Assessing the Threat of Blockchain-based Botnets

Leon Böck*, Nikolaos Alexopoulos*, Emine Saracoglu*, Max Mühlhäuser*, and Emmanouil Vasilomanolakis†
* Technische Universität Darmstadt, Germany
{boeck, alexopoulos, max}@tk.tu-darmstadt.de, e.saracoglu@outlook.com
† Aalborg University, Denmark
emv@cmi.aau.dk

***Abstract*—Time and time again the security community has faced novel threats that were previously never analyzed, sometimes with catastrophic results. To avoid this, proactive analysis of envisioned threats is of great importance. One such threat is blockchain-based botnets. Bitcoin, and blockchain-based decentralized cryptocurrencies in general, promise a fair and more transparent financial system. They do so by implementing an open and censorship-resistant atomic broadcast protocol that enables the maintenance of a global transaction ledger, known as a blockchain. In this paper, we consider how this broadcast protocol may be used for malicious behavior, mainly as a botnet command and control (C2) channel.**

Botmasters have been known to misuse broadcasting platforms, like social media, as C2 channels. However, these platforms lack the integral censorship-resistant property of decentralized cryptocurrencies. We set off by providing a critical analysis of the state of the art of blockchain-based botnets, along with an abstract model of such a system. We then examine the inherent limitations of the design, in an attempt to challenge the feasibility of such a botnet. With such limitations in mind, we move forward with an experimental analysis of the detectability of such botnets and discuss potential countermeasures. Contrary to previous work that proposed such botnets, we provide a broad overview of the associated risk and view the problem in relation to other existing botnet C2 channels. We conclude that despite its limitations, the blockchain, as a backup mechanism, practically renders attempts to suppress the control channel of a botnet futile. Thus, more focus should be put on detecting and disinfecting machines at the network edge (router) or even per-bot level.

I. Introduction

Bitcoin, introduced in Satoshi Nakamoto's famous 2008 white paper [1], implements a peer-to-peer electronic cash system with no trusted third party. Contrary to all previous e-cash systems, Bitcoin solves the double-spend problem in a decentralized manner by utilizing hash-based proof-of-work to achieve consensus and create an atomic broadcast channel, where transactions are totally ordered, everyone can participate, and no central entity can block (censor) transactions. Bitcoin and the blockchain community in general, have evolved greatly since then, growing from small Internet forum groups to a large socio-technical ecosystem spanning multiple disciplines, aiming to establish decentralized cryptocurrencies as a viable means of payment.

However, one of the most widely cited barriers to cryptocurrency adoption is the argument that they can be used to foster criminal activities, the most obvious being paying for widely-considered illegal items and services (drugs, assassination contracts, etc.). These arguments are arguably more political than technical and are well-documented [2].

Recently, another, mostly technical potential criminal act was brought to the attention of the research community [3], [4], [5]; that of establishing a botnet Command and Control (C2) channel via the blockchain. A botnet is a network of infected devices that performs illegal actions for the benefit of its owner, the so-called botmaster. Botnets use different architectures for their internal network communication. These vary from simple centralized structures to more complicated distributed/Peer-to-Peer (P2P) architectures (see Section VI for a complete picture). While centralized botnets offer the botmaster operation simplicity and a global picture of the infected network, they are easy to take down as they inherently have a single point of failure. Similarly, P2P botnets are also heavily studied, complicated to operate, and susceptible to monitoring attempts and sinkholing attacks [6], [7].

In general, botmasters appear to be continuously upgrading their techniques and arsenal (e.g. by utilizing Tor or Twitter [8], [9]). Based on the growing adoption of blockchain technology, the associated P2P traffic may soon be considered normal within networks. This, combined with the fact that the blockchain and its underlying P2P network provide a ready-to-use distributed P2P channel, may lead to its abuse for C2 purposes. Specifically, a public cryptocurrency (network) can provide the C2 with the resilience of a fully distributed network, without the effort of implementing custom P2P protocols. In fact, we have seen similar abuse of legitimate P2P protocols in the past [10].

Although the idea of a blockchain-based botnet lied exclusively in the academic/research world for long, recently a botnet using the blockchain as a DNS server was discovered [11]. Although this falls short of the generic blockchain-based botnet, as described later in the paper, it shows that the threat is real. Therefore, it is vital to proactively analyze the feasibility and advantages of such an advanced botnet, as well as potential countermeasures against it.

Blockchain-based Botnets: A myth? Moving beyond the hype of *blockchain-based-anything*, we analyze the possibilities offered by blockchains, along with the state of the art in botnets, in an attempt to clearly determine the novel/positive aspects of using a blockchain in such a manner. We identified a number of reasons why a botmaster would decide to build their botnet based on a blockchain:

- Anonymity: The design of blockchains intends to provide a degree of anonymity (most often pseudonymity) to its users.

- Robustness: The design of (public) blockchains provides censorship-resistance, i.e. it is robust in the sense that it is resistant to blocking.
- Enumeration Resistance: Bots retrieving commands from the blockchain are hidden among all other legitimate users.
- Stealthiness: Due to the growing popularity of blockchains[1]it is unsuspicious to interact with blockchain networks.
- Simplicity: Due to the large community and open source tools, taking part in a blockchain P2P network is simpler than setting up a custom P2P protocol.

Contributions. This paper provides a critical study on blockchain-based botnets. Starting with Section II, we analyze the existing botnets and literature and compare their features. Afterwards, we propose an abstract model of how such a botnet can be realized (Section III). We then discuss an inherent limitation of such a design, namely that it is unidirectional (Section IV), and move on to examine potential countermeasures for the defenders (Section V), both in terms of detecting the botnet, as well as for blocking it upon detection. The conclusion of our analysis is that the blockchain is a useful backup mechanism for botmasters, effectively rendering attempts to suppress their C2 channel ineffective. However, using the blockchain as the main C2 channel of a botnet would be hardly feasible. As an additional contribution of independent interest, we provide a comparison table of the trade-offs of different botnet communication mechanisms. Contrary to previous work aimed at proposing such mechanisms, our work provides a broad critical overview of the risk posed by blockchain-based botnets, while highlighting their limitations, and putting them in the broader perspective of botnet C2 mechanisms.

