International Journal of Information Security (2023) 22:1445–1480
https://doi.org/10.1007/s10207-023-00700-3

REGULAR CONTRIBUTION

Malicious uses of blockchains by malware: from the analysis to Smart-Zephyrus

Mar Gimenez-Aguilar[1] · Jose Maria de Fuentes[1] · Lorena Gonzalez-Manzano[1]

Published online: 12 May 2023

Abstract
The permanent availability and relative obscurity of blockchains is the perfect ground for using them for malicious purposes. However, the use of blockchains by malwares has not been characterized yet. This paper analyses the current state of the art in this area. One of the lessons learned is that covert communications for malware have received little attention. To foster further defence-oriented research, a novel mechanism (dubbed Smart-Zephyrus) is built leveraging smart contracts written in Solidity. Our results show that it is possible to hide 4 Kb of secret in 41 s. While being expensive (around USD 1.82 per bit), the provided stealthiness might be worth the price for attackers.

Keywords Blockchain · Covert channels · Malware

1 Introduction

Blockchain is a well-known technology that allows the execution of transactions ensuring their integrity. It can be described as "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way" [1]. It has been a growing technology since Satoshi Nakamoto proposed it in 2008 [2]. A plethora of applications have been devised, starting by cryptocurrencies like Ethereum [3].

Due to their anonymity and availability, blockchains are also used for malicious purposes [4]. For example, Bitcoin is widely used in the darknet to pay for forbidden products [5]. Similarly, malware developers and cybercriminals have already used this technology as well [6–9].

As a massive phenomenon, the relationship between malwares and blockchains is worth being studied. It must be noted that malwares have infected at least one-third of the computers worldwide [10] and their estimated global impact is $6 M per year [11]. Previous works have studied this relationship. Kshetri and Voas [12] analysed the effect of cryptocurrencies on ransomware. Similarly, Böck et al. [13] provided a systematic analysis of blockchain-based botnets. However, most of the previous studies focus on a single type of malware (e.g. botnets or ransomware) or a single type of blockchain technology (e.g. Ethereum or Bitcoin).

To overcome this limitation, this paper provides with an in-depth study of how malwares are using blockchains. We consider different perspectives such as the malware purpose, the blockchain elements at stake or the cost for the attacker. This study leads to the identification of research gaps, some of which are also addressed in this paper through a mechanism, called Smart-Zephyrus, for covert communications in blockchains. As they are typically hard to detect, this could be very useful for malwares. Thus, it should foster further defence-based research in this direction.

Therefore, the contributions of this paper are as follows:

- Study and categorization of the different ways in which blockchains have been used by malwares.
- Development of Smart-Zephyrus, a mechanism to carry out cover channel communication leveraging Ethereum smart contracts. Its development involves a large amount of real-world smart contracts to ensure the stealthiness of the secret. Besides, an open-source proof of concept is publicly released.

✉ Jose Maria de Fuentes
jfuentes@inf.uc3m.es

Mar Gimenez-Aguilar
mgimenez@inf.uc3m.es

Lorena Gonzalez-Manzano
lgmanzan@inf.uc3m.es

1 Computer Security Lab (COSEC), Universidad Carlos III de Madrid, Madrid, Spain

Paper organization Sect. 2 gives the background to understand the proposal. Section 3 describes the research methodology. The analysis on how malwares have leveraged blockchains is presented in Sect. 4. The proposed covert channel mechanism is introduced in Sect. 5, and assessed in Sect. 6. The related work is presented in Sect. 7. Lastly, Sect. 8 concludes the paper and points out future work directions.

2 Background

In this section, the main concepts of blockchains, covert channels and malwares are introduced. In particular, the foundations of blockchains are presented in Sect. 2.1. The definition of a covert channel is introduced in Sect. 2.2. Zephyrus, a steganographic tool for Ethereum, is described in Sect. 2.3. Finally, different types of malware are explained in Sect. 2.4.

2.1 Blockchains

Blockchain technologies enable having a distributed ledger in which data are appended [2]. One important matter is that there is no need for a single, centralized trusted party—trust is distributed among all nodes. Therefore, in order to add data to the ledger, a consensus is usually reached among all (or a qualified portion of) involved nodes [14].

Blockchains provide a set of properties by default. Besides, a blockchain can be classified based on its underlying technology, as well as the elements used. Each of these issues is introduced below.

Properties

- Immutability. It is the ability of the ledger to remain unchanged, unaltered and inedible [15]. This property provides some benefits, like complete data integrity and auditability. However, this characteristic has some limitations, namely the 51% attack—an adversary takes control of the majority of the nodes, allowing him to change the transaction data [16].
- Decentralization. These networks aim to reduce the level of trust among participants. Some benefits of decentralization include trustless intermediation, reducing points of weaknesses and optimizing resource distribution [17].
- Data availability. Data can be accessed and used when needed [17].
- Pseudo-anonymity. Anonymity can be defined as the situation in which someone's identity is not given or known [18]. Although some blockchain technologies are anonymous (e.g. Monero), most of them are pseudo-anonymous. The user has a public address that could be traced back to an exchange account or IP address via network analysis. Thus, under certain circumstances, it is possible to reveal the user's real identity [19].

Technology

One common use of blockchains is cryptocurrencies. They can be understood as systems in which tokens are used as a general or limited-purpose medium of exchange.

Bitcoin was the first cryptocurrency. It was proposed by Satoshi Nakamoto in 2008, and it allows two parties to send transactions between each other without the necessity of a third party. Its non-Turing-complete scripting language supports different advanced features, like the use of timelocks to prevent the execution of a given action before the time runs out.

On the other hand, Ethereum was released in 2015. Beyond transactions, it allows the execution of smart contracts. These are programs that are executed by Ethereum nodes through their Ethereum virtual machine. They are written in object-oriented programming languages like Solidity or Serpent, compiled into bytecode [3] and stored on chain. As most object-oriented languages, Solidity includes functions, which have assorted input arguments [20, 21]; modifiers, that restrict the behaviour of certain functions; variables; and events, that allow publishing information about something in the chain. Smart contracts work as decentralized applications. Example of applications built with smart contracts include trading, investing, gaming or voting. Indeed, there are smart contracts called tokens which represent a variety of transferable and countable goods such as digital and physical assets, shares, votes, memberships or loyalty points. The most widely used token is, by far, Ethereum's ERC20 [22], followed by others like ERC721, or ERC165. Such smart contracts can be found in websites like OpenZeppelin, which is an open-source platform for building secure distributed applications [23].

Other technologies and cryptocurrencies have been developed. For example, Monero uses privacy-enhancing technologies (ring signatures and stealthy addresses) that obfuscate transactions to achieve anonymity. Dash is an alternative currency that was forked from the Bitcoin protocol [24]. It includes other improvements such as InstantSend, which allows instant transaction confirmations without a centralized authority [25]. Other cryptocurrencies, just to mention a few, are BitcoinCash (Bitcoin fork), Emercoin and Litecoin.

Elements

Blockchains are a distributed database shared among nodes. The data are usually recorded in the form of transactions. When a transaction is recorded in the blockchain, details of the transaction are recorded, verified and distributed across all nodes [26]. Recorded data can vary among technologies, though some data are common in all of them. First, there is the sender address which produces the transaction. Depending on the technology, there may be one or more receiver addresses which are intended to receive the transaction. If the technology is a cryptocurrency, a field that represents the transferred amount or value is usually included. Some technologies also allow the addition of arbitrary data in the blockchain. This is the case of the data field in Ethereum [27], OP_RETURN transaction type in Bitcoin [28] or payment id in Monero [29].

Moreover, some technologies also allow the execution of programs in the blockchain, for example scripts in Bitcoin or smart contracts in Ethereum, as introduced before.

2.2 Covert channels

A covert channel is "any communication channel that can be exploited by a process to transfer information in a manner that violates the system security policy" [30]. A covert channel has different properties:

- Stealthiness: the capacity of avoiding detection. Thus, data sent or received through a covert channel cannot be easily discovered by an undesired party.
- Efficiency: the amount of useful information that can be sent using the covert channel in relation to the cost or total amount of data sent through the channel.
- Secret integrity/resilience: the capacity of hidden information to resist against changes.

In most cases, the goal is to maximize the amount of shared information (efficiency). Though it may impact stealthiness, a balance between both issues should be considered. On the other hand, in the case of the blockchain, as its content on chain is immutable, resilience is provided by default.

A typical technique to implement covert communications is steganography [31]. It consists of concealing a message or object within another that is used as a container for the covert channel. Its main strength is that the message or object goes unnoticed. The steganographic message can be coded inside of a transport layer, like an image, document, program or protocol.

2.3 Zephyrus

Zephyrus is a steganographic tool for Ethereum [32]. While most tools and previous examples of steganography on blockchain were based on Bitcoin, this technology explores Ethereum as a way to embed secret messages. It considers three uses cases: the panic button one, when a threatened individual is willing to release information in an emergency; the sabotage case when an attacker wants to exfiltrate data; and the censorship case when an individual wants to share information in a censored environment. It allows inserting secret messages in 8 different fields of Ethereum, like the receiver address or the value field. Regarding contracts, it hides information in the swarm hash or bytecode, as well as in the function arguments and constructor. The message can be split and encrypted before insertion. However, it is not able to use the high level language (i.e. Solidity) in which contracts are written.

2.4 Blockchain-related malware types

Malware can be defined as any software designed to cause intentional damage to a computer, server, client or network [33]. They can be classified in different types, depending on the goal of the attacker and the way it acts. Despite the amount of malware types, this section introduces those which are known to be related with blockchains.

In recent times, one of the most popular types of malware is ransomware. It prevents users from accessing their system or files and demands a payment as ransom in order to regain access [34].

On the other hand, botnets leverage the decentralization provided by blockchains. A botnet is a network of infected computers that can be remotely controlled and forced to perform different actions or attacks without the consent of the device owners [35]. Orders are given by a command and control (C&C) server, that is a computer controlled by an attacker to send commands to compromised systems, called bots, and receive stolen data from them. Bots can carry out assorted tasks such as hacking, spying or interrupting a service [36]. A botmaster is a person who operates the C&C server for remote process execution [37]. Note that a botnet is not considered a malware by itself but as they can be used for malware propagation and because of their use for malicious purposes, herein we considered botnets within malware types, in line with [38, 39].

On the other hand, a computer worm is a computer program that replicates itself to spread to other computers afterwards. It often relies on security failures on the target device to access it [40].

3 Research methodology

To study how different malwares use the blockchain and identify research gaps, the applied methodology starts by searching for relevant research papers (both journal and conference/workshop papers) in Google Scholar, as well as in relevant web articles or projects.

The following query has been developed to filter out relevant contributions based on their title:

(blockchain AND malware) OR (blockchain AND botnet) OR (blockchain AND ransomware)

The query above ensures that the main terms are considered, even in different forms. After this step, a total of 168 proposals were retrieved. Then, a manual review was carried out. This ensures that those papers that are not relevant for the sample (e.g. other literature surveys) or that do not contain any use of blockchain in malware (e.g. cryptominers) are filtered out. After this analysis, the sample is definitely formed by 104 proposals.

The identified proposals are studied in detail, classifying them according to different features. In the case that a certain proposal fits into more than one category per feature, (e.g. possibility of use different technologies), it is counted in each of them. The following features are analysed in each proposal.

- Type of malware. The analysis of the malware type related to blockchains may provide information to defenders for a better understanding of how to react against such malicious programs.
- Blockchain technology. As each one exhibits different features (recall Sect. 2.1), it may help defenders to identify which ones are present (e.g. real anonymity if Monero is at stake) to counter a threat.
- Desired properties. As mentioned in Sect. 2.1, blockchains provide some properties by default. Knowing which ones are relevant for malwares may be helpful for defending against them.
- Used blockchain elements. The analysis of the used blockchain elements (recall Sect. 2.1) provides a deeper understanding of how the malware operates, which might enable an early detection.
- Seed address. If a malware is using a blockchain, an address is needed for such interaction. How this address is delivered to the malware could help defenders prevent this communication taking place.
- Data protection. Blockchain data can be protected to provide confidentiality, or even secrecy with covert channels. In this way, the malware communication stealthiness is studied in order to provide a better understanding and early detection of possible hidden communications.
- Goal of using blockchains. Studying the use of blockchain for malicious purposes helps defenders understand what they can expect or search.
- Cost for the attacker. Whether there is a cost for the attacker and how much an attack would cost are important matters in terms of proposals' feasibility. If the cost of using a given approach is too high, its feasibility can be negatively affected.

These features, depicted in Fig. 1, are used as the basis of the following analysis. Numbers in brackets represent the amount of proposals in each category.

4 Malware and blockchain analysis

How malwares are being used leveraging blockchain is studied herein (recall Sect. 3). For the sake of clarity, the analysis is structured following the dimensions shown in Fig. 1.

In particular, Sect. 4.1 discusses the blockchain properties relevant for malware. Section 4.2 discusses the communication features. The set of blockchain elements at stake and the data protection issues are described in Sects. 4.3 and 4.4, respectively. The purpose, malware type, blockchain technology and cost to the attacker are presented in Sects. 4.5, 4.6, 4.7 and 4.8. The summary of the analysis is addressed in Sect. 4.9.

