Center for Cyber Security and International Relations Studies

Smart Contracts

February 2018

Written by: Ottavio Calzone

Abstract

The term *smart contracts* was coined to denote the implementation of contractual logic in computer systems. One of the goals of smart contracts is to create trustless mechanisms of exchange. Using a smart contract, the parties do not need to trust each other or rely on a neutral trusted third party: the computer system acts as guarantor. An example of a trustless mechanism of exchange is Darkleaks, a black market of confidential files based on the Bitcoin's blockchain technology. The convergence with blockchain has given new life to the development of smart contracts, changing however its original meaning. The present work shows the convergence between smart contracts and blockchain, and introduces the resulting new model of smart contracts highlighting the strengths and weaknesses of the model.

Keywords

Smart contracts, Blockchain, Darkleaks, Trustless exchanges, Ethereum

Table of Contents

1. Introduction

The writing of a contract often follows an "if ... then ..." logic: if event x occurs, action y must be taken. It is a type of reasoning that can easily be translated into algorithms and for this reason, as early as 1996, the term *smart contracts* specified computer systems that stored and managed contracts. By implementing contractual logic in a computer system, it is possible to create trustless mechanisms of exchange: in order to execute a transaction, the parties are not required to trust each other or rely on a neutral trusted third party.

An example of a trustless market system is Darkleaks, a black market of reserved files, such as government documents and unreleased movies. This trustless exchange system is based on the Bitcoin network to prevent two opportunistic behaviours:

- the seller receives the payment but does not send the purchased file;
- the buyer receives the asset from the seller but does not send the payment.

Blockchain technology, the heart of the Bitcoin cryptocurrency, fixes two main problems of smart contracts:

- one party changes the code to abort or modify the execution of the contract;
- one party tamper with the hardware running the smart contract to abort its execution.

The convergence of smart contracts and blockchain has modified the original meaning of smart contracts: nowadays smart contracts are programs, not necessarily implementing business or legal contracts, stored and executed by a distributed blockchain[1].

This article introduces the reader to the concepts of blockchain and trustless exchanges, highlights the influence of those concepts on the development of the new model of smart contracts, the technical implications and the strengths and weaknesses of the model[2].

2. Contracts and smart contracts

A contract is an agreement between two or more parties or institutions upon a set of actions to be done in the future as consequence of specific events[3]. Usually in business contracts one party promises to do something in the future in exchange for a payment from another party. A neutral third party registers the contract, certifies if the events under the contract have occurred and the agreed obligations are respected.

A contract often follows an "if… then…" logic: if event *x* occurs, action *y* must be done. It is a logic easily translatable into algorithms and for this reason, as early as 1996, the computer scientist Nick Szabo imagined[4] hardware and software systems used to store contracts, verify the conditions expressed in them and execute the agreed actions. He called these systems *smart contracts*, considering them more functional, more *smart*, than traditional contracts, based on laws, customs and institutions such as courts and notaries.

An example of a smart contract, given by Szabo[5], is the vending machine, where a commercial exchange is governed by a hardware and software system that stores the products, checks the choice of the buyer, collects and verifies the correct inserted amount of money and dispenses the selected product. Another example of a smart contract is Darkleaks[6], in which the business logic governing an exchange is formalized in and managed by a distributed ledger technology system.

Even if the business logic of a contract is often easily translatable into algorithms, implementing it in a computer system is not always an easy task. An action like "check if it is raining at time *x*" requires decisions such as: from what source to take the time, where to check if it is raining or not, how to define the

[1] With *blockchain*, unless otherwise specified, we refer to a distributed blockchain shared by multiple nodes of a network.
[2] The last access to the links reported in the document was done on 03/01/2018.
[3] For a legal definition of contracts: http://legal-dictionary.thefreedictionary.com/contract.
[4] Szabo, N. (1996).
[5] Szabo, N. (1996).
[6] The source code of Darkleaks can be downloaded from https://github.com/darkwallet/darkleaks.

concept of rain by the amount of water falling to the ground. Decisions that can be complicated by having multiple measurement instruments. If we have two clocks marking different hours or instruments reporting different weather conditions, we must be able to establish, inside the computer system, which information to take as good and which to discard.

Once verified if certain events have occurred, the computer system must be able to execute or monitor the execution of the set of agreed actions under the contract. For example, if a password should be sent after a payment, the system has to verify that the payment was successful and then has to manage the communication of the password.

To sum up, a smart contract, in the original meaning given by Nick Szabo, is a hardware and software system which has a representation of the states of the world, stores a contractual logic in computer code, checks and executes what it is under the contract performing actions on the world.