We note that the remainder of the paper assumes basic knowledge of the operation and terminology associated with blockchain technology, and in particular Bitcoin. For an unfamiliar reader, we recommend the Bitcoin Wiki[2] as a readily available source of information aiding the comprehension of the rest of the paper.

[1]To avoid any misunderstanding, we note that we use the term *blockchain* to refer not only to the data structure holding transactions, but to include the communication protocol between the peers.

[2]https://en.bitcoin.it/wiki/Main_Page

II. Analysis of blockchain-based botnets

During the last years, several proposals and Proof of Concept (PoC) implementations have been created by botmasters, researchers, and security companies, showing the possibility of *i)* of using a blockchain as an alternative to traditional Domain Generation Algorithm (DGA) [11], and *ii)* of using a blockchain for C2 purposes. In the following, we briefly discuss using the blockchain as an alternative for DGA based botnets. While this concept is interesting, it addresses the rendezvous problem between bot client and server and does not replace the C2 channel itself. Next, we analyze the state of the art proposals for using the blockchain as a novel C2 channel. Afterwards, we summarize the characteristics of those approaches in a more general model.

A. Robust Rendezvous Mechanism

As initially reported in [11], the Fbot botnet (one of the many variants of the Mirai botnet [12]) is exploiting the decentralized blockchain-based Domain Name System (DNS) of EmerCoin [13]. This solves a critical problem of traditional DGA based botnets. The use of a DGA algorithm allows bots to connect to their C2 server, even if a domain or IP address has been seized. However, since the DNS is outside of the control of the botmasters, the DGA can be reverse engineered and all related domains seized by officials [14], [15]. Using the blockchain-based DNS service of EmerCoin overcomes this point of failure; the domains are registered in a distributed fashion in the blockchain and cannot be seized by officials due to the censorship resistance of public blockchains.

Similarly to Fbot's mechanism, Curran and Geist suggest using Bitcoin's OP_RETURN field as a backup mechanism for a centralized botnet [16]. In the event of the main C2 server being unavailable, the botmaster can publish the address of an alternative C2 server on the blockchain. This will allow the bots to retrieve this information and re-connect to the botnet through the new C2 server.

B. Proof of Concept Blockchain Botnets

Although the idea probably existed before, the first written mention of using a blockchain as a botnet C2 can be tracked to a 2014 Reddit post [5] as a result of a student assignment. Then, the concept was further developed in ZombieCoin [3], [4] by Ali et al., which was first published in 2015 and later revised in 2017. It is the first work to discuss the possibility of using public blockchains for botnet C2. The focus of their work are different ways of storing and hiding commands in the Bitcoin blockchain. This is interesting as Bitcoin, contrary to blockchains such as Ethereum, is not designed to store data. Nevertheless, the authors describe four options to store a command within the Bitcoin blockchain. The straightforward option to store data is to use the OP_RETURN field of a transaction, initially designed to accommodate transaction identifiers, in a similar fashion to traditional bank transactions. Originally, the size of the OP_RETURN field was 40 bytes, but has been later upgraded to include up to 80 bytes.

An alternative to the OP_RETURN field, is the use of so-called *unspendable outputs*. This method sends small amounts of BTC to invalid output addresses effectively making the spent BTC inaccessible. This process allows to embed up to 20 bytes of data in the output address and was the standard way of storing data on the Bitcoin blockchain before the introduction of the OP_RETURN field.

The third approach, described in ZombieCoin, is the exploitation of key leakage on Elliptic Curve Digital Signature Algorithm (ECDSA) signatures. By choosing the same random factor twice, it is possible to leak the used private key. This method requires making two transactions signing the same message twice. By deriving the private key, bots can then read the C2 message contained in the private key, containing up to 32 bytes. As this approach requires a total of two transactions, voids the private key for further use, and provides less storage space, it is merely an alternative in case the OP_RETURN field cannot be used anymore.

Lastly, ZombieCoin proposes the use of a covert approach using subliminal channels [17], [18]. This requires repeatedly creating signatures on a transaction until the first x bits ($x = 14$ is used by the authors) match the intended command string. This is possible due to the randomness used in the signature algorithm. The authors show several optimizations to generate a suitable set of signatures allowing them to generate eight suitable signatures in two minutes. This method provides a total of 14 bytes to embed the command. While this is much less than using the OP_RETURN field, it is impossible to distinguish from regular transactions and therefore stealthy.

While the aforementioned options are sufficient to send simple commands from the botmaster to the bots, they are not suitable to transfer large updates to the bot or to receive data from the bots. While sending commands in multiple transactions is theoretically possible, this becomes expensive quickly (due to transaction fees); when for instance hundreds of Kbytes are transmitted this way. Furthermore, for bots to use the blockchain as a communication channel, each of them would require to hold some amount of cryptocurrency. Paying this fee from just a single wallet with every bot having the private key, could lead to a loss of all currency if a single instance is reverse engineered. However, when storing even small amounts of BTC and paying the transaction fee for messages originating from every bot, the cost would quickly explode for large botnets.

To overcome this limitation, the authors of ZombieCoin argue that it is possible to use the blockchain to announce rendezvous points at which updates can be downloaded, or information can be dropped off (by the bots). Among the suggested options are classical server-based approaches or the use of public services such as Dropbox, OneDrive, Tumbler, or WordPress.

Frkat et al. [19] pick up the idea of using such *subliminal channels* for botnet C2 on blockchains that was presented in ZombieCoin [4]. They present stealthier and more advanced mechanisms to hide the C2 activity on any blockchain that uses ECDSA or Edwards-curve Digital Signature Algorithm (EdDSA). The main contribution of their approach is to hide the existence of a C2 channel in the first place. However, once the malware falls into the hands of the defenders it can be reverse engineered; hence, the commands can be identified, read and acted upon. Moreover, the presented approach is likely to incur higher costs for transmitting new key material and botmaster addresses compared to the less stealthy option of using OP_RETURN fields. Lastly, another drawback of their design is that their concealed key leakage delays commands by at least one block. In the case of blockchains (e.g. Bitcoin) this introduces a delay of ten minutes compared to using OP_RETURN.

Falco et al. propose NeuroMesh a blockchain-based ”friendly” botnet that intends to improve Internet of Things (IoT) defense [20]. In this offensive security approach (that builds on top of the ZombieCoin idea), the authors argue for the exploitation of botnet mechanics for network defense. The botnet, among other technologies, makes use of the OP_RETURN field of the Bitcoin blockchain as its update channel. As an example use case, the authors describe how the Bitcoin blockchain can be used to securely send updates for blacklists stored on the IoT devices. The main benefits of this approach is the availability and censorship resistance of the blockchain itself.