4.1 Desired properties

Properties provided by blockchain in malware proposals are studied in the following:

- Availability. Blockchain information is always available, although accessibility is limited by the nature of the blockchain (private and public). In our study, 25 works of the sample aims to exploit this characteristic. For example [6, 41, 42], intend to provide availability.
- Immutability. Content, once mined, is very difficult to change as an attacker needs to be in control of 51% of the nodes of the network (recall Sect. 2.1). Only 5 proposals use this feature (e.g. [41–43]).
- Decentralization. As a distributed ledger, blockchain is decentralized by design and some level of decentralization is provided in all works.
- Anonymity. Authors use the blockchain with the intention of providing pseudo-anonymity in 99 works (e.g. [6, 7, 44]) and real anonymity in 2 of them [45, 46].

According to this study, blockchains are mainly used to provide some level of decentralization, thus making more difficult to locate and take a malware down, and anonymity/pseudoanonymity, to hide data origin. Immutability of the content is the least used property. As an example, [42] exploits this feature—a semi-autonomous ransomware is developed, and its code in the smart contract cannot be changed.

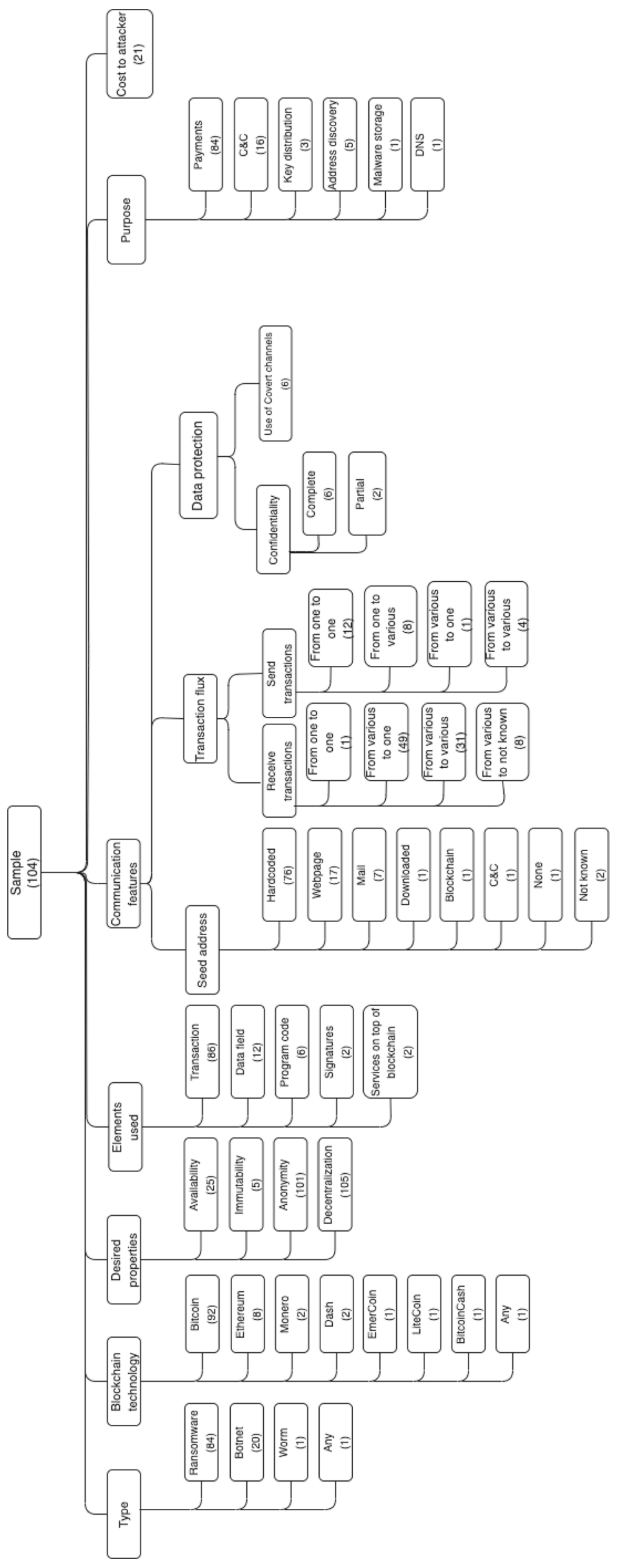

Fig. 1 Proposals analysis

4.2 Communication features

In this section, the analysis on how the seed address is known by the malware, as well as the transaction flux between the malware and the blockchain, are discussed.

4.2.1 Seed address

The seed address can be the one receiving payments [7, 44, 47]. It can also be used to manage the malware, for example, by using smart contracts [42]. There is only one case in which there is no seed address, as the system checks the validity of every OP_RETURN transaction posted in the blockchain [48].

The way the seed address is delivered can be classified as follows:

- Hardcoded. The address is written in the malware itself, for example in a file or in the code. Most of the seed addresses are hardcoded (76 proposals) such as in [7, 44] or [8].
- Webpage. The address is delivered via web when the client or victim accesses a website [47, 49, 50]. Delivery via website is adopted in 17 works, being the second most common method.
- Mail. The address is delivered via email, e.g. [51, 52] or [53]. This is the third most common category with 7 cases.
- Blockchain. The address is extracted from the blockchain itself. This happens with all addresses except for the first one in [40].
- C&C server. The address is delivered via a C&C server. Each time an address has been used, a component in charge of monitoring the health of the C&C server, sends the new address to this server. It then sends the address to bots [54]. 1 work uses this method.
- Downloaded. The file which contains the address is downloaded after infection, being [55] the only one in this category.
- Unknown. How the address is delivered to the malware is unknown. Delgado-Mohatar et al. [42] and Hurtuk et al. [45] are the ones that fall in this category.

4.2.2 Transaction flux

This section focuses on the communication patterns between the malware and the blockchain. In particular, this communication will take place by means of sent or received transactions.

Transactions can be sent from one or multiple sources (e.g. unique attacker address or different attackers) to one or various victims (e.g. smart contract). Most transactions are sent from one to one (12 works), as in [56] where the attacker only interacts with the smart contract address. The same happens in [43] where a unique attacker address per victim is created. After that, the second most common flux is from one (the attacker) to various (victims) with 8 proposals of this kind. For example [57, 58]. Moreover, there are 4 works [43, 46, 54, 59] in which there are either various attackers or various addresses used by the attackers to communicate with a set of victims. For instance, in [60] the author of the ransomware, or an individual renting its services, communicates with the smart contract handling the ransomware, thus being a communication following the 'various-to-one' pattern.

Regarding received transactions, most of them are from various to one (49 works). They are mostly ransomwares in which various victims pay the ransom to a unique attacker address (e.g. [7, 61]). This category includes those works whose attacker address is static for a long time, but it is changed at some point. For example, FakeGlobe [62] changes its address per malware campaign. The second largest group are those in which different clients communicate with a range of attacker addresses (31 proposals). For example, ransomware ZCrypt [55] applies a personalized unique address per victim, and in Locky [62] a wide range of attackers' addresses are found. However, one collector address which sends money back to the attacker address (from one to one) is used in [58]. On the other hand, the number of attacker addresses is not identified in 8 samples. For example, blind addresses are distributed by mail in [51]. Princess Locker [63] is another example in which it is unknown whether it uses a single address.

4.3 Used blockchain elements

This section studies what elements of the blockchain are used in the analysed proposals:

- Transactions common fields. In every blockchain system, there are fields that are usually common. Transactions usually need a sender address, a receiver address (in some of them can be null) and a value to exchange (this one is specially common in cryptocurrencies, recall Sect. 2.1). These fields are applied in 86 proposals, such as in [7, 44, 47].
- Data. Field or type of transaction in which arbitrary data can be included (Sect. 2.1). This has been considered apart from transaction common fields because some technologies (e.g. Bitcoin) only include this field in specific transaction types. This is the case of 12 proposals [41, 43, 48].
- Program code. A program hosted in the blockchain (e.g. smart contracts in Ethereum) is used in 6 works. They are often used to send commands to the bots of a botnet. For example [42, 56, 64].
- Nonce for digital signatures. Blockchain usually uses elliptic curves to generate signatures linked to transactions. In this process, a secure nonce is generated per message for later use in the computation of the curve. This value is modified in a pair of proposals [65, 66]. As it happens with the program code, the nonce is usually used as a way to send commands to bots.
- Services on top of blockchain. EmerDNS is used to provide a DNS service over EmerCoin and [9] uses it. Whisper is a message protocol on top of Ethereum, applied in [67]. Only a couple of works use these services.

4.4 Data protection

This section studies how information is protected by analysing whether it is confidential or if covert channels are used to conceal the data exchanged.

Some proposals need to exchange data in order for the malware to work properly. This data can be transmitted via normal channels or covert channels (recall Sect. 2.2). To evaluate covert channel stealthiness, we adopt the attacker model defined in [32], composed of three different types of attacker. A pair of them are considered passive, inspecting blockchain contents using a block explorer (e.g. Etherscan [68]). While one of them is an eavesdropper (Basic Eavesdropper, BE) and another one might carry out syntactic checks on each transaction (Advanced Eavesdropper, AE), the third one (Interactive Attacker, IA) is active, being able to make transactions. We will also consider a special case of BE, BE* (Basic Eavesdropper with a simple hiding technique) when there is no hiding technique in place, so the message will be easily spotted by anyone.

Data posted in the blockchain can be either in clear or hidden in some way, e.g. encrypted. For example, Curran and Geist [54] and Yin et al. [69] both use RC4 to encrypt data transmitted to the blockchain. It can also be obscured by sharing only hashes of the data [56].

Table 1 shows the maximum capacity (in bytes) of each proposal in which data are exchanged in the blockchain; which element is used to exchange the information; and the attacker model applied in terms of stealthiness in case they use a covert channel. There is some kind of information exchange in the blockchain in 24 works, while the remaining ones do not use this technology for that end.

According to the table, OP_RETURN in Bitcoin, the data field in Ethereum and Payment id in Monero are actually used to insert data in the blockchain. Because of that, very low stealthiness is considered as it is a field which usually contains messages (e.g. [43, 46, 69]).

On the other hand, some works use function arguments (e.g. [42, 64, 70, 71]). They do not hide the information in any way—data are exchanged using the corresponding argument type. For example, normal text is shared by using string arguments, hashes and keys by using bytes32, etc. Because of this, their stealthiness is considered low (BE*). Furthermore, their capacity usually depends on how many arguments each function has and which type they are (recall Sect. 2.1), categorized as Argument-related limit (ArL).

Hiding information as a receiver address is a common way to disguise it (e.g. [72, 73]) as, in general, no transformation is applied to the data, so reading information from the address is trivial. For these reasons, although they do use a covert channel, their stealthiness is considered low.

Formatting the data as the value field in the transaction (e.g amount of transferred Ethers or Bitcoins) has also been used in [74]. This is less common and often some transformation is applied previously to convert information to a suitable value. Their stealthiness is considered medium as the message is not easily readable, but usually a simple transformation from integer to hexadecimal allows to retrieve the message.

Whisper messages [67] are encrypted and private. However, all the members know there is a message being exchanged even though they cannot decrypt it. Finally, embedding the message in the nonce of the signature is first proposed by Frkat et al. [66], and data are difficult to distinguished from normal transactions. Furthermore, information is also obscured, so even if it is retrieved, it is difficult to extract the message. Therefore, its stealthiness is considered high.

4.5 Purpose

Different proposals use the blockchain with different goals in mind. The following categories are distinguished:

- Payments. Works use the blockchain as a way to send payments, generally to the attacker. There are 84 proposals with this purpose, such as [7, 44] or [47].
- C&C. The blockchain is used to control bots and send instructions to them. Platdrag [56], Frkat et al. [66] and Yin et al. [69] use the blockchain as a C&C server. This happens in 16 proposals.
- Address discovery. The blockchain is used to provide the malware with new sources of information/commands, generally to new C&C servers after a takedown [48, 54]. Five works use the blockchain to provide some kind of data source.
- Key distribution. The blockchain is used to distribute keys. For example, keys to decrypt data encrypted by a ransomware, i.e. [42, 43]. Three proposals are included within this category.
- DNS. The blockchain is used as a DNS service in [9].
- Malware storage. Malicious software is stored in the blockchain in one proposal, namely [41].

4.6 Malware type, blockchain properties and elements

Malware type, together with the purpose, blockchain properties and elements are studied herein. Table 2 summarizes this data.

Different kinds of malware use the blockchain (recall Sect. 2.4). Most proposals study ransomwares (84 works), e.g. [6, 7, 44], followed by botnets (20 proposals), e.g. [57, 66, 69]. Worms [40] and any type of malware [41] are barely relevant, with 1 work each.

Concerning purpose, results show that when the malware is a ransomware, the blockchain is mostly used as a way to obtain payments (84 proposals). This happens, for example, in [6, 7, 44]. Key distribution (3 works) is the second purpose most used for this kind of malware [42, 43, 64]. C&C and Address discovery is the goal in 1 work, [6, 64], respectively. Regarding botnets, the main purpose of the blockchain is working as a C&C server (15 proposals), i.e. [56, 66] or [69], followed by distributing addresses (4 works), i.e. [48, 54, 75] or [71]. Besides, DNS is the purpose for 1 proposal [9]. In [41], any type of malware uses the blockchain to store itself. On the other hand, in the work concerning worms, the blockchain is used as a C&C server [40].

In terms of blockchain properties, ransomware mostly takes advantage of the blockchain's decentralization (84 works) and pseudo-anonymity (83 works, such as [7, 44] or [6]), and the same happens with worms and botnets but this latter to a lesser extent (decentralization in 20 works and pseudo-anonymity in 18 proposals). Moreover, in botnets, blockchains are sometimes applied due to their availability (10 works such as [56, 57] or [8]), in contrast to traditional systems, where servers can be easily taken down. By contrast, the paper that is suitable for any kind of malware uses blockchains to provide decentralization, availability and immutability in complete and equal manner. As anything posted in the blockchain cannot be altered without a huge effort (recall Sect. 2.1), it ensures that different parts of the malware remain available and unalterable.