In a business contract managed by a computer code it is important, once the code to be executed has been established, to prevent two fraudulent behaviours:

- one party changes the code in order to abort or modify the execution of the contract;
- one party destroys or tampers with the system running the contract to abort its execution.

These two possibilities cannot occur if the code is stored and executed by all the nodes of a distributed ledger technology network:

- the code can no longer be modified after inserting it in the blockchain;
- to avoid the execution of the code it is necessary to destroy or temper with all the nodes of the network.

Even if the Bitcoin blockchain network has a scripting language[7], specific platforms such as Ethereum[8] have been developed to allow the storage and execution of code on a distributed ledger technology system. Their development has changed the original meaning of smart contracts. Today a smart contract is a computer code, which does not necessarily translate the logic of a legal or business contract, stored and executed on a distributed ledger technology system.

3. Blockchain and trustless exchanges

The Bitcoin network[9] is based on *blockchain* or *distributed ledger technology*. Blockchain is a technology developed to allow electronic exchanges of an economic value expressed in a unit of measure shared by the network itself. This measure of value is called "cryptocurrency". The economic value of a cryptocurrency is created by markets[10] where the cryptocurrency is traded for other assets, such as currencies and gold. Therefore, a cryptocurrency acts as a kind of electronic letter of credit exchanged on a peer-to-peer network working on a distributed ledger technology system.

Distributed ledger technology is a special database in which new information is added to the previous information in a chain of chronologically ordered blocks. Therefore, new information does not modify or

[7] Bartoletti, M. Pompianu, L. (2017).
[8] https://ethereum.org.
[9] Sathoshi, N. (2009).
[10] For example, Coinbase.com, https://www.coinbase.com.

delete the previous stored information and the database keeps all pieces of information stored from the moment of its creation. For this reason, a synonym of distributed ledger technology is blockchain.

The name distributed ledger technology highlights the fact that a blockchain is shared and replicated on multiple nodes of a network. Each node stores and updates its copy of the ledger (a register of economic transactions)[11]. Cooperation and competition mechanisms between nodes allow the register to be kept updated without a trusted third party checking that all the nodes keep the same information.

The information stored in a distributed ledger technology system is:

- *time persistent*, because stored in a chronological ordered blockchain in which a new piece of information creates a new block;
- *space persistent*, because replicated on multiple nodes;
- *independent from an authority*, because based on a behaviour shared by different parties.

These tree proprieties, and the possibility to add text messages to the economic transactions, allow the website Eternitywall.it[12] to use the Bitcoin network as a "virtual wall" and let people write text messages on it that will be stored forever in the blockchain.

The information stored in a block is identified by date and time: each block records when it has been added to the chain[13]. Hence it is possible to have a *proof of existence* of a file: providing undeletable evidence of the existence of a file, in a particular format, at specified time[14]. Using a *hash*[15] function it is possible to generate a unique code of the file and then add it to a Bitcoin transaction. In this way the transaction and the time it occurred will be kept forever in a specific block of the blockchain. It is not possible to modify this information so we can conclude that the file existed in that particular version (the same used to generate the hash code) at the time recorded in the blockchain[16]. A website using this mechanism to offer the service of proof of existence is Proofofexistence.com[17].

The examples of Eternitywall.it and Proofofexistence.com show the possibility of using blockchain platforms for purposes not strictly concerning the exchange of virtual money. Other systems integrate additional functionalities with the exchange of e-money. Darkleaks uses the Bitcoin network to run a black market of reserved files, such as government documents and unreleased movies. Darkleaks prevents two opportunistic behaviours in an online transaction between parties that do not know each other:

- the seller receives the payment but does not send the purchased file;
- the buyer receives the asset from the seller but does not send the payment.

It is possible to download Darkleaks source code from GitHub[18]. By using Darkleaks, the seller uploads the file to sell. The file is split by Darkleaks into *n* parts, each of which is the input of a hash function. From each generated digest (the hash code output of the hash function) the system creates a Bitcoin address and the related public and private key. The address and the two keys will be associated with the part of the file

[11] The node map of the Bitcoin network can be found at https://bitnodes.21.co.
[12] https://eternitywall.it.
[13] To analyse the information contained in the Bitcoin network chain block, check the site *Blockchain.info*, https://blockchain.info.
[14] The possibility that the file existed before the proof of existence is not excluded, but not assured.
[15] For an explanation of the hash functions refer to the Appendix.
[16] The complete procedure for creating a proof of existence is shown in the video available at https://youtu.be/qn13TIhoYUY.
[17] https://proofofexistence.com.
[18] https://github.com.

used to generate them. Another key, used by the system to encrypt the related segment of the file, is then calculated from the public key using the hash function.