Moving beyond Bitcoin, Botract [21] discusses the possibility of using Ethereum *smart contracts* for botnet C2. Contrary to Bitcoin transactions, the size of smart-contract transactions is theoretically much less limited, while they also allow to program functionality into the blockchain itself. Furthermore, the authors also discuss the possibility of bots registering to the smart contract. However, Botract does not discuss in detail the costs of doing such operations. This is technically significant, as each bot (or group of bots) would require their own wallets and funds to make changes (register) to the smart contract. In fact, the approach by Zohar [22] uses a similar technique and acknowledges the very high costs incurred by such an approach as a limiting factor. Lastly, Botract does not address how bots contact the botmaster. As discussed before, doing so on the smart contract itself will incur very high costs for larger botnets.

Moreover, Omer Zohar presents a PoC for a botnet C2 running completely on the Ethereum blockchain [22]. Contrary to other approaches, all communication between botmaster and bots is realized through a smart contract. For that, each bot has its own wallet with some Ethereum, is uniquely registered in the smart contract, and all communication is fully encrypted. While this approach is very resilient and difficult to attack, it has a major flaw in its operational costs. In fact, the author states that this approach can realistically only be used by substantially wealthy parties, e.g. nation-funded agencies, and is unrealistic to be deployed in criminal scenarios due to its high cost.

Finally, Sweeny presents an approach to use a private blockchain for botnet C2 [23]. While their system is similar to other approaches (use of smart contracts in a private Ethereum clone), their approach does not cost any money, as it is not run on the public platform. However, herein lies the main issue with the approach. Given that it is running entirely on a P2P network used and controlled only by the botnet, it removes the censorship resistance and makes it similar to other P2P botnets. Therefore, a well designed custom P2P protocol is most likely better suited for all potential purposes.

C. Comparison

We summarize the characteristics of the different approaches in Table I. We differentiate between three different uses of the blockchain in the C2 process: 1) using the blockchain to send commands to the bots, 2) using the blockchain to retrieve information from the bots. and 3) using the blockchain as a backup mechanism to allow bots to reconnect to the C2 infrastructure after a failure. Furthermore, we list if the proposed mechanisms use alternate channels, have high costs attached, and whether they use public or private blockchains.

The findings indicate that most proposals intend to use the blockchain to send commands unidirectional from botmaster to bot. To send messages back to the botmasters an alternate channel is commonly used, as using the blockchain for upstream purposes is attached to high costs. Furthermore, only Sweeny et al. propose to use a private blockchain. However, doing so makes it susceptible to a variety of defenses and loses the main purpose of having a highly robust C2 channel.

	Blockchain Downstream	Blockchain Upstream	Blockchain as Backup	Alternate Channel	Reasonable Cost	Public Blockchain
ZombieCoin [3], [4]	✓	✗	✗	✓	✓	✓
Frkat [19]	✓	✗	✗	✓	✓	✓
NeuroMesh [20]	✓	✗	✗	✓	✓	✓
Botract [21]	✓	?	✗	?	?	✓
Zohar [22]	✓	✓	✗	✗	✗	✓
Sweeny [23]	✓	✓	✗	✗	✓	✗
Curran [16]	✗	✗	✓	✓	✓	✓
Fbot [11]	✗	✗	✓	✓	✓	✓

TABLE I
COMPARISON OF EXISTING PROPOSALS FOR BLOCKCHAIN BASED BOTNET C2: ✓= PROVIDES PROPERTY, ✗= DOES NOT PROVIDE PROPERTY, ? = NOT SPECIFIED.

III. GENERALIZED CONCEPT FOR BLOCKCHAIN-BASED BOTNETS

In this section we present a generalized model for blockchain-based botnets. More specifically, we focus on approaches that use the blockchain as a means to distribute commands, i.e. we do not focus on approaches that use the blockchain only as a backup channel to reconnect to another C2 infrastructure, such as [16], [11].

Even though the details of where, and how commands are published to the blockchain vary, all PoC implementations store the commands on the blockchain. While some use encryption or subliminal channels to hide the commands from plain sight, the reverse engineering of the malware will eventually allow defenders to identify and potentially read the commands.

To reduce the delay between issuing a command and it being published in a new block (approximately 10 minutes for Bitcoin and 15 seconds for Ethereum), several works propose to read the commands before they are published in the blockchain. For this, the bots need to be able to retrieve commands from the underlying P2P network of the blockchain.

Irrespective of the mechanism preferred, bots need to run either a full client[3] or a lightweight client[4] of the selected blockchain. A full client will obtain all transactions by joining the P2P overlay. This will allow the bots to filter for the commands they are interested in and process them accordingly. Light clients do not download every transaction and instead rely on other (full) nodes to obtain the transactions they are interested in. This is commonly achieved by sending a bloom filter to a full node, that has all addresses of interest encoded in it. If a transaction matches such a bloom filter, the full node will forward the information to the light client. For the application scenario of botnet C2 the main concerns are the robustness, bandwidth, hard-disk and computational resource consumption. As the robustness is not impacted heavily by running a light client (assuming there are enough network participants answering light client requests), it is usually more suitable to run a light client implementation to avoid detection on the infected machines due to high resource consumption.

A major concern with blockchain-based C2 is the financial cost. One could envision only using the P2P network for command dissemination without ever spending money to publish commands on the blockchain. However, this is not possible, as transactions are only gossiped within the network if they are valid [24] (this is a simple anti-spam mechanism). Therefore, even if the command is read before being published, the transaction costs have to be paid eventually, since it will eventually be included in the blockchain.

Based on the state of the art and the aforementioned discussion, we have derived a generalized model for botnet command flow on any blockchain. Figure 1 graphically depicts this model. At time t_0 the botmaster issues a command cmd embedded within a transaction tr to any Full Node (FN) within the blockchain network. This command can then be retrieved by any bot running a full or light client (Full Node Bot (FNB) or Light Node Bot (LNB)) at time t_1. Furthermore, the command can be considered irrevocably stored within a block on the blockchain at time t_{lag}, where the lag in Bitcoin is due to the probabilistic consensus algorithm (6 periods is commonly considered enough for most transactions [24]). This ensures that bots coming online at any later point in time $t_{1+x}, x \in \mathbb{N}$, will be able to eventually (considering the lag) retrieve the command by querying other full nodes for the contents of missing blocks.