Regarding the blockchain elements used per type of malware, ransomwares mostly use transaction common fields (82 works). They also use program code in 2 proposals and data fields in 1. Botnets, on the other hand, use data fields in 10 of them. Program code are used in 4, transaction common fields in 3, the nonce of the signatures as well as services on top of the blockchain in 2. Worms use transaction common fields and the proposal that can use any malware uses the data one.

4.7 Blockchain technology

Most proposals use Bitcoin (93 works) such as [44] or [6]. Ethereum is the second most used one, with 8 works, i.e.

Table 1 Data protection and cost

Source	Max. capacity (bytes)	Confidentiality	Field	Attacker model	Cost for the attacker	Current value(dollars)
[41]	80	No	OP_RETURN	BE*	0.008 btc	164.63
[6]	6	No	Receiver address	BE	0.10 btc	2057.89
[42]	64	No	Function arguments	BE*	0.002582 eth	3.79
[43]	80	No	OP_RETURN	BE*	Free	0
[64]	ArL	No	Function arguments	BE*	0.069062 eth	101.25
[48]	80	No	OP_RETURN	BE*	0.00000546 btc	0.11
[56]	20,000	Obscured	Function arguments	BE*	0.0113092 eth	16.58
[66]	32	Obscured	Nonce of signature	ALL	Not known	–
[69]	80	Encrypted	OP_RETURN/Data field	BE*	0.00000226 btc	0.047
[75]	80	Encrypted	OP_RETURN	BE*	0.00000546 btc	0.11
[70]	ArL	No	Function arguments	BE*	Not known	–
[59]	20	No	Value & Receiver address	BE,AE	0.00000001 btc	0.0002
[57]	80	No	OP_RETURN	BE*	0.0000675 btc	1.39
[76]	ArL	No	Function arguments	BE*	0.0004725 eth	0.69
[54]	80	Encrypted	OP_RETURN	–	0.00000546 btc	0.11
[8]	2	No	Value	AE	Not known	–
[71]	ArL	No	Function arguments	BE*	Not known	–
[67]	64,000	Encrypted	Whisper message	BE*	Free	0
[58]	1	No	Value	AE	0.00000154 btc	0.031
[65]	80.3	No	OP_RETURN & signature	BE,AE	0.001 btc	20.57
[46]	32	Encrypted	Payment id	BE*	0.00006 xmr	0.009
[77]	80	No	OP_RETURN	BE*	0.00000546 btc	0.11
[78]	80	No	OP_RETURN	BE*	0.00001897 btc	0.39
[79]	50,000	Encrypted	OP_RETURN	BE*	Free	0

*Simple hiding technique

Table 2 Malware-type summary

	Ransomware	Botnet	Worm	Any
Purpose				
Payments	84	0	0	0
Key distribution	3	0	0	0
Address discovery	1	4	0	0
C&C	1	15	1	0
DNS	0	1	0	0
Malware storage	0	0	0	1
Blockchain properties				
Decentralization	84	20	1	1
Pseudo-anonymity	83	18	1	0
Availability	4	10	0	1
Immutability	3	1	0	1
Blockchain elements				
Transaction common fields	82	3	1	0
Data fields	1	10	0	1
Nonce of signature	0	2	0	0
Program code	2	4	0	0
Services on top of blockchain	0	2	0	0
Total	84	20	1	1

[64] or [56]. Dash [47, 69] and Monero [45, 46] are used in 2 works each of them, while BitcoinCash, Emercoin and Litecoin are applied in a single proposal [69]. On the other hand, in [66] any type of blockchain technology can be used.

Blockchain technologies in relation to malware purposes, type and blockchain properties and elements are analysed. Table 3 shows a summary.

In Bitcoin, the blockchain is mostly used for payment purposes (80 proposals), such as in [7, 44]. Moreover, blockchain is also used as a C&C server in 8 proposals (e.g. [40, 69]), to provide addresses to the system in 5 works (e.g. [48] or [75]) and just in one proposal Bitcoin is used for key distribution [43] and malware storage [41]. In contrast, Ethereum is applied for payment purposes and key distribution in 2 proposals [42, 64], while it is significantly used as a C&C server (7 works), probably because of the possibility of using smart contracts (recall Sect. 2.1), e.g. [56] or [70]. Besides, Dash and Monero are equally used as a mean to pay (1 proposal) [45] and to establish a C&C server (1 work) [46]. Emercoin is used to provide a DNS service by using EmerDNS [9]. Litecoin and BitcoinCash [79], as well as [66] in which any technology can be used, are exclusively applied as C&C server.

Regarding desired properties, those works which use Bitcoin look for the provision of decentralization and pseudo-anonymity (93 and 91 works, respectively). However, availability is demanded in 9 proposals and immutability in 2 of them. Similarly, Ethereum main purposes are to provide decentralization (8 works), as well as pseudo-anonymity (6 works), availability (5 proposals) and immutability (3 works), although this latter to a lesser extent. In the case of Monero, real anonymity is provided in a couple of works, as well as decentralization and availability just in one. Dash, EmerCoin, Litecoin and Bitcoin cash just focus on decentralization and pseudo-anonymity provision.

Used blockchain elements and technology are also jointly studied. In Bitcoin, common transaction fields are used in the majority of proposals (83 of them), while the data field is used in 11 and the nonce of the signature just in 1. In Ethereum, however, most of the works use program code (6 proposals) and data field and services on top of blockchain are used just in 1 each. Dash and Monero both use the data and transaction common fields equally (1 work). Emercoin uses services on top of blockchain and Litecoin and BitcoinCash both use data fields.

By type of malware, Bitcoin is mostly used by ransomwares (80 proposals), followed by botnets (12 works). Ethereum, on the other hand, is mostly used for botnets (6 works) and ransomware (2 works) cases. Dash and Monero are the least relevant as they are applied in only one proposal for ransomware and botnet. The rest of technologies are used for botnets.

4.8 Cost for the attacker

The attacker usually has to assume some cost, specially in those cases in which transactions are sent to the blockchain because they contain some kind of information.

Table 3 Blockchain technologies summary

	Bitcoin	Ethereum	Dash	Monero	BitcoinCash	Emercoin	Litecoin	Any
Total	93	8	2	2	1	1	1	1
Purpose								
Payments	80	2	1	1	0	0	0	0
Key distribution	1	2	0	0	0	0	0	0
Address discovery	5	0	0	0	0	0	0	0
C&C	8	7	1	1	1	0	1	1
DNS	0	0	0	0	0	1	0	0
Malware storage	1	0	0	1	0	0	0	0
Blockchain properties								
Decentralization	93	8	2	2	1	1	1	1
Pseudo-anonymity/Anonymity	91	6	2	2	1	1	1	1
Availability	9	5	0	1	0	0	0	0
Immutability	2	3	0	0	0	0	0	0
Blockchain elements								
Transaction common fields	83	0	1	1	0	0	0	0
Data field	11	6	1	1	1	0	1	0
Nonce of signature	1	0	0	0	0	0	0	1
Program code	0	6	0	0	0	0	0	0
Services on top of blockchain	0	1	0	0	0	1	0	0
Malware types								
Ransomware	80	2	1	1	0	0	0	0
Botnet	12	6	1	1	1	1	1	1
Any	1	0	0	0	0	0	0	0
Worm	1	0	0	0	0	0	0	0

Table 1 shows the different costs incurred in each case. In order to compute the cost, the mean (average) of the value of each coin from the last three months (from December 10, 2022 to March 10, 2023) has been considered: 1 ether is $1,466.03 [80], 1 xmr (Monero) is $157.907 [81] and 1 btc is $20,578.9 [82]. For those which use OP_RETURN method and do not indicate the cost, the cost of the minimum transaction value is considered [83].

Even considering previous costs, most proposals are relatively cheap, with the exception of [6] which spends 0.10 btc ($2,057.89 currently) and [64] which spends 0.069062 eth ($101.25). There are some of them, which are actually free, such as [43], where a fee to the victim is asked first and the transaction the attacker sends reuses this money, so there is no extra cost for the attacker. Moreover, Baden et al. [67] applied a service on top of the Ethereum blockchain, named Whisper. This service does not send traditional transactions and exchanging messages is always free. On the other hand, Franzoni et al. [79] uses the Bitcoin testnet, which, among other things, is cost free as Bitcoin can be obtained from public faucets (an app or a website that distributes small amounts of cryptocurrencies as a reward for completing easy tasks) [84].

As a result, the benefits of using blockchain are higher than the cost for attackers. For example, Cerber [6] ransomware generated $2.3 million in annual revenue. Profit when using botnets is also high, for example Methbot was making $3–5 million a day on its peak [85]. Indeed, given that traditional attacks may also involve some cost [86], e.g. for using or maintaining servers (an hour-long DDoS attack using a cloud server will cost criminals $7 [87]), the use of blockchain seems to be affordable.

4.9 Summary of the analysis

This section presents the main findings of the analysis (Sect. 4.9.1) and the identified open research issues (Sect. 4.9.2).

4.9.1 Lessons learned

According to the data study, the main findings are:

- Bitcoin is the most used technology. It is followed by Ethereum as the second most common technology.

- Malware proposals do not use the full potential of the blockchain. Some level of decentralization is always provided but some characteristics which are intrinsic to the blockchain like immutability, availability or anonymity are not used at full in most cases.
- Most seed addresses are hardcoded. If the address is flagged or the private key is lost or stolen, the malware becomes useless.
- Blockchain transactions are preferred as the way for malwares and blockchains to interact. Very few use smart contracts or other services on top of the chain.
- There is not much exchange of information on the chain. The blockchain is barely used to share data and when applied, data is almost never hidden or covert. When used, it is typically for sharing command information to a botnet.
- Blockchain primary use is for payments. In relation to the previous point, blockchain is mostly used as a mean to provide payments to the attacker by a high majority of proposals.
- The most common type of malware is ransomware. This is related to the previous point, as it is used to pay ransoms.
- It is cheap to use the blockchain. The cost for the attacker is usually cheaper using the blockchain compared to traditional attacks, which also involve some costs. For example, a DDoS attack using botnets is estimated to cost from $50 to several thousand dollars in the case of a 24-h operation [88].

4.9.2 Research gaps

Based on the previous analysis, the following research gaps have been identified:

- Covert channels are barely used. This entails that communications between attackers and victims are relatively easy to identify. Although their contents are not always understandable due to encryption, the existence of the communication can be discovered.
- Smart contracts have been applied in malware, but not used for sophisticated covert channel purposes. Information is usually formatted as the corresponding arguments types in functions in the contracts, allowing trivial data retrieval. Furthermore, high level languages like Solidity have not been used to insert information.
- Blockchains have not been used in well-known malware like Trojans, spyware or keyloggers. These are malware types that can take advantage of the blockchain suitability to transfer information and its availability and immutability.

5 Smart-Zephyrus

At the light of the identified research gaps, a new steganographic mechanism for Ethereum that uses a smart contract programming language (Solidity) as a covert channel is proposed. It is called Smart-Zephyrus. Remarkably, contracts with hidden content can be used as normal ones, so there is no need for them to be dedicated only to the purpose of exchanging information.

The motivation of building a covert channel leveraging blockchains is of direct interest for attackers. If communications are hidden (e.g. to share a key, to send a command or to receive exfiltrated data), it reduces the chance of being intercepted.

This section starts by introducing an overview (Sect. 5.1). The preliminary analysis on existing smart contracts is addressed in Sect. 5.2. Lastly, the mechanism design is presented in Sect. 5.3.

5.1 Overview

Smart-Zephyrus works over Zephyrus (recall Sect. 2.3). However, instead of only using EVM/bytecode elements of the smart contracts, a high level programming language to embed information is used. Figure 2 shows a comparison between Zephyrus and our current proposal, Smart-Zephyrus. While Zephyrus embeds information on blockchain raw data, Smart-Zephyrus allows embedding information in a high-level programming language (Solidity). This language is then compiled and transformed to contract bytecode to be sent to the blockchain. In order to retrieve the Solidity code, the contract code needs to be verified in a blockchain explorer, which checks that the Solidity code and the bytecode deployed on chain match.

To embed information in smart contracts without raising suspicions, a study is initially carried out (Sect. 5.2). We have retrieved a sample of contracts to mimic them. Once their features are analysed, the selection and design of contracts to embed information is performed. As explained later, contracts will be built by altering different issues of a pre-existing one, such as the amount of contracts to inherit (called parent contracts) or the types of libraries they use, to name a few. Two different versions of the mechanism are developed, one more statistically similar to the studied sample, and, thus, stealthier, and another one that allows more capacity, but less similar to the studied data.

After that, a practical experiment has been carried out to validate the similarity of the smart contracts produced by our mechanism to the studied sample. Furthermore, the time and cost of sending and retrieving different sizes of hidden messages is also analysed.