The Bitcoin address is used to receive and send the e-money and can be public. The private key is used to authorize money transfers from the related address and therefore must be kept secret. The public key is used by the network, and will be known by all the nodes when the e-money is sent from the related address.

The seller decides the *m* number of the *n* segments to be automatically decrypted by the system at a specific time, which is also determined by the seller. At the specific time Darkleaks randomly decrypts *m* segments of the file. These decrypted segments can be verified by the potential buyers.

The seller is paid at the Bitcoin addresses linked to the parts of the file not yet decrypted. When this e-money is used, the network will know the public keys of those addresses and every member of the network will be able to generate the keys used to encrypt the segments.

Working in this way, Darkleaks allows the buyer to verify the file to be bought and ensure that the seller will not be able to use the money without revealing the segments of the file being exchanged. At the same time, the seller knows that the payment will be received for what is being sold.

Darkleaks, however, has its limits[19]. In particular:

- *absence of time constraint* – the seller can wait to reveal the file using the money received only after the file has lost its value, perhaps because it has already been released by another source;
- *no privacy of the keys* – once all the public keys of the segments are known to the network, the file can be decrypted not only by the buyers but also by other users of the network;
- *fragmentation in the revelation of the keys* – the transfer of funds from only some addresses will mean that only some segments of the file will be decrypted;
- *determinism in keys generation* – the legitimate owner of the file can recreate the set of keys, uploading it in *Darkleaks*, and therefore, even if the sale cannot be avoided, possession can be taken of the payments sent to the Bitcoin addresses (using the private keys).

Despite its limitations, Darkleaks implements the logic of a business transaction in a computer system. Moreover, the exchange is trustless: the parties are not required to trust each other or rely on a trusted third party to execute the transaction.

The trustless nature of Darkleaks lies in its mission as a means of exchange and in the technology used, since the Bitcon network is based on the trustless cooperation of nodes. This double trustless nature, purposive and technical, is also at the core of the development of the idea of smart contracts.

4. A new model of smart contract

Today a smart contract is a computer code, which does not necessarily translate the logic of a legal or business contract, stored and executed on a distributed ledger technology system.

The definition just given of a smart contract is technical and not purposive as the original given by Nick Szabo: it emphasizes the technical characteristics that a smart contract must have and not the goals it must achieve. However, the technical characteristics have functional implications.

[19] Juels, A., Kosba, A. and Shi, E. (2013),

First of all, the execution of a code stored in a blockchain takes place on all the nodes of the network. The execution of a program transforms information into new information on the basis of the program code. In our case the output must be the same for all nodes to allow all the copies of the blockchain to continue to store the same shared history. For this reason, smart contracts cannot directly access information sources outside the network[20]. The code is in fact executed by all the nodes and the execution could take place at different times with the possibility of finding different pieces of information outside (e.g. at one moment it is raining and five minutes after it is not raining anymore). In these cases, the nodes would not produce the same output. This observation has general validity: if a smart contract directly performed an action in the external world, the same action would be performed by all the nodes of the network (but not at the same time).

To overcome this problem, smart contracts interact only within the blockchain; it is then the world outside the blockchain that writes or reads information on it and behaves accordingly. For example, an external service can send a piece of information to a specific smart contract[21]. To know this piece of information the other smart contracts in the blockchain will query this specific smart contract, which thus acts as an "oracle[22]".

Passive interaction with the outside world makes the input the same for all nodes. The code stored in the blockchain is also identical. Therefore, to get an identical output it is sufficient that all the nodes execute the code in the same way. This goal can be achieved by equipping each node with the same virtual machine: an environment that simulates, on different platforms, the same computer.

The space persistency, typical of distributed ledger technology systems, not only requires the use of some constraints to ensure the production of identical output on all the nodes of the blockchain, but also implies the public nature of the shared information[23]. In many practical applications it is desirable to have some level of anonymity and privacy. To do this, the solution suggested by some researchers[24] is to execute part of the code outside the blockchain, altering in this way the concept of smart contract.

The persistence of information in space is not the only element that characterizes the distributed ledger technology. Another main characteristic is time persistency. The fact that what is already included in the blockchain cannot be deleted or changed is a critical factor for the storage and execution of software code[25]. In fact, someone could store a malfunctioning or intentionally malicious smart contract in the blockchain. A typical case is a program with an infinite cycle: once activated, the code would continue forever to use computational resources. To avoid this problem, it is possible to exploit one of the characteristics of the blockchain: the ability to manage virtual money. If we define that for the execution of a smart contract it is necessary to consume a defined amount of cryptocurrency, on the basis of the computational resources and memory required by the code, we will be able to ensure that the execution of the code will stop. This is a solution used, for example, by Ethereum[26].