To send messages back to the botmaster, we pick up the suggestions for alternative channels made by ZombieCoin [4] (and to some degree by Curran and Geist [16]) which use the alternative channels for bidirectional communication and only use the blockchain as a backup. While the works of Zohar and Majid et al. [22], [21] discuss the possibility of also sending messages from the bots to the botmaster using the blockchain, we excluded it from the model on purpose. Within Section IV

[3]Botmasters may implement a full client that only checks for commands but does not verify or store the blocks to save hard disk and computational resources.

[4]Similar to full clients, botmasters may implement their own light clients to further reduce resource consumption.

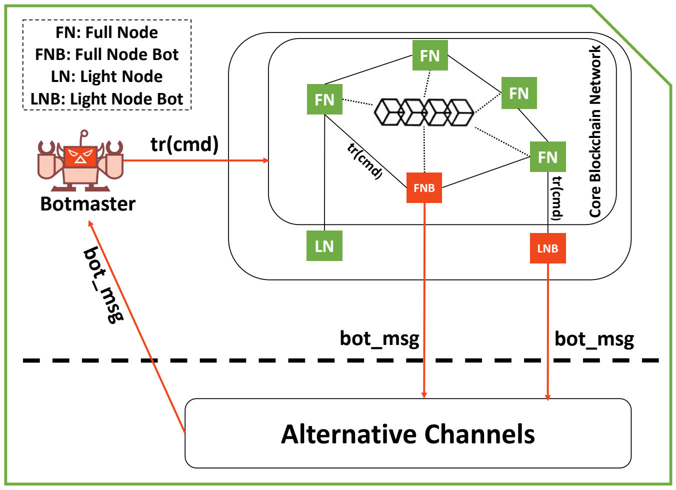


Fig. 1. Abstract view of blockchain-based C2 message flow

we discuss in greater detail why this approach is financially infeasible.

To summarize, we analyzed existing works and proposals on how a blockchain based C2 for botnets could be realized. Furthermore, we derived a general model on how blockchain-based C2 can be realized and elaborated the different approaches of command retrieval. In the model, we abstract from the specific means of transmitting the commands and instead consider a general approach of storing data on *any* blockchain platform, and how it can be accessed by bots. Here, it is worth noting that privacy-preserving cryptocurrencies like *Monero* [25] or *Zcash* [26] do not differ significantly from Bitcoin when used as botnet C2 platforms. Their main additional functionality is breaking the link between sequences of transactions, which is not particularly relevant for our scenario. In the next sections, we will analyze the limitations and potential countermeasures for blockchain based C2 in greater detail.

IV. Upstream Channels

In the previous section we outlined how blockchain-based C2 can be realized in general. However, the approach highlighted in Figure 1 has several limitations and possible countermeasures attached to it. Moreover, regardless of the specific implementations, two crucial factors, namely churn and a back-channel, are disregarded or not addressed satisfyingly by the proposed blockchain-based C2 mechanisms.

At a glance, the greatest limitation of any potential blockchain-based botnet C2 is, that it is only unidirectional. While it provides a highly robust way of sending commands downstream from a botmaster to its bots, bots cannot cost-efficiently send messages upstream to the botmaster using the blockchain. In the following paragraphs, we discuss why an upstream channel is crucial for most botnet business models and how alternative upstream channels greatly weaken the traits of a blockchain-based C2.

A. The need for upstream channels

While in theory, it might suffice to just send commands to the bots, and wait for them to execute their mission, this is highly impractical for many botnet application scenarios. In fact, due to the effects of bot churn, i.e. bots going on- and off-line, a botmaster will not know how many (if any) bots are available at a given time.

Furthermore, even for simple tasks, such as a Distributed Denial of Service (DDoS) attack, this information is crucial to avoid wasting of resources or failing the attack due to a lack of resources. If a botmaster would simply assume that the botnet is large enough to carry out a DDoS and instructed all bots to take part in the attack, that would introduce a high risk of these bots being identified by DDoS protection mechanisms and subsequently blacklisted. This would quickly diminish the capabilities for future attacks and disclose the botnet population to defenders. If, however, the botmaster chooses to instruct a random subset of bots to conduct an attack, this group of bots may be insufficient to effectively conduct the DDoS. In this scenario a customer of the botmaster will be unsatisfied with the DDoS service and likely not use it again in the future. Both scenarios can be avoided by botmasters if they have more accurate knowledge on the availability of their bots.

Considering the application scenario of any kind of data theft, having an upstream channel becomes even more critical. Potential use cases are banking credential theft, espionage, or generic credential theft. Since bots are usually not available at all times (due to churn), permanent and high-bandwidth upstreams provide the best option to realize such an application scenario. Furthermore, a fast two-way communication channel also allows conducting more dynamic queries to identify and steal data.

A third reason for obtaining frequent updates from bots are machine details and location information. Depending on the capabilities of a machine, a botmaster could decide to use it for different purposes. For instance, one may use a powerful server for crypto-mining, whereas a weak IoT-device with fast uplink can be used for DDoS instead. Location also plays a crucial role in the value of an infected device. According to a Trend Micro report of 2014 [27] the price for installing a malicious software on 1 000 machines ranges from \$ 120 – 600 on Australian machines to \$ 10 – 12 on a mix of globally distributed machines. Furthermore, machines located within company or government networks may be significantly more valuable, as they can be used as an entry point for more targeted attacks.

In summary, having an upstream channel is an absolute necessity for many botnet application scenarios and generally increases the profitability of the botnet for the botmaster in any scenario. In the following, we examine why establishing an upstream channel using the blockchain is financially inefficient; afterwards we discuss why alternative upstream channels greatly reduce the benefits of using blockchains for C2.

B. Financial infeasibility of blockchain upstream

Having discussed the need for upstream channels, one could argue that works such as [21], [22] present techniques for using the blockchain for both up- and down-stream communication. However, as discussed by the authors themselves this comes at great cost. In fact, we analyzed the current pricing of transactions on the Bitcoin and Ethereum blockchains resulting in the following equation to estimate the daily cost of running a botnet fully on the blockchain.