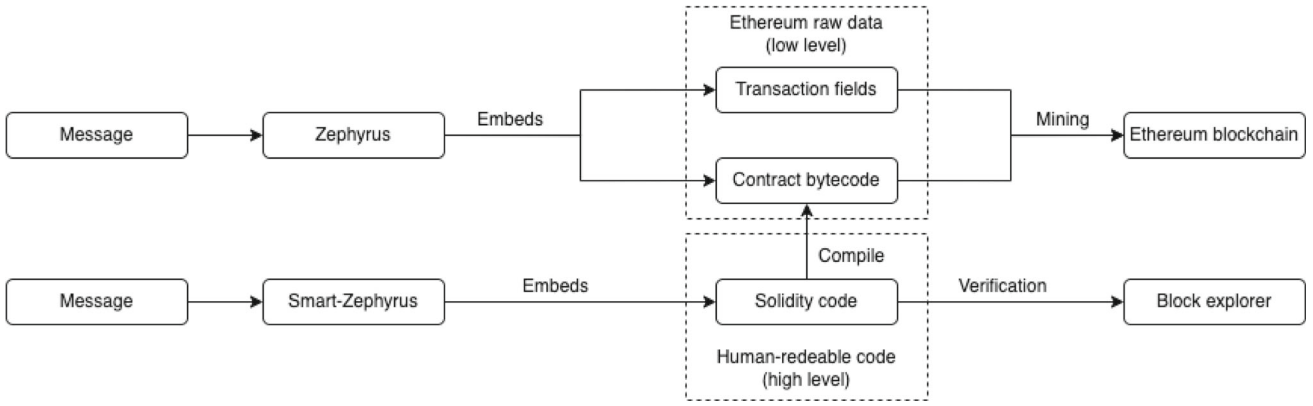

Fig. 2 Zephyrus versus Smart-Zephyrus

5.2 Preliminary smart contract study

A study of Ethereum smart contracts is firstly carried out to generate stealthy smart contracts afterwards. These contracts were downloaded from the Ethereum blockchain. A total of 103,106 contract files are analysed (available on Gitlab[1]). Within those files, it is possible to find numerous contracts, interfaces and libraries, which have relations and dependencies among them.

Contracts in Solidity are similar to classes in object-oriented languages. Interfaces are one kind of contracts that do not contain any logic, just definitions. Libraries, on the other hand, contain logic that can be later referenced.

After processing the smart contracts, the following findings are highlighted. As shown in Table 4, most contracts belong to OpenZeppelin. Ownable is intended to allow the transfer or withdrawal of a contract ownership, so it has little real-world application by its own. On the contrary, ERC20 is a well-known token—a blockchain-based asset that can be traded. Therefore, Smart-Zephyrus will leverage ERC20 token contracts.

To ensure the representativeness of the considered ERC20 contracts, 7143 of them were retrieved from [89]. To the best of authors' knowledge, it is the biggest dataset in this regard. Those whose Solidity code was available (6632 contracts) were analysed according to the ERC20 standard peculiarities. It must be noted that their code is used for embedding purposes. The results of the analysis are shown in the left column of Table 6. Thus, the number of contracts, libraries and interfaces per file are characterized. Moreover, the prevalence of the order of functions and contracts, as compared to the original ones (i.e. the version released by its creator), is measured. Lastly, different ERC20-related issues such as its token name and symbol are characterized. For example, a large variety of names have been found. The most common word in the name was "Token" in 6.83% of the cases, and 13,197 pairs of words have been found.

Table 4 Top 10 smart contracts

Name	Appearences
Ownable	28,292
ERC20	21,227
Context	19,426
StandardToken	11,835
ERC20Basic	9748
BasicToken	7350
Token	5341
Pausable	5109
Owned	5070
ERC20Interface	4972

5.3 Mechanism design

In this section, the design of Smart-Zephyrus is described. For this purpose, the design choices are introduced in Sect. 5.3.1, whereas its different operation modes are presented in Sect. 5.3.2.

5.3.1 Design decisions

Based on the findings explained in the previous section, some design decisions for the construction of steganographic smart contracts have been taken.

On the one hand, the base contract will be ERC20 due to its popularity (recall Sect. 5.2). On top of this choice, it will inherit or use a number of contracts, will utilize a set of libraries and will provide with some interfaces. All these decisions will be inspired by our preliminary study. Thus, the number of chosen contracts (besides ERC20) the token contract will either use or inherit from will be 2, and the

[1] https://gitlab.com/MarGA2503/retrieved-contracts.

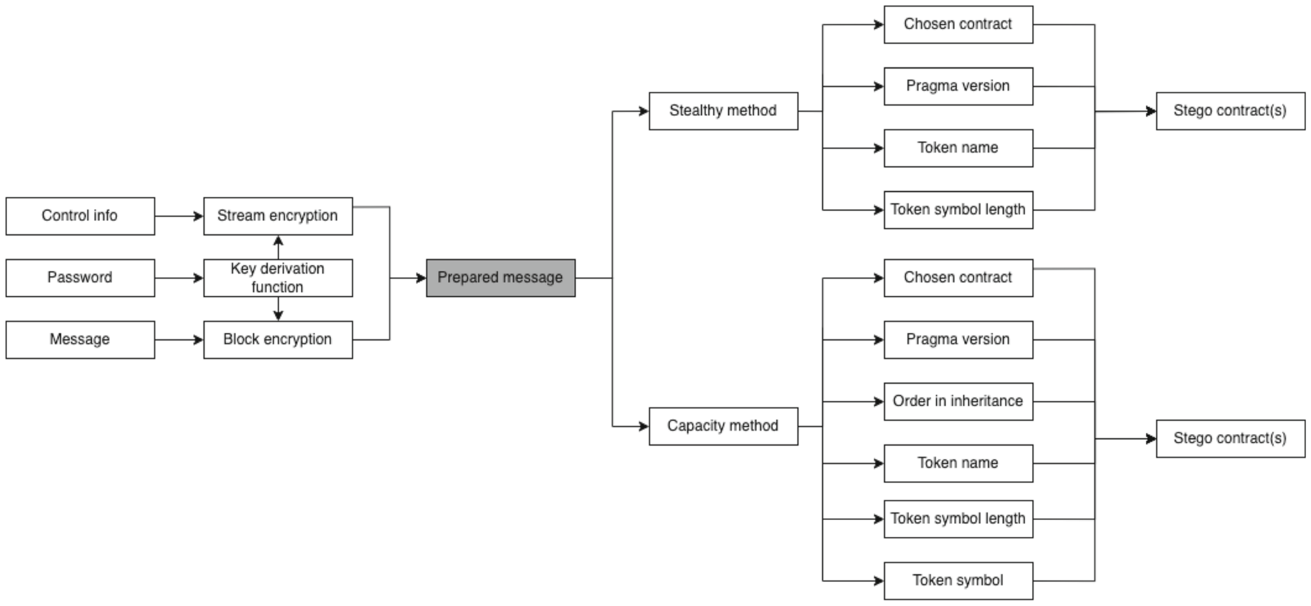

Fig. 3 Smart-Zephyrus main steps

amount of libraries and interfaces will be adapted to the actual choice of contracts.

In what comes to the selection of contracts, libraries and interfaces must be compatible among them. They must also be valid for the current Solidity compiler (Pragma 0.8) and should not produce inheritance loops. Thus, the set of contracts to choose from are Ownable, Pausable, ERC20Burnable, ERC20Capped, ERC165, Reentrancy Guard, ERC20Permit, TokenTimelock and ERC20Snapshot. All of them belong to OpenZeppelin [23], in line with our analysis, to promote stealthiness.

5.3.2 Data insertion mechanism: operation modes

In this proposal, two modes of operation will be defined considering different levels of capacity and stealthiness:

- Stealthy. Mechanism with lower capacity but more accurate and similar to the studied values extracted from original contracts.
- Capacity. Allows more capacity, but it sacrifices stealthiness making the steganographic contract more different from legitimate ones.

Figure 3 shows the main steps of Smart-Zephyrus. The message is symmetrically encrypted in first place. Thus, confidentiality is provided making difficult any statistical analysis and the possibility of an entity to spot the malicious communication. Control information, required in the revealing process, is also included and encrypted, but with a stream cipher to reduce length at a minimum. Finally, data are embedded according to proposed operation modes, producing one or more smart contracts with portions of the secret.

There are different ways in which information is embedded:

- Chosen contract encoding. Each contract (except from ERC20 which will always be present) will be assigned a value (3 bits). For example: Contract0=0, Contract7= 7. Once Contract0 is chosen, it is removed from the list, and Contract8 is added and values are assigned again. Contracts are ordered by appearance in the studied (i.e. real-world) sample, meaning that the most common contracts have a bigger chance of being chosen.
- Order of the contract in inheritance. A contract can inherit from different contracts. The order in which these contracts appear in the token child contract inheritance definition can be used to insert information. For example if a contract inherits from Ownable and Pausable, they can be ordered in two ways. Each combination is then assigned a value to insert information. As a majority of contracts preserve that same order (recall Sect. 5.2), this will only be used in the Capacity mode.
- Pragma version. It goes from 0.8.0 to 0.8.17 (by now) so $\log_2 18 = 4$ bits can be used for embedding.
- Token name. In order to embed as much information as possible, a dictionary has been used to give a number for each word. The Word Game Dictionary[2] was adopted. In particular, $\log_2 266{,}336 = 18$ bits were used per word in the Capacity mode. For the Stealthy mode,

[2] https://www.wordgamedictionary.com/sowpods/download/sowpods.txt.

all words shorter than 4 were removed (as they were infrequent in token names, only 13.64% of them). This leads to $\log_2 11{,}202 = 13$, bits per word. In any case 2 words appear in token names in line with our observations (recall Sect. 5.2).
- Token symbol length. 3 and 4 characters are chosen, allowing 1 bit for embedding information.
- Token symbol characters. This is only used in the Capacity version. Considering the target length, a substring of that size is produced based on all character combinations from the selected token name. Embedding is thus done by assigning a value to each combination.

Let nph be the number of permutations of parent contracts and ncl the combinations without repetition of the token name characters. Table 5 shows a summary of the different methods used to embed information and their capacity.

6 Assessment

The proposed approach is assessed by creating a list of smart contracts and confronting them against real-world ones. Furthermore, the cost and time for the attacker are also measured.

The experimental settings are described in Sect. 6.1. Afterwards, the statistical comparison against existing contracts is shown in Sect. 6.2. Lastly, the measurements on cost and time are presented in Sect. 6.3.

6.1 Experimental settings

Experiments have been run in an Intel Core i7-1165G7 processor equipped with Debian WSL for Windows 10 with 16 Gb. of RAM. A proof-of-concept implementation of Smart-Zephyrus is publicly released to foster further research.[3]

Note that the mining process is not part of our system, and Smart-Zephyrus works in any computer with similar characteristics and once installed Python 3.8 (or higher) and used libraries. Concerning the blockchain, Sepolia has been used to carry out the experiments because it is the one recommended by Ethereum.org as the default testnet for application development [90] and the rest of the testnets are currently deprecated. Addresses have been provided with enough funds to carry out all transactions and Infura nodes have been used to connect to the blockchain. To ensure the validity of our results, each embedding and revealing operation related to network usage in Sect. 6.3 has been carried out 5 times, while for program computation times (embedding/retrieving of the message and ciphering) it has been repeated 250 times, thus ensuring the soundness of results. Note that network time has been limited to 5 repetitions for being a time consuming task.

[3] https://gitlab.com/MarGA2503/smart-zephyrus.

Afterwards, the arithmetic mean and the standard deviation have been computed.

6.2 Comparison of studied versus generated smart contracts

First, in order to ensure the mechanism compliance with the studied sample, a comparison among studied contracts and the ones generated is depicted in Table 6. The embedded secret message consists of *lorem ipsum* text encrypted with AES, each time with a different key. The same quantity of smart contracts as in the sample is selected to make a fair comparison.

The number of smart contracts, libraries and interfaces in the generated contract files is really similar to those studied. The highest differences are in the number of contracts, as the mean of the generated samples is 6.05 in both versions versus 5.23 of the existing ones, and in the number of interfaces, with 1.12 of average in the generated contracts as compared to 2.67 in the existing ones.

In what comes to the order and names of functions, the mean of files that include OpenZeppelin contracts as they are (same order and function names) is 95.94% for the same compiler version as our generated contracts. This has been replicated in the output of Smart-Zephyrus—OpenZeppelin contracts will always be present as they can be retrieved from the creator.

Concerning token names, for the stealthy insertion method, the number of unique words is a little less with 48.75% and "Token" appears 2.67% of the times. This is slightly lower than the original sample. Note that the appearance of the word "Token" follows a random distribution which tries to imitate the original one and then, the difference between the generated samples and the original ones may change depending on the status of the applied random generator. On the contrary, the difference is noticeable in the capacity method, in which the number of unique words increases to 94.68%.

Regarding the number of words used in the name, the results between the studied and generated sample are really similar, being the mean of the former 1.76 with a standard deviation of 1.01 and 2.05 and 2.06 with a standard deviation of 0.22 and 0.23 for the stealthy and capacity method, respectively.

The percentage of unique symbols in the studied contracts is 85.62%, while in the generated contracts this percentage is a bit smaller, 77.97% for the capacity version but more similar (86.65%) for the stealthy version.

Something similar happens with the mean and standard deviation of the number of letters of symbols. In the studied sample, the mean is 4.02 letters per symbol, with a standard deviation of 2.04. In the generated contracts, the mean is 3.51 characters per symbol for the stealthy method and 3.50 in the capacity one with standard deviations of 0.49.