The need to spend money to execute the computer code implies an economic decision by an agent. The execution of a smart contract is therefore reactive and not proactive[27]: smart contracts have to wait for a call made by an external agent, physical person or other smart contract, willing to consume economic resources in exchange for the execution of the code.

[20] Greenspan, G. (2016).
[21] Bartoletti, M. Pompianu, L. (2017).
[22] "Oracles" is the name used also for the services pushing information on the blockchain.
[23] Greenspan, G. (2016).
[24] Kosba, A. et al. (2016).
[25] In Ethereum a smart contract cannot be deleted or modified once inserted in the blockchain but it is possible to insert in its code a clause to make it no longer functional when certain conditions occur.
[26] Tommasicchio, A. (2017).
[27] Rikken, O. (2017).

Since the execution of a smart contract is reactive and requires e-money, every smart contract should be able to manage the blockchain cryptocurrency to call another contract stored on the blockchain, for example to ask for information from an oracle.

To sum up, nowadays smart contracts broadly mean computer code stored on and executed by a distributed ledger technology system. This definition is technical but has some functional implications mainly related to the persistence in time and space of the information stored on a distributed blockchain. As a consequence of these implications, a smart contract is a program:

- which can be activated on call;
- which will eventually stop;
- which interacts using blockchain transactions;
- able to manage virtual money and information;
- able to call other smart contracts that can act as oracles;
- stored and executed on a distributed blockchain shared by a set of network nodes.

5. Strengths and weaknesses of the new model

Smart contracts allow the limits of Darkleaks to be overcome. If the problem of password determinism can be resolved using the Bitcoin network, the other highlighted[28] limits can be solved by creating a system similar to Darkleaks through Ethereum smart contracts[29] implementing constraints in the flow of money and information, such as the possibility of retrieving the funds sent if the keys used to encrypt the segments of the files being exchanged are not released by a certain date.

The ability to manage information and virtual money allows a smart contract to act as an economic agent or to work as a financial instrument[30]. For example, it is possible to imagine a smart contract working as a piggy-bank, making it possible to withdraw money only after an established date. Imagining an application like this, it is necessary to remember the characteristics of the technology. The smart contract will be public, and so there could be privacy issues. It will also be immutable: an error in the code could mean losing all the cryptocurrency stored in the piggy-bank. It would be like a malfunctioning vending machine that doesn't deliver the chosen product after inserting the right amount of money. Unlike the vending machine, for which it is theoretically possible to contact an authority responsible for it, one of the characteristics of the blockchain running a smart contract is the absence of an authority.

This also impinges on another aspect: when the virtual money is managed by a smart contract, who is the owner of the cryptocurrency[31]? The concept of authority makes it possible to give custody of the money to a third party, such as a bank, remaining however the owner of the money and continuing to have the right of carrying out actions based on that property. Possibilities which in the case of smart contracts are meaningless.

The weakness of smart contracts seem related to some characteristics of the distributed ledger technology, in particular, space persistency and time persistency of the stored information and its independence from an authority. For example, we can develop a smart contract working as a lottery[32] using its capacity to manage

[28] Absence of time constraint, no privacy of the keys, fragmentation in the revelation of the keys.
[29] Juels, A., Kosba, A. and Shi, E. (2013).
[30] Greenspan, G. (2016).
[31] Greenspan, G. (2016).
[32] Brown, R. G. (2015).

money and information. Once created and stored in the blockchain, if the lottery becomes illegal somewhere, it would not be possible to eliminate or modify the smart contract (time persistency). Furthermore, the smart contract will be stored and executed on all nodes of a distributed ledger technology system (space persistency) so it could be legal for some nodes operating from a jurisdiction and illegal for other nodes from another jurisdiction where the lottery is legal. Finally, as a result of the absence of an authority, who is responsible for the smart contract and the lottery? The programmer who created the code and stored it? The person who uses it? The owner of the computer on which runs a node of the distributed blockchain?

The strengths related to smart contracts seem to depend on the same aspects. Smart contracts could be used to build an accounting database[33], shared between a customer and a supplier, that does not need to be reconciled and that is able to execute some clauses if and when certain events occur. The accounting history is publicly shared between the parties (space persistency), perfectly traceable in its historical evolution (time persistency) and not dependent on a third party that could modify its content (independence from an authority).