Equation 1 presents the general formula based on the transaction fee (tx_fee), number of bots (#bots) and the frequency at which bots contact the botmaster within a day. According to churn measurements of bots by Haas et al. [28], 70% of bots stay online for less than one hour. For this reason, we assume that bots report hourly to the botmaster for our calculations. Based on the transaction costs[5] of Bitcoin and Ethereum, this resulted in daily costs of $\$\,39\,600$ (USD) and $\$\,216\,000$ (USD) respectively for running a botnet with a population of $50\,000$ bots.

$$\text{tx\_fee} \cdot \#\text{bots} \cdot \text{frequency} = \text{daily\_costs} \tag{1}$$

$$ETH: \$\,0.033 \cdot 50\,000 \cdot 24 = \$\,39\,600 \tag{2}$$

$$BTC: \$\,0.18 \cdot 50\,000 \cdot 24 = \$\,216\,000 \tag{3}$$

To put these numbers into perspective, we collected information on how much money botmasters commonly earn. In the book Spam Nation [29], Brian Krebs reports the following incomes for different spam botmasters: Cosma (Rustock Botnet) $\$83\,000/month$, Severa (Storm, Waledac Botnet) $\$4\,000/month$[6], GeRa (Grum Botnet) $\$75\,000/month$. In another report by the US Department of Justice [30], the losses[7] caused by the GameOver Zeus banking trojan are estimated at $\$\,100\,000\,000$ for a period of seven years. However, the size of the botnet was also estimated at $500\,000$ to $1\,000\,000$, which is much larger than the $50\,000$ we used in our example calculations. These results clearly indicate that it is economically infeasible for a botmaster to run a botnet with blockchain upstream based on these costs. Therefore, to establish an upstream from bots to the botmaster, an alternative channel will have to be established.

C. Using alternative upstream channels

Due to the necessity of having an upstream channel for bots, and considering the costs of establishing such a channel on the blockchain, a separate channel as proposed by ZombieCoin [4] is the most viable approach. However, establishing an additional channel for upstream communication introduces additional channel-specific weaknesses and drawbacks. Overall, any previously known botnet C2 channel, e.g. centralized [31], P2P [32], or public resource based [33], can be used to realize an upstream channel.

As these channels facilitate bidirectional communication, they can also be used to establish a cheaper downstream of commands. However, the pseudonymity, enumeration resistance and stealthiness of the entire botnet will be affected based on the choice and characteristics of the alternative channel. While the blockchain provides a high level of pseudonymity to the botmaster, accessing the alternative channel may reveal details about the botmaster. Furthermore, the enumeration resistance will be weakened significantly, as usually anything connecting to traditional C2 infrastructures is part of the botnet. Lastly, if detection techniques for the alternate channel exist, the malware as a whole will be detected, allowing the blockchain C2 protocol to be reverse engineered.

While all of this may deem a blockchain C2 channel obsolete, it still has a very unique strength: *robustness*. Due to being built on top of a public P2P network with the goal of censorship resistance in its design, the blockchain based C2 channel is very difficult to take down. Therefore, it is an excellent backup channel in the case of a compromise of the upstream / main communications mechanism. Previously, botnets such as GameOver Zeus [32] already implemented a DGA-based backup mechanism, as a contingency mechanism if the P2P channel fails. Using the blockchain as a backup channel for a server based C2 channel is also proposed by Curran and Geist [16].

V. Countermeasures

In this section, we investigate potential countermeasures against blockchain-based botnets. First, we look into their stealthiness and how to detect such botnets. Afterwards, we discuss ways to mitigate and neutralize the threat (upon detection).

A. Detection

With regard to the *stealthiness* of blockchain-based botnets, we investigate two hypotheses: (a) *encrypted commands in Bitcoin OP_RETURN transactions are detectable (as anomalies)*; if this is true then it would be possible to reverse-engineer the botnet and decrypt all commands before any attack takes place. (b) *the network traffic of a bot is detectable; if this is true then again we would be able to locate infected machines before they perform any attacks.*

If the answer to the two hypotheses above turns out to be negative, then this would mean that it is impossible to detect a botnet which uses an established cryptocurrency as the communication mechanism only by analyzing blockchain content and network traffic. This would suggest that it would be unlikely to detect the botnet in its waiting stages, i.e., bots waiting for commands to attack. This is a common goal [34], as detecting the malicious activity itself is oftentimes easier but also too late to prevent it.

Note that we focus on Bitcoin specifically. We argue that if the communication channel is not detectable when viewing the

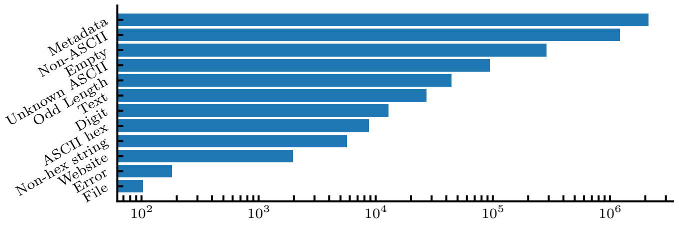
[5] accessed on February 20th 2019

[6] Krebs states in his book that Severa made a lot more money renting out his botnet, but does not state how much.

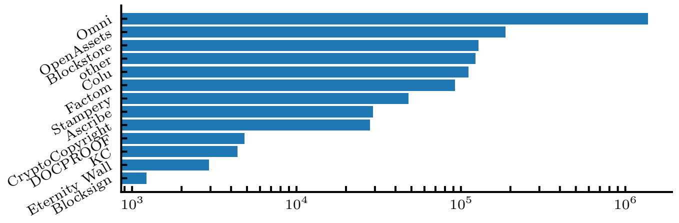
[7] In addition, we argue that the losses do not imply an equivalent income for the botmasters.

Bitcoin blockchain, it is very unlikely that it will be detectable in a platform with much richer functionality, like Ethereum, or in a privacy-preserving cryptocurrency (e.g. *Monero* or *Zcash*).

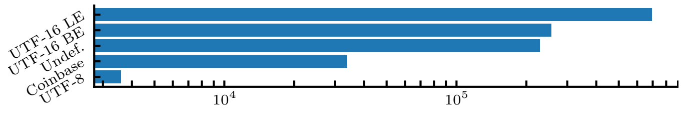
1) Transactions.: To investigate the first hypothesis, we analyzed the outputs of all OP_RETURN Bitcoin transactions until November 2018. In this subsection, we present a summary of our results.