Table 5 Steganographic capacity per method

Method	Possible values	Quantity of bits	Type of mechanism
Chosen contract encoding	6	3*2	Both
Order of the contracts in the token inheritance	nph	$\log_2$ nph	Capacity
Pragma version	18	4	Both
Token name	11,202(Stealthy)/ 266,336 (Capacity)	2*13/2*18 (Capacity)	Both
Token symbol length	2	1	Both
Token symbol letters	ncl	$\log_2$ ncl	Capacity

Table 6 Original contracts versus Smart-Zephyrus generated contracts

Feature	Measurement	Original ERC20 contracts	Generated contracts (stealthy mode)	Generated contracts (capacity mode)
Number of contracts	Mean (std) in file	6.05 (16.87)	5.23 (0.42)	5.23 (0.42)
Number of interfaces in file	Mean (std)	1.12 (2.96)	2.67 (0.59)	2.67 (0.59)
Number of libraries in file	Mean (std)	1.39 (3.54)	1.56 (1.58)	1.53 (1.59)
Contracts and functions in the same order as standard (all)	% Mean (std)	70.31 (32.07)	100 (0)	100 (0)
Contracts and functions in the same order as standard (pragma 8)	% Mean (std)	95.94 (5.79)	100 (0)	100 (0)
Token names	% of unique words	55.65	48.75	94.68
Token name lengths	Mean	1.76 (1.01)	2.05 (0.22)	2.06 (0.23)
Token names	% of appearance of the word "Token"	6	2.67	2.93
Token symbols	% of unique symbols	85.62	86.65	77.97
Token symbols lengths	Mean (std)	4.02 (2.04)	3.51 (0.49)	3.50 (0.49)
Token symbols lengths	% of all letters in token name	81.80	81.51	100
Token herency contracts number	Mean (std)	1.49 (1.04)	2.07 (0.54)	2.06 (0.54)
Token herency contracts	% of different order	10.44	5.48	49.24

With respect to the order in inheritance, for the studied sample, only 10.44% of contracts differs in order. For the generated contracts, in the stealthy method this percentage is 5.49%, while for the capacity is 49.24%. This is because the stealthy mode is randomly generated according to the reference value, while in the capacity mode the goal is to embed a significant amount of data.

According to these results, the mechanism generates contracts that are fairly similar to the legitimate contracts already deployed on the chain.

6.3 Experimental results

Experiments are carried out to measure the gas cost, according to the capacity of generated steganographic smart contracts, and the time of the embedding and revealing process. The length of embedded messages is: 256, 1024 and 4096 bits. They are based on common key lengths as a possible way to exchange keys for any kind of malicious purposes.

Fig. 4 Gas cost per method

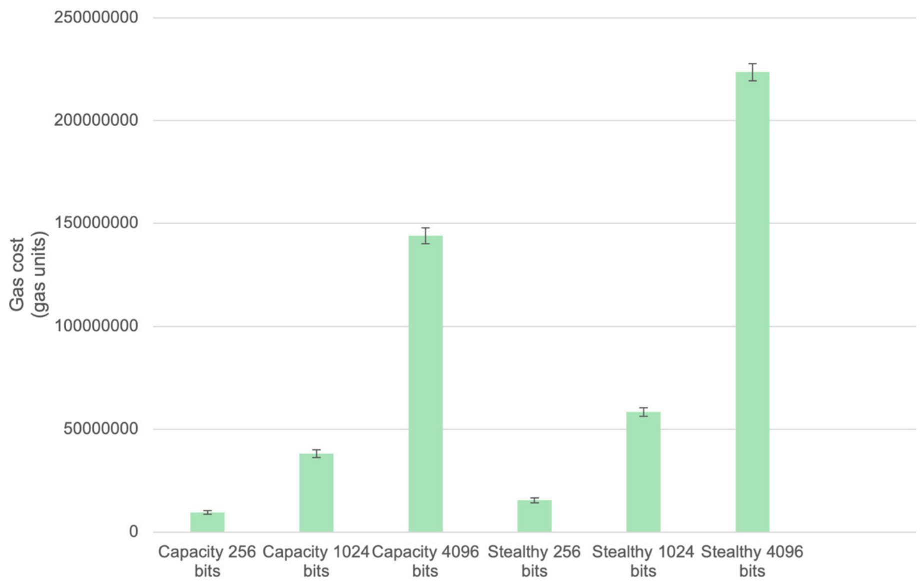

Table 7 Steganographic capacity per method

Method	Gas cost (mean)	Cost in ether	Cost in USD
Capacity 256 bits	9,574,380	0.2606	382.07
Capacity 1024 bits	38,184,016	1.0393	1523.74
Capacity 4096 bits	144,001,955	3.9197	5746.45
Stealthy 256 bits	15,447,935	0.4204	616.46
Stealthy 1024 bits	58,412,600	1.5899	2330.97
Stealthy 4096 bits	223,530,292	6.0844	8920.05

6.3.1 Cost assessment

Concerning the actual costs incurred by Smart-Zephyrus, Fig. 4 shows the gas cost per method depending on the message size. As expected, as it allows less capacity per contract and thus it needs more contracts for the same size, the stealthy method is more expensive than the capacity one. The cost also increases with the secret size. For the maximum secret size (4096 bits), the mean gas cost for the capacity method is 144,001,954.7 gas versus 223,530,291.9 for the stealthy one. Table 7 shows the cost on Ether and USD of each mechanism and secret. In order to calculate the cost on Ether, the average gas price for the last three months (December 10, 2022 to March 10, 2023) [91] has been considered, being 27.22 gwei. On the other hand, 1 ether is \$1466.03 (recall Sect. 4.8).

These costs seem expensive, but there are a couple of issues to take into account. First, traditional attack methods also incur a cost for the attacker [86]. For example, an hour-long DDoS attack using a cloud server costs criminals \$7 [87] and such server could be still taken down if discovered. In this regard, in botnets like Zeus or Mirai, the malware package costs from \$700 up to $<$ \$10K and $<$ \$30, respectively [92]. Besides, while the cost of maintenance in Mirai is unknown, in Zeus is of \$62k [92]. Furthermore, revenue from ransomware is usually higher than these costs. For example, Wannacry has three known Bitcoin addresses with payments of 54.43 BTC [93]. In today's value that is around USD 1,120,109.53 so a cost of USD 8920.05 (our highest) is not much in comparison.

On the other hand, our mechanism provides something previous proposals do not—a high level of stealthiness. According to studied works (recall Sect. 4.4), most of the time the communication is in clear and not hidden, thus traceable and blockable. Furthermore, among those proposals which actually use covert channels to hide information [6, 8, 59, 65, 66], the only proposal that effectively hides information surpassing all models of attackers is Chainchannels [66], but its cost is unknown. Furthermore, it does not use contracts to hide information. Our proposal, on the other hand, imitates fully usable smart-contracts and hides information in a way that makes them indistinguishable from the normal ones. Moreover, information can be encrypted, thus achieving equal stealthiness (ALL—attacker model) to [66] and also providing an estimated cost.

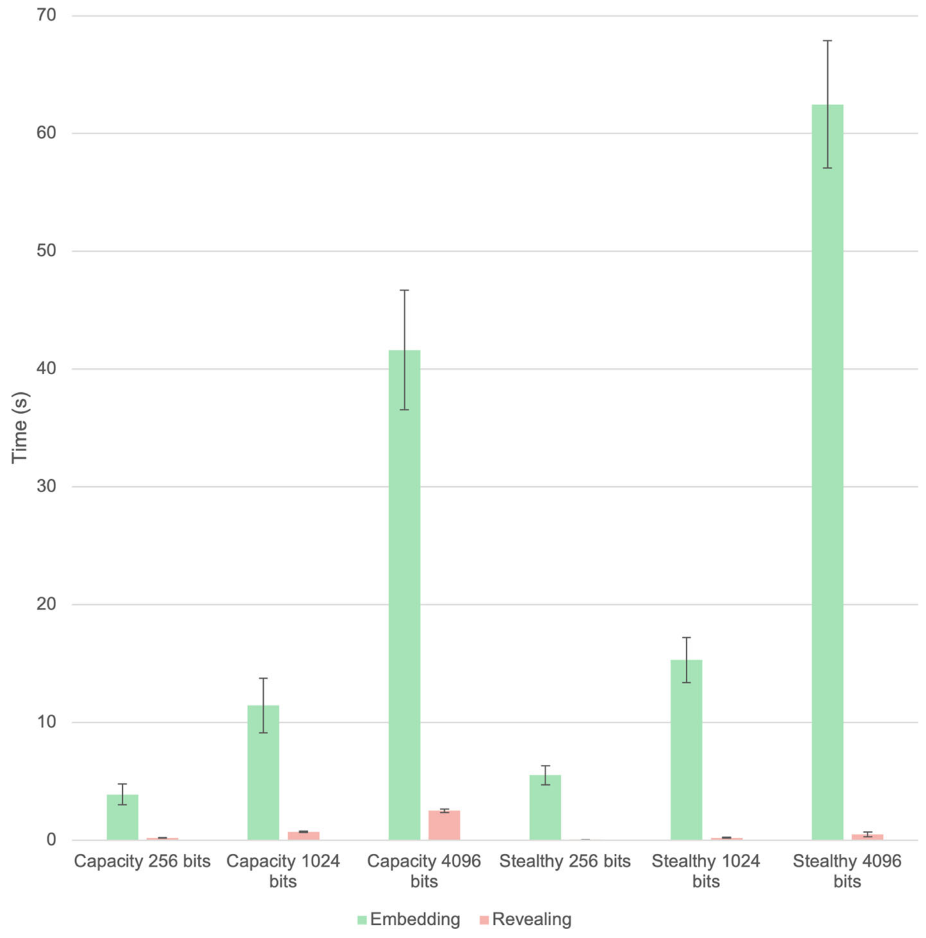

Fig. 5 Embedding and revealing time per method

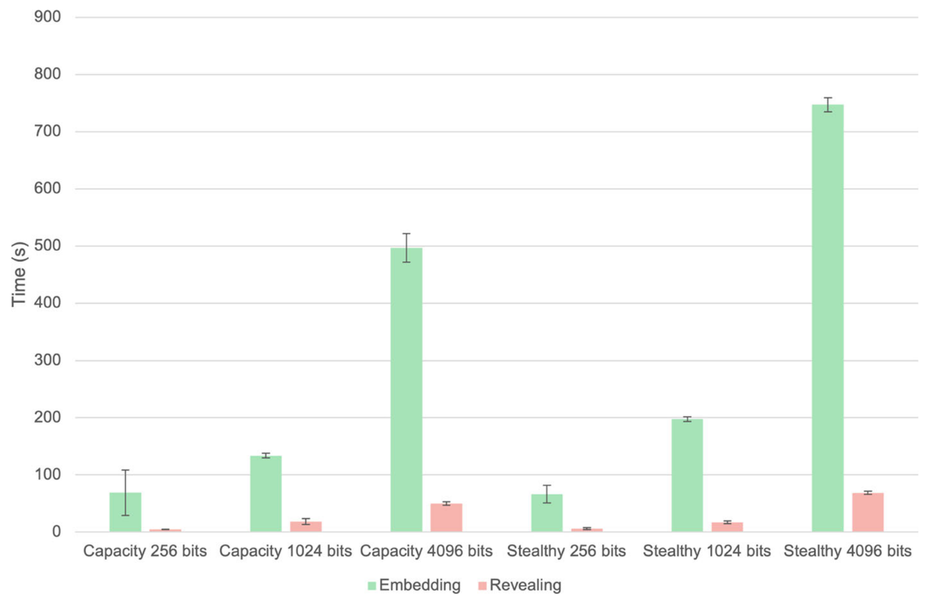

Fig. 6 Network management time per method

Table 8 Malware in blockchain study

Proposal	Study				Blockchain technologies	Types of malware
	Elements used	Desired properties	Data exchange	Cost for the attacker		
[12]	X	X	X	X	Bitcoin	Ransomware
[94]	X	X	√	X	Bitcoin	Ransomware
[62]	X	X	X	X	Bitcoin	Ransomware
[97]	√	X	√	X	Bitcoin	Ransomware
[95]	X	X	X	X	Bitcoin	Ransomware
[96]	X	X	X	X	Bitcoin	Ransomware
[13]	√	√	√	√	All	Botnet
Ours	√	√	√	√	All	Botnet, Ransomware, Worm

6.3.2 Time assessment

The time taken by Smart-Zephyrus is divided in computation and network time. The former refers to the time required for secret preparation, retrieval and encryption, while the latter refers to sending the transaction to the blockchain, its retrieval, mining time and contract verification time on Etherscan. Although network time is out of the scope of Smart-Zephyrus because it depends on external factors, the real-world suitability analysis of the mechanism requires such time.

The embedding and revealing time for each method depending on the secret's size is presented in Fig. 5. Embedding times includes encryption and preparation of the message and its concealment in contracts. On the other hand, revealing time includes extracting the secret and decrypting it afterwards. The time of encryption and decryption is quite small regarding the total time, and very similar among the same message sizes. It ranges from 0.001 to 0.0035 s. As expected, the size of the message directly affects the time of embedding and revealing. As the capacity method performs more operations and transformations to embed the message, it takes longer on the revealing whereas the stealthy needs more contracts to embed the same quantity of information and then, it takes longer. It is specially noticeable on the 4096 bits case, where it takes an average of 41 s with a standard deviation of 5.07 for the capacity method in the embedding versus 62 s and 5.40 for the stealthy one. Regarding the revealing times, it takes 2.56 s to reveal a 4,096 bit message for the capacity method with a standard deviation of 5.07 s versus 0.47 s with a standard deviation of 0.20 s for the stealthy method. These times can be considered very low as the applied scale has a maximum of 70 *s*.