6. Conclusions

The computer scientist Nick Szabo coined in 1996 the term smart contracts to indicate computer systems storing and managing business and legal contracts. One application of smart contracts is to create trustless mechanisms of exchange, in which the parties do not need to trust each other or rely on a trusted third party. However, using computer code to manage and execute a contract it is necessary, once the code to be executed has been established, to prevent two possible fraudulent behaviours:

- one party changes the code in order to abort or modify the execution of the contract;
- one party destroys or tampers with the system running the contract to abort its execution.

These possibilities are eliminated if the code is stored in a blockchain and executed by all the nodes of the distributed ledger technology network, because:

- the code can no longer be changed after inserting it in the blockchain;
- it is necessary to destroy or tamper with all the nodes of the network to avoid the execution of the code.

The combination of these three elements – smart contracts, distributed blockchain and trustless transactions – has given new life to the idea of smart contracts, changing however its original meaning. Today smart contracts mean computer code stored and executed on a distributed blockchain. This definition is technical but has functional implications mainly related to the time persistence and space persistence of the information stored in a distributed ledger technology system. A smart contract is therefore a program:

- which can be activated on call;
- which will eventually stop;
- which interacts using blockchain transactions;
- able to manage virtual money and information;
- able to call other smart contracts that can act as oracles;
- stored and executed on a distributed blockchain shared by a set of nodes of a network.

[33] Greenspan, G. (2016).

The strengths and weaknesses of the smart contracts seem to depend on some characteristics of the distributed ledger technology, in particular the time persistence and space persistence of the stored information and the independence of such information from an authority. These three characteristics, together with the ability to manage money and information, allow, among other actions, the creation of trustless markets for the exchange of confidential documents or shared accounting databases in which the execution of certain clauses by the computer code is based on the occurrence of specific events.

Appendix. Hash functions

Hash functions calculate a fixed-length string of characters from a message, document, or sequence of characters. The generated code is the *message digest*, also called *digital fingerprint* or just the *hash* of the message. In the creation of the hash code, the function guarantees:

- *effectiveness and speed*: all the participants in the network can easily calculate the digest of a message and quickly obtain the same result;
- *preimage resistance*: it is not computationally possible to calculate the message that generated a digest;
- *collision resistance*: it is not computationally possible to find two different messages that generate the same digest.

The website *Hash.online-converter*[34] allows to calculate the message digest of a file or a message. By applying, for example, the SHA-256 algorithm to the text "this is only a test" we get the code:

1661186b8e38e79f434e4549a2d53f84716cfff7c45d334bbc67c9d41d1e3be6.

We can verify that by adding a single letter at the end of the input text, e.g. writing "this is only a testt", we get a completely different code:

727910c03657899a3fef8cf7d56243728edf4b98099a02599a8333fb64b7025c.

References

Bartoletti, M. and Pompianu, L. (2017), *An empirical analysis of smart contracts: platforms, applications, and design patterns*, https://arxiv.org/pdf/1703.06322v1.pdf.

Brown, R. G. (2015), *A Simple Model for Smart Contracts*, February 10, 2015, https://gendal.me/2015/02/10/a-simple-model-for-smart-contracts.

Cudi, Z. (2015), *Introducing Darkleaks*, in «Medium», January 28, 2015, https://medium.com/@ZozanCudi/darkleaks-information-blackmarket-1ee5ac28c892.

Greenspan, G. (2016), *Why Many Smart Contract Use Cases Are Simply Impossible*, Coindesk, April 17, 2016, https://www.coindesk.com/three-smart-contract-misconceptions.

[34] *Hash.online-converter* (algorithm SHA-256), https://hash.online-convert.com/sha256-generator.

Juels, A., Kosba, A. and Shi, E. (2013), *The Ring of Gyges: Using Smart Contracts for Crime*, http://www.arijuels.com/wp-content/uploads/2013/09/Gyges.pdf.

Kosba, A et al. (2016), *Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts*, IEEE, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7546538.

Rikken, O. (2017), *Smart contracts: 3 leggende sui contratti intelligenti (3 legends on smart contracts)*, Etherevolution, February 3, 2017, https://etherevolution.eu/leggende-metropolitane-smart-contracts.

Satoshi, N. (2009), *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2009, https://bitcoin.org/bitcoin.pdf.

Szabo, N. (1996), *Smart Contracts: Building Blocks for Digital Free Markets*, in «Extropy #16», 1996, http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html.

Tommasicchio, A. (2017), *Ethereum Gas Price: il vero costo delle transazioni (the real cost of transactions)*, Etherevolution, March 24, 2017, https://etherevolution.eu/ethereum-gas-price.