(a) Number of transactions per OP_RETURN category (log x axis)



(b) Metadata services (log x axis)



(c) Content of non-ascii (log x axis)

Fig. 2. OP_RETURN investigation results

The OP_RETURN script was introduced in the Bitcoin transaction specification with core client version 0.9.0 (spring 2014), offering a standard way for storing arbitrary content in the blockchain, with the intention of reducing other non-standard ways of storing data that were used at the time and that bloated the UTXO database (unspent transaction output). Its maximum size was originally planned to be 80 Bytes, but was reduced to 40 for the first compliant core client release (0.9.0). It was then increased to 80 B (0.11.0) and afterwards to 83 Bytes (0.12.0 – remains the same until the time of writing). The size and need for existence of OP_RETURN has repeatedly been the subject of heated discussions in public forums. Since 2015 it is the only technique for adding non-financial data on Bitcoin with non-negligible utilization [35].

To collect and filter all OP_RETURN transactions we developed a python script, building on the script offered in [36], that we have made publicly available with a permissive license on github[8]. In total, we ended up with a database of 3 815 944 transactions. Overall, only $\sim 20\%$ of blocks mined contained an OP_RETURN transaction, while OP_RETURN transactions represent $\sim$ 1% of the total number of transactions in the Bitcoin blockchain. Moving a step further, we classified the content of these transactions by applying several filters and subsequently checking for known services by using web search engines, looking into associated forums (e.g. Reddit), and related prior work [36]. The content was attributed to 12 main categories, as shown in Figure 2a.

Metadata, corresponding to known services that use the Bitcoin blockchain (see a summary in Fig. 2b) represent the majority of transactions. In previous work, the authors of [36] focused on the metadata services, while those of [35] on visible illegal content (pictures, etc.). Contrary to them, we move on to examine other categories that could potentially be compatible with a botnet C2, i.e. resembling a covert channel. To keep the C2 channel as secret as possible, we do not consider issuing commands with the syntax of other services classified as metadata. This is based on the assumption, that service providers monitor OP_RETURN entries matching their own syntax. Therefore, entries not originating from the service providers may be suspicious and trigger further investigation.

After filtering all content that could be associated with different use-cases (e.g. text excerpts in foreign languages), we end up with 319 784 transactions that could match a covert channel, i.e. they looked like encrypted messages. These include content that could be decoded in ASCII or UTF (see also Figure 2c) but looked random, and content that could not be decoded and showed no patterns. The OP_RETURN fields of those transactions are therefore potentially outputs of cryptographic operations (hashes, encryption, signatures). Of those transactions, 47 592 were issued by two addresses and had a common receiver address (we could not find any notable pattern in the rest). The OP_RETURN content in the first case is always 40 Bytes, while in the second always 20 Bytes. Since no additional pattern or interesting repetition was noted in the content, we had to stop our investigation there.

We plot the per day number of transactions with unknown content against transactions originating from known services for the years 2017 and 2018 in Figure 3 (note logarithmic y axis). We can see an average of around 5 000 OP_RETURN transactions per day, of which at least 100 are unknown. Therefore, we conclude that when sending a small number of commands per day, the activity should not be detectable in the Bitcoin blockchain (as there is no observable anomaly in the number of unknown OP_RETURN transactions. In cases where a large number of commands has to be sent at once, subliminal channels (as e.g. proposed in [19]) may be required.

2) Network Traffic.: To investigate the second hypothesis (*the network traffic of a bot is detectable*), we performed the following experiments:

First, we set up[9] a Bitcoin full node (Bitcoin Core v. 0.16.2) with default settings, synced to the network and subsequently captured its network traffic over 7 hours. As expected, bandwidth was relatively stable over time, averaging 5 kB/sec. This

[8]https://anonymous.4open.science/repository/8201c218-d03a-4ab4-98ee-0e76c6f6d29d/ (anonymous version)

[9]Our setup included a laptop with an Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz and 16GB DDR3L of RAM. Not attempting to generalize (but only to provide the reader with a hint regarding the computational overhead), we note that throughout our measurement period the CPU utilization remained between $1.0 - 1.2\%$.

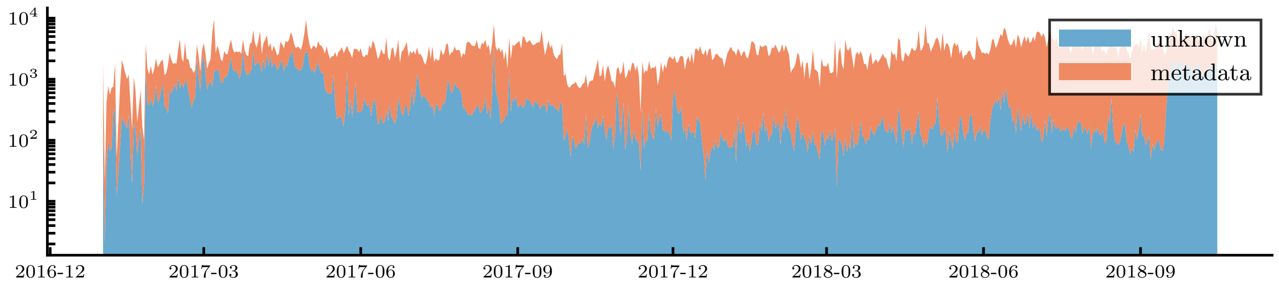


Fig. 3. Time series of metadata and unknown transactions per day (log y axis)

would result in 300 kB/min, or 3 MB/block if we assume the average block creation time for Bitcoin is 10 min. The maximum Bitcoin block size is set to 1 MB (this might change in the near future), so on average the total network traffic of a node per block epoch is 3 times the size of the block. Due to the relatively low bandwidth, we consider this kind of traffic difficult to detect, since it is unlikely to cause disruptions to the users.

Second, we repeated the experiment using a light Bitcoin client in the form of an Electrum wallet[10]. Light nodes only receive the block headers and query for specific addresses they want to know more about. As a response, they receive the transactions they are interested it, along with a short (logarithmic in the size of the block) proof that the transaction actually is part of the block. Since they already have the block headers, they can be sure that the transaction is part of the blockchain. The average bandwidth measured in this occasion was around 0.2 kB/sec, or 4% of the bandwidth consumed by the full node. Hence, a light node would be very difficult to detect and feasible to run on low bandwidth devices. From a network perspective, it would be completely indistinguishable from a mobile cryptocurrency wallet.