Figure 6 presents the time of embedding and revealing messages for each method. It can be observed that the network time, although it usually increases with the message size is largely dependent on the state of the network. Most of the time is consumed by the mining and verification process. For example, for the capacity method with 4096 bits of information the average mining time is 88 s, while the verification time is 383 s. Nonetheless, it should be noticed that verification times are similar to each other, with a standard deviation of 4.28 s, while the mining time has a standard deviation of 27 s.

7 Related work

In this section, we study the related works for each of the contributions of this paper. For the sake of clarity, we address them separately. Thus, Sect. 7.1 describes previous works on the use of blockchains by malwares. On the other hand, Sect. 7.2 focuses on works that have used blockchain and smart contracts to embed steganographic information in the blockchain.

7.1 Malware in blockchain

There are some works related to blockchain and malware. Financial issues are significantly considered, Kshetri and Voas [12] studies the effect of cryptocurrencies on ransomware and what could influence a victim to pay. Complementary, Orman [94] explains how the blockchain is used for payment and key delivering, also describing how modern ransomware works and possible defenses. Moreover, Paquet-Clouston et al. [62] provides a comprehensive, evidence-based picture on the global direct financial impact of ransomware attacks. They empirically analyse Bitcoin transactions related to 35 ransomware families. Also in the economic field, Conti et al. [95] present a comprehensive study on all recent ransomware and report their economic impact from the Bitcoin payment perspective. Besides, in Huang et al. [96] a measurement framework is developed for performing a large-scale, two-year, end-to-end measurement of ransomware payments, victims and operators.

Table 9 Smart contacts' embedding proposals

Reference	Technology	Method	Max capacity (bits)	Stealthiness	Use of high level language	Secret integrity	Comparison of embedded data with normal content in blockchain
[72]	Bitcoin	ScriptPubKey:paytopubkeyhash	160	BE	X	✓	X
[74]	Bitcoin	ScriptSig	8	ALL	X	✓	X
	Bitcoin	ScriptPubKey:paytoscripthash-multisig	2766	ALL	X	✓	X
[98]	Bitcoin	ScriptPubKey:paytopublickey	520/264	BE,AE	X	✓	X
	Bitcoin	ScriptPubKey:paytoscripthash	160	BE,AE	X	✓	X
	Bitcoin	ScriptSign(Redeem script):paytoscripthash	4136	BE,AE	X	✓	X
	Bitcoin	ScriptSign(Data input):DataDropwithoutSignature	13,040	BE,AE	X	✓	X
	Bitcoin	ScriptSign(Data input):DataDropwithSignature	12232	BE,AE	X	✓	X
	Bitcoin	ScriptSign(Data input):DataHashwithoutSignature	12480	BE,AE	X	✓	X
	Bitcoin	ScriptSign(Data input):DataHashwithSignature	11688	BE,AE	X	✓	X
[99]	Bitcoin	ScriptSign:multisignature	448	BE,AE/ALL	X	✓	X
[100]	Bitcoin, Ethereum	OP_RETURN + sender address/order	640 (hash of key words), 32 (message)	BE, AE/ALL (encryption)	X	✓	X
[101]	Ethereum	Timestamp in smart contract	29	BE	X	✓	X
[32]	Ethereum	Constructor arguments	160	ALL	X	✓	✓
	Ethereum	Bytecode	46	AE	X	✓	✓
	Ethereum	Swarm hash	256	ALL	X	✓	✓
Smart-Zephyrus	Ethereum	Solidity contract code	42/32	ALL	✓	✓	✓

On the other hand, Faisal et al. [97] studied the ability of Bitcoin to store metadata and showed basic approaches to improve blockchain privacy. It identifies and classifies blockchain transactions embedding metadata of major protocols running on top of Bitcoin. It also exposes the possibility of using stealthy addresses, as well as the use of smart contracts to automate ransom payments.

In the field of botnets, Böck et al. [13] provide a comprehensive systematization of the state of the art of blockchain-based botnets, along with an abstract model of such system.

Table 8 shows a comparison between previous works and our study. None of existing proposals analyses in depth how the blockchain is used. They do not study desired properties, the different elements of the blockchain being used or how the information is exchanged in the system. They usually focus on one technology (namely Bitcoin) and ransomware. By contrast, our study analyses, from a cybersecurity point of view, different technologies and malwares, as well as other issues like malware purposes and the cost for the attacker.

7.2 Embedding information in smart contracts

Some works already embed information in smart contracts. Ken Shirriff [72] identified how to embed information in the blockchain or Bitcoin scripts, which can be considered analogous to smart contracts. In particular, the hash of the public key script (P2PKH) is used to insert information.

Also with the focus on Bitcoin, Okupski [74] presents different fields to hide messages without the use of encryption. In the case of Pay-to-Pubkey and Pay-to-PubKeyHash scripts, the public key and the signature are used to embed information. In multisignature scripts, keys located in the public key script are used to transmit data. Besides, Sward et al. [98] propose different insertion methods in Bitcoin. Information is included in the public key in a Pay to Public Key (P2PK) transaction, or in the hash of the public key in a Pay to Script Hash (P2SH), being ScriptPubKey of a transaction used for this purpose. Information can be additionally embedded in ScriptSig, which is the complementary part of a complete and valid transaction. R. Recabarren at al. [99] also propose the use of ScriptSig for inserting information. They present Thitonious, an anti-censorship Bitcoin tool in which scriptSig of a P2SH multisignature transaction is used to embed a message on the 28 most significant bytes.

Zhang et al. [100] propose the use of Ethereum smart contracts and Bitcoin for information exchanges. Two types of smart contracts are defined: a voting contract which uses the OP_RETURN to transmit a hash, and applies the order and option of the voting addresses or the addresses themselves for embedding purposes; and a bidding contract which also uses the OP_RETURN to transmit a hash, and embeds data in the bids.

By contrast, just Basuki et al. [101] applied Ethereum. Their purpose is on hiding instructions for recovering secrets within images in the smart contract's timestamp, having 29 bits of capacity. Additionally, Gimenez-Aguilar et al. [32] proposed the use of Ethereum smart contracts using the bytecode, constructor arguments and the swarm hash to hide information. The maximum capacity in bits per transaction is 46, 160 and 256, respectively.

In sum, considering Table 9, Smart-Zephyrus is the only one who actually uses a high-level language to insert information in the blockchain providing also a high level of stealthiness (ALL) against the attacker models defined by Gimenez-Aguilar et al. [32], which is more powerful than most proposals (except for [32, 74, 99, 100]). In Smart-Zephyrus, the capacity is approximately 42 bits for the capacity method and 32 for the stealthy one according to experimental results. Although the maximum capacity is lower than other works, it is still higher than in [74] for the ScriptSig, in Basuki et al. [101] and in Zephyrus for the Bytecode method [32]. Furthermore, contracts are completely verifiable on Etherscan making them seem legitimate and increasing the level of trust in the ecosystem [102]. On the other hand, the embedding procedure is compatible with other methods, like Zephyrus' Swarm hash method or constructor arguments one.

8 Conclusions

Blockchain provides certain characteristics that malware can use to improve their attacks, i.e. permanent availability and immutability. We have studied how this technology has been used by different types of malware and presented a comprehensive analysis from different perspectives. Among other open research issues, it has been found that the use of covert communication channels has not been explored in this area. To further motivate future research works, an open-source tool (called Smart-Zephyrus) has been proposed. It uses a high-level smart contract language (Solidity) to insert information achieving a high level of stealthiness and thus reducing the chance for an attacker to be detected. The time and cost for the attacker have also been characterized.

As future research directions, the development of detection techniques against this type of communication is needed. On the other hand, the use of other languages for smart contracts (e.g. Vyper) would contribute to characterize the generalization of this technique.

Author Contributions Mar Giménez-Aguilar carried out the literature review, implemented SmartZephyrus and carried out the experiments. She also contributed to the paper writing, which was carried out and supervised by Lorena Gonzalez and Jose Maria de Fuentes. Lorena Gonzalez and Jose Maria de Fuentes designed the experiments.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This work was supported by the Madrid Government (Comunidad de Madrid-Spain) under the multiannual agreement with UC3M ("fostering young doctor research", DEPROFAKE-CM-UC3M) and in the context of the V PRICIT research and technological innovation regional program; by CAM by grant CYNAMON P2018/TCS-4566-CM, co-funded with ERDF; by 1208 Min. of Science and Innovation of Spain by grant ODIO PID2019-1209 111429RB-C21 (AEI/10.13039/50110 12100011033); and by Funding for APC: Universidad Carlos III de Madrid (Read & Publish Agreement CRUE-CSIC 2023).

Data Availability The datasets generated during and/or analysed during the current study are available in the repository cited within the manuscript.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose. The authors have no competing interests to declare that are relevant to the content of this article. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript. The authors have no financial or proprietary interests in any material discussed in this article.

Appendix: Tables

See Tables 10, 11, 12.

Table 10 Studied sample (I)

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[41]	Custom	Any	Immutability, availability, decentralization	Data field (OP_RETURN)	Bitcoin	Hardcoded	–	From 1 to 1	Malware storage	No	No	Yes
[7]	Petya	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	–	No
[44]	NotPetya	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	–	No
[6]	Cerber	Ransomware	Availability, anonymity, decentralization	Transaction (Receiver address)	Bitcoin	Hardcoded	Various to 1	–	Payments and domain generator	No	No	Yes
[42]	Custom	Ransomware	Immutability, availability, decentralization	Program code (Function arguments)	Ethereum	Not known	Various to 1	From 1 to various	Key distribution, payments	No	Yes	Yes
[47]	GrandCrab	Ransomware	Anonymity, decentralization	Transaction	Dash	Webpage	Various to 1	–	Payments	No	No	No
[49]	TeslaCrypt	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to various	–	Payments	No	No	No
[55]	ZCrypt	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Downloaded	Various to various	–	Payments	No	No	No
[103]	Maktub	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to 1	–	Payments	No	No	No

Table 10 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[50]	Samsa	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to 1	–	Payments	No	No	No
[103]	BitPaymer	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Dharma	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Not known	Various to not known	–	Payments	No	No	No
[103]	Cryptowall	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to various	–	Payments	No	No	No
[103]	CryptoLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[104]	KeyHolder	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to various	–	Payments	No	No	No
[62, 103]	Locky	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to 1	–	Payments	No	No	No
[103, 105]	Spora	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to 1	–	Payments	No	No	No
[103]	mmm	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No

Table 10 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[103]	rapid	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to not known	–	Payments	No	No	No
[103]	saturn	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to not known	–	Payments	No	No	No
[106, 107]	Jigsaw	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	globeimporster	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[51, 103]	blind	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to not known	–	Payments	No	No	No
[63]	Princess Locker	Rasomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to not known	–	Payments	No	No	No
[103, 108]	Ryuk	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[43]	CTBLocker (webpage version)	Immutability, Availability, Ransomware	Anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Webpage	Various to various	From various to various	Payments, key distribution	No	No	No
[109]	TorrentLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	CryptXXX	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to one	–	Payments	No	No	No
[62]	DMALockerv3	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No

Table 10 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[61]	CryptoTorLocker2015	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to one	–	Payments	No	No	No
[52]	Globe	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to various	–	Payments	No	No	No
[50]	SamSam	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	NoobCrypt	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[110]	EDA2	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Webpage	Various to various	–	Payments	No	No	No
[111]	Flyper	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Globev3	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Cryptohitman	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to one	–	Payments	No	No	No
[112]	TowerWeb	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to one	–	Payments	No	No	No

Table 11 Studied sample (II)

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[103]	s7v7m	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[113]	Bucbi	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Buddy	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Chimera	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[114]	CryptoHost	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	ThunderCrypt	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Trump Locker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[115]	Doxware	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Badblock	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[116]	AdamLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No

Table 11 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[117]	Alphabet	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	DMALocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	AngleWare	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	APT	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	BadEncript	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Black Feather	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	BTCWare	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to 1	–	Payments	No	No	No
[103]	Comrade Circle	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[118]	CryptConsole	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Crypto-SweetTooth	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Cyber Splitter	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Domino	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No

Table 11 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[119]	Exotic	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[120]	FakeGlobe	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Fantom	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to not known	–	Payments	No	No	No
[121]	Fantom (variant)	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	FireCrypt	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[122]	Globe2	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to various	–	Payments	No	No	No
[123]	Ransomeer	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[124]	1me_expl0it	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[125]	Nemucod	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Nullbyte	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	PayDay	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[126]	Philadelphia	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	Phoenix	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No

Table 12 Studied sample (III)

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[127]	PopCornTime	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Random6	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Mail	Various to not known	–	Payments	No	No	No
[103]	RansomPlus	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	Razy	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	REKTLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[103]	VenusLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[103]	XLocker	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[128]	XTPLocker 5.0	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No

