.617](https://doi.org/10.1126/science.359.6376.617). eprint: <http://science.sciencemag.org/content/359/6376/617.full.pdf>. URL: <http://science.sciencemag.org/content/359/6376/617>.
- [14] *Residential Rates*. [Online] Available: <https://app.bchydro.com/accounts-billing/rates-energy-use/electricity-rates/residential-rates.html>. Accessed June 06, 2018.
- [15] Christian Sillaber and Bernhard Walzl. “Life Cycle of Smart Contracts in Blockchain Ecosystems”. In: *Datenschutz und Datensicherheit - DuD* 41.8 (2017), pp. 497–500. ISSN: 1862-2607. DOI: [10.1007/s11623-017-0819-7](https://doi.org/10.1007/s11623-017-0819-7). URL: <https://doi.org/10.1007/s11623-017-0819-7>.
- [16] Stephen Wolfram. *Computational Law, Symbolic Discourse and the AI Constitution*. [Online] Available: <http://blog.stephenwolfram.com/2016/10/computational-law-symbolic-discourse-and-the-ai-constitution/>. Accessed July 07, 2018.
- [17] *EOS Emergency Actions*. [Online] Available: <https://bitcoinexchangeguide.com/eos-undergoes-emergency-actions-after-block-producers-freeze-accounts-without-community-input/>. Accessed June 26, 2018.

General References

- [18] *Bitcoin mining*. [Online] Available: <https://www.investopedia.com/terms/b/bitcoin-mining.asp>. Accessed May 27, 2018.
- [19] Daniele Magazzeni and McBurney, and William Nash. “Validation and verification of smart contracts: a research agenda”. English. In: *COMPUTER* 50.9 (Sept. 2017), pp. 50–57. ISSN: 0018-9162. DOI: [10.1109/MC.2017.3571045](https://doi.org/10.1109/MC.2017.3571045).
- [20] *What are sidechains*. [Online] Available: <https://hackernoon.com/what-are-sidechains-1c45ea2daf3>. Accessed May 31, 2018.
- [21] M. Niranjnamurthy, B. N. Nithya, and S. Jagannatha. “Analysis of Blockchain technology: pros, cons and SWOT”. In: *Cluster Computing* (2018). ISSN: 1573-7543. DOI: [10.1007/s10586-018-2387-5](https://doi.org/10.1007/s10586-018-2387-5). URL: <https://doi.org/10.1007/s10586-018-2387-5>.
- [22] Guido Governatori et al. “On legal contracts, imperative and declarative smart contracts, and blockchain systems”. In: *Artificial Intelligence and Law* (2018). ISSN: 1572-8382. DOI: [10.1007/s10506-018-9223-3](https://doi.org/10.1007/s10506-018-9223-3). URL: <https://doi.org/10.1007/s10506-018-9223-3>.

A. Code Listings

Listing 1: Smart contract in Solidity

```
// pragma is used to specific version of Solidity
pragma solidity ^0.4.16;
contract HelloWorld {
    uint256 counter = 5;
    //state variable we assigned earlier
    address owner = msg.sender;
    //set owner as msg.sender
    function add() public { //increases counter by 1
        counter++;
    }
    function subtract() public { //decreases counter by 1
        counter--;
    }
    function getCounter() public constant returns (uint256) {
        return counter;
    }
    function kill() public { //self-destruct function,
        if(msg.sender == owner) {
            selfdestruct(owner);
        }
    }
    // Fallback function
    function () public payable {

    }
}
```