Since we already referred to the network bandwidth as a potential resource that could limit the feasibility of such a botnet running on a constrained device, we take this opportunity to expand on this topic a bit more. According to Ali et al. [4] a light client for Bitcoin requires approximately 7 MB of storage space. Comparatively, the Hide'n'Seek P2P botnet has a total binary size of 71 kB. To put this into perspective, an ESP 32, one of the most popular micro-controllers for connected actuators and sensors, is usually equipped with 4-16 MB of flash storage.

Assuming that cheap off-the-shelf IoT devices probably have similar or less memory, running common light clients may not be feasible. However, considering that the bots are only interested in the commands, a lot of functionality could be stripped from existing light clients to make them compatible with resource constrained IoT devices. Furthermore, given the low network traffic of a light client (see previous paragraphs), the processor of the ESP 32 should be capable of handling the traffic and extracting the commands (while ignoring the verification of transactions).

[10]https://electrum.org/#home

B. *Mitigation*

At some point, either before or after being utilized, a botnet will inevitably be detected and reverse-engineered. Here, we investigate methods for taking down such a botnet.

One approach that comes to mind is banning any blockchain related traffic within a controlled network. While such an approach may be effective, we argue that its application is limited. Especially taking into account that in the future it is highly probable that a significant number of ordinary users will utilize a cryptocurrency wallet. Therefore, banning all blockchain traffic is not a applicable solution.

A second approach frequently proposed in reports on Bitcoin-related criminal activities, is "regulating" Bitcoin by utilizing a blacklist mechanism. This, however, is both technically and practically infeasible. Technically, a botmaster could disseminate hundreds of potential addresses from which bots will expect commands from. Some of these addresses may belong to users not related to the botnet at all. Hence, it is not technically possible to selectively block future botmaster addresses. Note that including addresses not belonging to the botmaster will not allow the owners of those addresses to control the botnet, as basic public key authentication mechanisms can guarantee that bots only obey their master.

Practically, the initiation of an address blacklist on Bitcoin would go against its whole philosophy and would potentially cause great turbulence in the community (e.g. see Ethereum's hard fork after the DAO [37]). Therefore, it is extremely unlikely that such a practice would be introduced to address common criminal activity performed by botmasters.

Taking into account the above discussion, using the blockchain as a backup channel will render any attempts to suppress the control channel (e.g. sinkholing P2P botnets or seizing centralized servers) futile. The botmaster could simply broadcast a new C2 server whenever their previous one is compromised. Consequently, the community needs to develop better defenses on the network edge, meaning mechanisms for detecting and disinfecting machines on a local network level. Furthermore, collaboration between peers of the blockchain network that act like full nodes and serve light clients would be invaluable in enumerating bots.

VI. Related Work

In this section we summarize the related work with regard to botnet research as well as blockchain metadata analysis.

A. Botnet C2 Channels

While blockchain-based C2 channels are a fairly new concept, many other structures and services are being used by botmasters to command their bot armies.

The easiest and most straightforward approach to implement a C2 infrastructure are centralized client server models. Historically, IRC and HTTP-based botnets such as Rustock [31], Dorkbot [38], and Conficker [39] were very popular among botmasters. However, such botnets have a single point of failure in the central server and once detected can be taken down [40]. They also provide minimal anonymity to botmasters, as they have to connect to the same C2 server frequently.

To safeguard their control infrastructure, botmasters started to implement techniques to hide the main C2 servers. Fast flux networks [41] make use of round-robin DNS to resolve a domain name to multiple entries. Behind each of theses entries is a bot, which redirects the traffic to a command and control server. This effectively hides the IP address of the C2 server and therefore improves the robustness, anonymity and enumeration resistance of centralized botnets.

Another approach to increase the robustness of botnets is the use of Domain Generation Algorithms (DGAs) [39], [32]. The main goal of a DGA is to increase robustness against seizing of domain names or blacklisting of domains. Once the control structure is not reachable anymore under a given domain, bots will start to run the DGA and contact the computed domains. The botmaster can similarly compute the domains and register any of them and link it to the C2 structure. Eventually, the bots will connect to that new domain and receive commands again.

Some highly advanced botnets such as GameOver Zeus, Sality or ZeroAccess [32], [42], [43] use a C2 channel based on custom unstructured P2P protocols. In such an approach every bot can function as a C2 server and forward commands injected by the botmasters. This makes it highly resilient against any kind of takedown attempt, as the entire bot population has to be taken down, or their connections severed. However, this requires accurate intelligence about the botnet's structure and an exploitable flaw in the protocol. Furthermore, several works discuss possible means to impede intelligence gathering attempts [44], [45], [46], [47], [7]. For these reasons, P2P botnets provide great anonymity and robustness to its botmasters, but are highly complex to implement with custom protocols.

Moreover, a number of botnets [33], [48], [49] exist that use public resources, e.g. Twitter, to relay their commands to bots. The main benefit of this approach is that it blends in well with regular Internet traffic. This makes it more difficult to detect and block on a network level. However, due to recent issues with social bots meddling in elections, companies, such as Twitter, have started to increase their effort in detecting and removing bots from their services [9]. While botnet C2 is different from typical social bots, they may be affected by the cleanup processes as well. Generally, using public services for the C2 makes the botnet vulnerable to be shutdown swiftly by the service providers if the botnet becomes known.

Lastly, the option of using the Tor network to relay C2 commands is discussed in research papers and used by real botnets [50], [51], [52]. The main goal of this approach is to have the simplicity of a centralized C2 hidden within the tor network. However, Casenove et al. [8] highlight several issues regarding the resilience, stealthiness and scalability of Tor-based botnets. Furthermore, even though the bots connect to the C2 through Tor, the traffic at the C2 itself could be observed on an Internet Service Provider (ISP) level.