Table 12 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[103]	Zyka	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to 1	–	Payments	No	No	No
[64]	Custom	Ransomware	Immutability, availability, anonymity, decentralization	Program code (Function arguments)	Ethereum	Webpage	–	From various to 1	Payments, C&C, Key distributions	No	No	Yes
[129]	Wannacry	Ransomware	Anonymity, decentralization	Transaction	Bitcoin	Hardcoded	Various to various	–	Payments	No	No	No
[45]	Custom	Ransomware	Anonymity, decentralization	Transaction	Monero	Not known	Various to not known	–	Payments	No	No	No
[48]	Custom	Botnet	Availability, decentralization	Data field (OP_RETURN)	Bitcoin	None	–	From 1 to 1	Address discovery	No	No	Yes
[56]	Unblockable chains	Botnet	Availability, anonymity decentralization	Program code (Function arguments)	Ethereum	Hardcoded	Various to 1	From 1 to 1	C&C	Obscured	No	Yes
[66]	ChainChannels	Botnet	Availability, anonymity decentralization	Nonce of signature	Any	Hardcoded	–	From 1 to 1	C&C	Obscured	Yes	Yes
[69]	Coinbot	Botnet	Availability, anonymity decentralization	Data field (Data/OP_RETURN)	Bitcoin, Ethereum, Dash, Litecoin, Bitcoin-Cash	Website	- -	From 1 to 1	C&C	Encrypted	No	Yes
[75]	Glupteba	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Hardcoded	–	From 1 to 1	Address discovery	Encrypted	No	Yes

Table 12 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[70]	Custom	Botnet	Availability, decentralization	Program code (Function arguments)	Ethereum	Hardcoded	–	From 1 to 1	C&C	No	No	Yes
[59]	Custom	Botnet, Worm	Availability, anonymity, decentralization	Transaction (Receiver address)	Bitcoin	Hardcoded, Blockchain	–	From various to various	C&C	No	Yes	Yes
[57]	Neuromesh	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Hardcoded	–	From 1 to various	C&C	No	No	Yes
[76]	Botract	Botnet	Availability, anonymity, decentralization	Program code (Function arguments)	Ethereum	Hardcoded	–	From 1 to 1	C&C	No	No	Yes
[54]	Custom	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	C&C	–	From various to various	Address discovery	Encrypted	No	Yes
[9]	Fbot	Botnet	Availability, anonymity, decentralization	Services on top of blockchain(ErmerDNS)	Emercoin	Hardcoded	–	From 1 to various	DNS	No	No	No
[8]	Pony	Botnet	Availability, anonymity, decentralization	Transaction(value)	Bitcoin	Hardcoded	–	From 1 to 1	Address discovery	No	No	Yes
[71]	Custom	Botnet	Immutability, availability, anonymity, decentralization	Program code (Function arguments)	Ethereum	Hardcoded	–	From 1 to 1	C&C	No	No	No

Table 12 continued

Source	Name	Type	Cybersecurity properties	Elements used	Blockchain technology	Data exchange			Purpose	Data protection		
						Seed address	Transaction flux			Confidentiality	Covert channel	Cost to attacker
							Receive transactions	Send transactions				
[67]	Custom	Botnet	Availability, anonymity, decentralization	Services on top of blockchain (Whisper)	Ethereum	Hardcoded	From various to 1	From 1 to various	C&C	Encrypted	No	No
[58]	LNBot	Botnet	Availability, anonymity, decentralization	Transaction(value)	Bitcoin	Hardcoded	From 1 to 1	From 1 to various	C%C	No	Yes	Yes
[65]	Zombiecoin	Botnet	Availability, anonymity, decentralization	Data field& signature (OP_RETURN)	Bitcoin	Hardcoded	–	From 1 to various	C&C	No	Yes	Yes
[46]	Custom	Botnet	Availability, anonymity, decentralization	Payment id	Monero	Hardcoded	From various to 1	From various to various	C&C	Encrypted	No	Yes
[77]	Dustbot	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Hardcoded	From various to 1	From 1 to 1	C&C	No	No	Yes
[78]	Botchain	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Webpage	–	From 1 to various	C&C	No	No	Yes
[79]	Custom	Botnet	Availability, anonymity, decentralization	Data field (OP_RETURN)	Bitcoin	Hardcoded	From various to various	From 1 to various	C&C	Encrypted	No	No

References

1. Iansiti, M., Lakhani, K.R.: The truth about blockchain. Harv. Bus. Rev. **95**(1), 118–127 (2017)
2. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System (2008)
3. Wood, G.: Ethereum Yellow Paper (2019). https://ethereum.github.io/yellowpaper/paper.pdf
4. Kane, E.: Is Blockchain a General Purpose Technology? SSRN (2017)
5. Fenton, A.: Almost Half of Bitcoin Payments are Now Made on the Darknet (2019). https://micky.com.au/almost-half-of-bitcoin-payments-are-now-made-on-the-darknet/
6. Pletinckx, S., Trap, C., Doerr, C.: Malware coordination using the blockchain: an analysis of the cerber ransomware. In: 2018 IEEE Conference on Communications and Network Security (CNS), pp. 1–9 (2018). https://doi.org/10.1109/CNS.2018.8433199
7. Aidan, J.S., Verma, H.K., Awasthi, L.K.: Comprehensive survey on petya ransomware attack. In: 2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS), pp. 122–125. IEEE (2017)
8. Eisenkraft, K., Olshtein, A.: Pony's c&c servers hidden inside the bitcoin blockchain, Tech. rep., Technical Report. Check Point (2019). https://research.checkpoint.com/2019/ponys-cc-servers-hidden-inside-the-bitcoin-blockchain/
9. Ilascu, I.: New Botnet Hides in Blockchain DNS MIST and Removes Cryptominer (2018). https://www.bleepingcomputer.com/news/security/new-botnet-hides-in-blockchain-dns-mist-and-removes-cryptominer/
10. Report: Malware Poisons One-Third of World's Computers (2014). https://www.technewsworld.com/story/report-malware-poisons-one-third-of-worlds-computers-80707.html
11. Ventures, C.: Cybercrime Damages are Predicted to Cost the World $6 Trillion Annually by 2021 (2018). https://www.prnewswire.com/news-releases/cybercrime-damages-are-predicted-to-cost-the-world-6-trillion-annually-by-2021-300540158.html
12. Kshetri, N., Voas, J.: Do crypto-currencies fuel ransomware? IT Prof. **19**(5), 11–15 (2017). https://doi.org/10.1109/MITP.2017.3680961
13. Böck, L., Alexopoulos, N., Saracoglu, E., Mühlhäuser, M., Vasilomanolakis, E.: Assessing the threat of blockchain-based botnets. In: 2019 APWG Symposium on Electronic Crime Research (eCrime), pp. 1–11. IEEE (2019)
14. Axon, L., Goldsmith, M.: Pb-PKI: A Privacy-Aware Blockchain-Based PKI (2016)
15. Srivastav, K.: A Guide to Blockchain Immutability and Challenges—Dzone Security (2021). https://dzone.com/articles/a-guide-to-blockchain-immutability-and-chief-chall
16. EC-Council: What is Blockchain Immutability and How Does It Help? (2021). https://blog.eccouncil.org/what-is-blockchain-immutability-and-how-does-it-help/
17. Ozili, P.K.: Decentralized finance research and developments around the world. J. Bank. Financ. Technol. (2022). https://doi.org/10.1007/s42786-022-00044-x
18. ISO: It security and privacy—a framework for identity management—part 1: Terminology and concepts. In: ISO/IEC 24760-1, ISO (2019)
19. Adrian, M.: Is Cryptocurrency Anonymous? The Myth of Anonymity Debunked. https://www.ulam.io/blog/is-cryptocurrency-anonymous/
20. Cvllr, J.: Solidity Tutorial: All About Functions (2021). https://jeancvllr.medium.com/solidity-tutorial-all-about-functions-dba2ccb1e931
21. Soldiity types. https://docs.soliditylang.org/en/v0.8.10/types.html
22. Somin, S., Gordon, G., Altshuler, Y.: Network analysis of erc20 tokens trading on Ethereum blockchain. In: International Conference on Complex Systems, pp. 439–450. Springer (2018)
23. What is openzeppelin? The ultimate guide "moralis" the ultimate web3 development platform (2021). https://moralis.io/what-is-openzeppelin-the-ultimate-guide/
24. Daly, L.: What is Dash Cryptocurrency? (2021). https://www.fool.com/investing/stock-market/market-sectors/financials/cryptocurrency-stocks/dash-cryptocurrency/
25. Dashpay, Whitepaper · dashpay/dash wiki. https://github.com/dashpay/dash/wiki/Whitepaper
26. Ray, S.: Blockchains: The Technology of Transactions (2021). https://towardsdatascience.com/blockchains-the-technology-of-transactions-9d40e8e41216
27. What is the Ethereum transaction data structure? (1964). https://ethereum.stackexchange.com/questions/1990/what-is-the-ethereum-transaction-data-structure
28. Op_return. https://en.bitcoin.it/wiki/OP_RETURN
29. Moneropedia: Payment id. https://www.getmonero.org/resources/moneropedia/paymentid.html
30. Understanding covert channels of communication. https://www.isaca.org/resources/news-and-trends/isaca-now-blog/2017/understanding-covert-channels-of-communication
31. Kahn, D.: The history of steganography. In: Information Hiding: First International Workshop Cambridge, UK, May 30–June 1, 1996 Proceedings, pp. 1–5. Springer (2005)
32. Gimenez-Aguilar, M., De Fuentes, J.M., González-Manzano, L., Camara, C.: Zephyrus: an information hiding mechanism leveraging Ethereum data fields. IEEE Access **9**, 118553–118570 (2021). https://doi.org/10.1109/ACCESS.2021.3106713
33. Archiveddocs, Defining Malware: Faq. https://docs.microsoft.com/en-us/previous-versions/tn-archive/dd632948(v=technet.10)?redirectedfrom=MSDN
34. Ransomware: What is Ransomware: Ransomware Attack. https://www.malwarebytes.com/ransomware
35. Belcic, I.: (2021). [link]. https://www.avast.com/c-botnet?redirect=1
36. Kaspersky, What are bots?—Definition and Explanation (2021). https://www.kaspersky.com/resource-center/definitions/what-are-bots
37. Radware, Botmaster. https://www.radware.com/security/ddos-knowledge-center/ddospedia/botmaster/
38. Jadhav, S., Dutia, S., Calangutkar, K., Oh, T., Kim, Y.H., Kim, J.N.: Cloud-based android botnet malware detection system. In: 2015 17th International Conference on Advanced Communication Technology (ICACT), pp. 347–352. IEEE (2015)
39. Vengatesan, K., Kumar, A., Parthibhan, M., Singhal, A., Rajesh, R.: Analysis of Mirai botnet malware issues and its prediction methods in internet of things. In: International conference on Computer Networks, Big data and IoT, pp. 120–126. Springer (2018)
40. Security, P.: Computer Worms—Panda Security. https://www.pandasecurity.com/en/security-info/worm/
41. Moubarak, J., Chamoun, M., Filiol, E.: Developing a K-ary malware using blockchain. In: 2018 IEEE/IFIP Network Operations and Management Symposium, pp. 1–4 (2018). https://doi.org/10.1109/NOMS.2018.8406331
42. Delgado-Mohatar, O., Sierra-Cámara, J.M., Anguiano, E.: Blockchain-based semi-autonomous ransomware. Future Gener. Comput. Syst. **112**, 589–603 (2020). https://doi.org/10.1016/j.future.2020.02.037
43. Sinegubko, D.: Website Ransomware—CTB-locker Goes Blockchain (2018). https://blog.sucuri.net/2016/04/website-ransomware-ctb-locker-goes-blockchain.html

44. Fayi, S.Y.A.: What Petya/NotPetya ransomware is and what its remidiations are. In: Latifi, S. (ed.) Information Technology-New Generations, pp. 93–100. Springer, Cham (2018)
45. Hurtuk, J., Chovanec, M., Kičina, M., Billík, R.: Case study of ransomware malware hiding using obfuscation methods. In: 2018 16th International Conference on Emerging eLearning Technologies and Applications (ICETA), pp. 215–220 (2018). https://doi.org/10.1109/ICETA.2018.8572218
46. Mengidis, A.: Blockchain-based command and control for next generation botnets (2019)
47. Lemmou, Y., Souidi, E.M.: Inside gandcrab ransomware. In: Camenisch, J., Papadimitratos, P. (eds.) Cryptology and Network Security, pp. 154–174. Springer, Cham (2018)
48. Kamenski, D., Shaghaghi, A., Warren, M.J., Kanhere, S.S: Attacking with bitcoin: using bitcoin to build resilient botnet armies. (2020). arXiv:2004.01855
49. Lemmou, Y., Souidi, E.M.: Infection, self-reproduction and overinfection in ransomware: the case of teslacrypt. In: 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), pp. 1–8 (2018). https://doi.org/10.1109/CyberSecPODS.2018.8560670
50. Grunzweig, J.: Samsa Ransomware Attacks: A Year in Review (2018). https://unit42.paloaltonetworks.com/unit42-samsa-ransomware-attacks-year-review/
51. Labs, M.: Napoleon: A New Version of Blind Ransomware (2021). https://blog.malwarebytes.com/threat-analysis/2017/12/napoleon-ransomware/
52. Abrams, L.: The Globe Ransomware Wants to Purge Your Files (2016). https://www.bleepingcomputer.com/news/security/the-globe-ransomware-wants-to-purge-your-files/
53. Meskauskas, T.: Random6 Ransomware (2020). https://www.pcrisk.com/removal-guides/11409-random6-ransomware
54. Curran, T., Geist, D.: Using the bitcoin blockchain as a botnet resilience mechanism (2016)
55. Labs, M.: Zcrypt Ransomware: Under the Hood (2021). https://blog.malwarebytes.com/threat-analysis/2016/06/zcrypt-ransomware/
56. Platdrag: Platdrag/Unblockablechains: Unblockable Chains—A Poc on Using Blockchain as Infrastructure for Malware Operations. https://github.com/platdrag/UnblockableChains
57. Falco, G., Li, C., Fedorov, P., Caldera, C., Arora, R., Jackson, K.: Neuromesh: Iot security enabled by a blockchain powered botnet vaccine. In: Proceedings of the International Conference on Omni-Layer Intelligent Systems, COINS '19, Association for Computing Machinery, New York, NY, USA, p. 1–6 (2019). https://doi.org/10.1145/3312614.3312615
58. Kurt, A., Erdin, E., Cebe, M., Akkaya, K., Uluagac, A.S.: Lnbot: a covert hybrid botnet on bitcoin lightning network for fun and profit. In: European Symposium on Research in Computer Security, pp. 734–755. Springer (2020)
59. Roffel, D., Garret, C.: A-novel-approach-for-computer-worm-control-using-decentralized-data-structures (2014). https://archive.org/stream/pdfy-E2ZwuLAVfC44kEQk/250009335-A-Novel-Approach-for-Computer-Worm-Control-Using-Decentralized-Data-Structures_djvu.txt
60. Karapapas, C., Pittaras, I., Fotiou, N., Polyzos, G.C.: Ransomware as a service using smart contracts and IPFS. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–5 (2020). https://doi.org/10.1109/ICBC48266.2020.9169451
61. Grinler: New cryptotorlocker2015 Ransomware Discovered and Easily Decrypted—Archived News (2015). https://www.bleepingcomputer.com/forums/t/565020/new-cryptotorlocker2015-ransomware-discovered-and-easily-decrypted/
62. Paquet-Clouston, M., Haslhofer, B., Dupont, B.: Ransomware payments in the Bitcoin ecosystem. J. Cybersecur. **5**(1), tyz003 (2019). https://doi.org/10.1093/cybsec/tyz003
63. Labs, M.: Princesslocker—Ransomware with not So Royal Encryption (2021). https://blog.malwarebytes.com/threat-analysis/2016/11/princess-ransomware/
64. Karapapas, C., Pittaras, I., Fotiou, N., Polyzos, G.C.: Ransomware as a service using smart contracts and IPFS. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–5 (2020). https://doi.org/10.1109/ICBC48266.2020.9169451
65. Ali, S. T., McCorry, P., Lee, P. H.-J., Hao, F.: Zombiecoin: Powering next-generation botnets with bitcoin. In: International Conference on Financial Cryptography and Data Security, pp. 34–48. Springer (2015)
66. Frkat, D., Annessi, R., Zseby, T.: Chainchannels: Private botnet communication over public blockchains. In: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 1244–1252 (2018). https://doi.org/10.1109/Cybermatics_2018.2018.00219
67. Baden, M., Ferreira Torres, C., Fiz Pontiveros, B.B., State, R.: Whispering botnet command and control instructions. In: 2019 Crypto Valley Conference on Blockchain Technology (CVCBT), pp. 77–81 (2019). https://doi.org/10.1109/CVCBT.2019.00014
68. Etherscan: https://etherscan.io/. Last access April 2021
69. Yin, J., Cui, X., Liu, C., Liu, Q., Cui, T., Wang, Z.: Coinbot: a covert botnet in the cryptocurrency network. In: International Conference on Information and Communications Security, pp. 107–125. Springer (2020)
70. Sweeny, J.: Botnet Resiliency via Private Blockchains. SANS Institute Information Security Reading Group, North Bethesda (2017)
71. Oliveira, A., Gonçalves, V., Filho, G.R.: Using Ethereum Smart Contracts for Botnet Command and Control, copyright—Copyright Academic Conferences International Limited Jun 2020; Última actualización—2021-07-13 (2020). https://www.proquest.com/conference-papers-proceedings/using-ethereum-smart-contracts-botnet-command/docview/2453793786/se-2?accountid=14501
72. Shirriff, K.: Hidden Surprises in the Bitcoin Blockchain and How They are Stored. http://www.righto.com/2014/02/ascii-bernanke-wikileaks-photographs.html#ref6. Last access Nov. 2018
73. Partala, J.: Provably secure covert communication on blockchain. Cryptography **2**(3), 18 (2018)
74. Okupski, K.S.: (ab) Using Bitcoin for Anti-censorship Tool. Technische Universiteit Eindhoven Master Thesis (2014) (2014)
75. Horejsi, J., Chen, J.C.: Glupteba Hits Routers and Updates c&c Servers (2019). https://www.trendmicro.com/en_us/research/19/i/glupteba-campaign-hits-network-routers-and-updates-cc-servers-with-data-from-bitcoin-transactions.html
76. Malaika, N.M.A., Al Ibrahim, O.: Botract: Abusing smart contracts and blockchains for botnet command and control
77. Zhong, Y., Zhou, A., Zhang, L., Jing, F., Zuo, Z.: Dustbot: a duplex and stealthy p2p-based botnet in the bitcoin network. PLoS ONE **14**(12), e0226594 (2019)
78. Pirozzi, A.: Botchain aka the dark side of blockchain (2018)
79. Franzoni, F., Abellan, I., Daza, V.: Leveraging bitcoin testnet for bidirectional botnet command and control systems. In: International Conference on Financial Cryptography and Data Security, pp. 3–19. Springer (2020)
80. Ethereum historical data. https://www.investing.com/crypto/ethereum/historical-data
81. Monero historical data. https://www.investing.com/crypto/monero/historical-data

82. Bitcoin historical data. https://www.investing.com/crypto/bitcoin/historical-data
83. Minimum for sending BTC from BTC wallet. https://bitcoin.stackexchange.com/questions/105214/minimum-for-sending-btc-from-btc-wallet
84. CoinMarketCap: What is a Crypto Faucet?: Coinmarketcap (2021). https://coinmarketcap.com/alexandria/article/what-is-a-crypto-faucet
85. Carr, S.: How do Botnets Make Money from Your Ads? (2021). https://ppcprotect.com/blog/ad-fraud/how-botnets-make-money/
86. Namestnikov, Y.: The Economics of Botnets, Analysis on Viruslist. com. Kapersky Lab (2009)
87. Makrushin, D.: The Cost of Launching a DDOS Attack (2021). https://securelist.com/the-cost-of-launching-a-ddos-attack/77784/
88. Namestnikov, Y.: The Economics of Botnets. https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2009/07/01121538/ynam_botnets_0907_en.pdf
89. Ethereum lists. https://github.com/MyEtherWallet/ethereum-lists
90. Networks. https://ethereum.org/nb/developers/docs/networks/#sepolia
91. Ethereum average gas price. https://ycharts.com/indicators/ethereum_average_gas_price#:~:text=Ethereum%20Average%20Gas%20Price%20is,84.76%25%20from%20one%20year%20ago
92. Putman, C., Nieuwenhuis, L.J., et al.: Business model of a botnet. In: 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), pp. 441–445. IEEE (2018)
93. Bistarelli, S., Parroccini, M., Santini, F.: Visualizing bitcoin flows of ransomware: Wannacry one week later. In: ITASEC (2018)
94. Orman, H.: Evil offspring—ransomware and crypto technology. IEEE Internet Comput. **20**(5), 89–94 (2016). https://doi.org/10.1109/MIC.2016.90
95. Conti, M., Gangwal, A., Ruj, S.: On the economic significance of ransomware campaigns: a bitcoin transactions perspective. Comput. Secur. **79**, 162–189 (2018)
96. Huang, D.Y., Aliapoulios, M.M., Li, V.G., Invernizzi, L., Bursztein, E., McRoberts, K., Levin, J., Levchenko, K., Snoeren, A.C., McCoy, D.: Tracking ransomware end-to-end. In: IEEE Symposium on Security and Privacy (SP), pp. 618–631. IEEE (2018)
97. Faisal, T., Courtois, N., Serguieva, A.: The evolution of embedding metadata in blockchain transactions. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–9 (2018). https://doi.org/10.1109/IJCNN.2018.8489377
98. Sward, A., Vecna, I., Stonedahl, F.: Data insertion in bitcoin's blockchain. Ledger 3 (2018)
99. Recabarren, R., Carbunar, B.: Tithonus: a bitcoin based censorship resilient system. In: Proceedings on Privacy Enhancing Technologies, pp. 68–86 (2019). https://content.sciendo.com/view/journals/popets/2019/1/article-p68.xml
100. Zhang, L., Zhang, Z., Wang, W., Jin, Z., Su, Y., Chen, H.: Research on a covert communication model realized by using smart contracts in blockchain environment. IEEE Syst. J. **16**, 2822–2833 (2021)
101. Basuki, A.I., Rosiyadi, D.: Joint transaction-image steganography for high capacity covert communication. In: 2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA), pp. 41–46 (2019)
102. Lukic, M.: 5 Important Reasons to Verify Smart Contracts—How to Do It (2022). http://blog.tenderly.co/guide-to-smart-contract-verification-methods/
103. Meskauskas, T.: Ransomware Information. https://www.pcrisk.com/search?searchword=ransomware&ordering=&searchphrase=all
104. Grinler: Keyholder Ransomware Support and Help Topic (2014). https://www.bleepingcomputer.com/forums/t/559463/keyholder-ransomware-support-and-help-topic-how-decryptgifhow-decrypthtml/
105. Labs, M., Labs, M.: Explained: Spora Ransomware: Malwarebytes Labs. https://www.malwarebytes.com/blog/news/2017/03/spora-ransomware
106. Goodin, D.: Meet Jigsaw, the Ransomware that Taunts Victims and Offers Live Support (2016). https://arstechnica.com/information-technology/2016/06/meet-jigsaw-the-ransomware-that-taunts-victims-and-offers-live-support/
107. Settle, A., Leonard, C.: Piecing Together the Jigsaw Puzzle (2019). https://www.forcepoint.com/es/blog/x-labs/piecing-together-jigsaw-puzzle
108. Palmer, D.: Over $1m in ryuk Ransomware Bitcoin was 'Cashed Out' on Binance: Report (2020). https://www.coindesk.com/markets/2020/08/24/over-1m-in-ryuk-ransomware-bitcoin-was-cashed-out-on-binance-report/
109. Torrentlocker: Crypto-ransomware Still Active, Using Same Tactics (2016). https://www.welivesecurity.com/2016/09/01/torrentlocker-crypto-ransomware-still-active-using-tactics/
110. Paganini, P.: Eda2, Derived from The Educational Ransomware, Is Easy to Break (2016). https://securityaffairs.co/wordpress/45336/malware/eda2-easy-decryption.html
111. CagedTech: Flyper Ransomware (2020). https://www.enigmasoftware.com/flyperransomware-removal/
112. Demonslay335: Towerweb Ransomware Help (2016). https://www.bleepingcomputer.com/forums/t/618055/towerweb-ransomware-help-support-topic-payment-instructionsjpg/
113. Bucbi ransomware spreading via RDP brute force attacks. https://www.securityweek.com/bucbi-ransomware-spreading-rdp-brute-force-attacks
114. Abrams, L.: Cryptohost Decrypted: Locks Files in a Password Protected rar File (2016). https://www.bleepingcomputer.com/news/security/cryptohost-decrypted-locks-files-in-a-password-protected-rar-file/
115. Malanga, M.: Everything You Wanted to Know About Doxware (2017). https://monstercloud.com/blog/2017/02/17/what-is-doxware/
116. GoldSparrow: Korean Adamlocker Ransomware (2020). https://www.enigmasoftware.com/koreanadamlockerransomware-removal/
117. Alphabet ransomware virus (removal steps and protection updates) (2017). https://bestsecuritysearch.com/alphabet-ransomware-virus-removal-steps-protection-updates/
118. Morelli, O.: Remove cryptconsole ransomware/virus (removal instructions)—Jun 2018 update (Jun 2018). https://www.2-spyware.com/remove-cryptconsole-ransomware-virus.html
119. Krastev, V.: Exotic 3.0 Ransomware Delete and Fix the Affected Data (2017). https://sensorstechforum.com/exotic-3-0-ransomware-delete-fix-affected-data/
120. Ramos, P.: Fakeglobe and Cerber Ransomware: Sneaking Under the Radar While Wecry (2017). https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/fakeglobe-and-cerber-ransomware-sneaking-under-the-radar-while-wecry/
121. Bilbao, B.: New fantom virus—remove and restore .locked files (2017). https://sensorstechforum.com/new-fantom-virus-remove-restore-locked-files/
122. Globe2 ransomware. https://anti-spyware-101.com/remove-globe2-ransomware
123. Ramsomeer ransomware. https://ransomware.fandom.com/wiki/Ramsomeer_Ransomware

124. l1me_expl0it. https://www.virustotal.com/gui/search/l1me_expl0it/comments
125. Krastev, V.: Remove Nemucod Ransomware and Restore .crypted Encrypted Files (2017). https://sensorstechforum.com/remove-nemucod-ransomware-and-restore-crypted-encrypted-files/
126. 25, A., Staff, P.: Philadelphia Ransomware Brings Customization to Commodity Malware: Proofpoint Us (2019). https://www.proofpoint.com/us/threat-insight/post/philadelphia-ransomware-customization-commodity-malware
127. GoldSparrow: Popcorn Time Ransomware (2020). https://www.enigmasoftware.com/popcorntimeransomware-removal/
128. Remove the xtp locker 5.0 ransomware from your PC (2017). https://bestsecuritysearch.com/remove-xtp-locker-5-0-ransomware-pc/
129. Satheesh Kumar, M., Ben-Othman, J., Srinivasagan, K.: An investigation on Wannacry ransomware and its detection. In: IEEE Symposium on Computers and Communications (ISCC), vol. 2018, pp. 1–6 (2018). https://doi.org/10.1109/ISCC.2018.8538354