B. Blockchain Meta-data analysis

Regarding work on exploring and analyzing non-financial transaction data on blockchains, in [36] (and the extended version [53]), the authors analyze metadata associated with different services utilizing the Bitcoin blockchain (e.g. financial, notary services). This work provides a good understanding of the service ecosystem around Bitcoin. In [35], the authors focus on identifiable content not associated with known services, with the aim of assessing whether objectively illegal content is part of the Bitcoin blockchain. They focus on text and photographs and conclude that only very few instances of such content exists, although some are quite serious (mainly links to child pornography). In [54], the same authors discuss potential countermeasures, such as mandatory transaction fees or even cryptographic proofs that inserted content exclusively consists of financial transactions (and no metadata can potentially be inserted, not even via subliminal channels). In a paper by Kuzuno et al. [55], they propose Blockchain Explorer. Their system is aimed at at providing a process and tool for law enforcement investigation of bitcoin transactions. They showcase how their approach can be used to identify bitcoin addresses related to illegal marketplace transactions, ransomware payments and DDoS extortion cases. Contrary to the works above, we neither investigate the services using the blockchain, nor look for identifiable illegal content. Instead, we focus on the existence and frequency of non-identifiable content (possibly encrypted) that could be associated with botnet C2 (see Section V-A).

VII. Conclusion

In this paper we provide a detailed analysis of the threat posed by blockchain-based botnets. We use Bitcoin as our point of reference, but our results should hold for any other existing and upcoming blockchain with the same goals. While most of the works on bitcoin based botnets highlight the theoretical strength of the blockchain as a C2 channel, they neglect its major drawback of it being only unidirectional. To provide a more comprehensive overview, Table II summarizes and compares the strength and weaknesses of blockchain-based C2 with alternative channels discussed in Section VI.

We conclude that due to the lack of (cost-effective) bidirectionality the blockchain excels as a backup mechanism, as it is the most difficult to ever be blocked. In fact, combining it with any of the other mechanisms will render attacks against the C2 futile, as the blockchain channel can always be used

C2 Channel	Anonymity	Stealthiness	Robustness	Enum. Resistance	Bidirectional	Simplicity
Blockchain						
Blockchain Bidirectional	●	◐	●	○	●	○
Blockchain Downstream	●	◐	●	●	○	◐
Blockchain as Backup	—	—	●	—	—	◐
Client Server						
Central Server (e.g. HTTP)	○	◐	○	○	●	●
Fast Flux	◐	◐	◐	◐	●	◐
DGA	◐	○	◐	◐	●	◐
Tor	●	○	◐	◐	●	◐
Public Resource (e.g. Twitter)	◐	◐	○	◐	●	◐
Peer-to-Peer	●	◐	●	◐	●	○

TABLE II
COMPARISON OF PROPERTIES BETWEEN DIFFERENT C2 CHANNELS: ●= PROVIDES PROPERTY, ◐= PARTIALLY PROVIDES PROPERTY, ○= DOES NOT PROVIDE PROPERTY, — = NOT APPLICABLE (DEPENDS ON THE MAIN CHANNEL).

to reorganize the botnet. Therefore, the community should focus more on detection and defense mechanisms targeting edge networks or individual infections. Such approaches could ultimately strip botmasters of their most valuable resource, the bots themselves. Overall, the blockchain may provide an advantage to botmasters, but as in most cases, there exist countermeasures that do not involve "regulating" cryptocurrencies. Lastly, our empirical study's results can be summarized in the following:

- *Will we ever see a fully autonomous blockchain-based botnet*? Probably not.
- *Will we see more botnets abusing the blockchain as a backup channel?* Definitely.

ACKNOWLEDGMENTS

This work was supported by the Royal Bank of Canada within the project Novel P2P Botnet Detection under Grant Agreement 2761478.4. This work was also supported by the German Federal Ministry of Education and Research (BMBF) as well as by the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within CRISP.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] S. Foley, J. R. Karlsen, and T. J. Putniņš, "Sex, drugs, and bitcoin: how much illegal activity is financed through cryptocurrencies?" *Review of Financial Studies, Forthcoming*, 2018.

[3] S. T. Ali, P. McCorry, P. H.-J. Lee, and F. Hao, "Zombiecoin: powering next-generation botnets with bitcoin," in *International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 34–48.

[4] ——, "Zombiecoin 2.0: managing next-generation botnets using bitcoin," *International Journal of Information Security*, vol. 17, no. 4, pp. 411–422, 2018.

[5] D. Roffel and C. Garrett, "Using the Blockchain as a C&C for a botnet," 2014, accessed: 2019-02-20. [Online]. Available: https://www.reddit.com/r/netsec/comments/2pmmfu/using_the_blockchain_as_a_cc_for_a_botnet/

[6] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos, "Sok: P2pwned - modeling and evaluating the resilience of peer-to-peer botnets," in *2013 IEEE Symposium on Security and Privacy*, May 2013, pp. 97–111.

[7] L. Böck, E. Vasilomanolakis, M. Mühlhäuser, and S. Karuppayah, "Next generation p2p botnets: monitoring under adverse conditions," in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2018, pp. 511–531.

[8] M. Casenove and A. Miraglia, "Botnet over tor: The illusion of hiding," in *2014 6th International Conference On Cyber Conflict (CyCon 2014)*. IEEE, 2014, pp. 273–282.

[9] "Twitter is sweeping out fake accounts like never before, putting user growth at risk," https://www.washingtonpost.com/technology/2018/07/06/twitter-is-sweeping-out-fake-accounts-like-never-before-putting-user-growth-risk/, 2018, accessed: 2019-02-22.

[10] S. Herwig, K. Harvey, G. Hughey, R. Roberts, and D. Levin, "Measurement and Analysis of Hajime, a Peer-to-peer IoT Botnet," in *Network and Distributed System Security Symposium*, 2019. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/measurement-and-analysis-of-hajime-a-peer-to-peer-iot-botnet/

[11] I. Ilascu, "New Botnet Hides in Blockchain DNS Mist and Removes Cryptominer," 2018. [Online]. Available: https://www.bleepingcomputer.com/news/security/new-botnet-hides-in-blockchain-dns-mist-and-removes-cryptominer/

[12] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *USENIX Security Symposium*, 2017, pp. 1092–1110.

[13] E. Karaarslan and E. Adiguzel, "Blockchain based dns and pki solutions," *IEEE Communications Standards Magazine*, vol. 2, no. 3, pp. 52–57, SEPTEMBER 2018.

[14] J. Spooren, D. Preuveneers, L. Desmet, P. Janssen, and W. Joosen, "Detection of algorithmically generated domain names used by botnets: A dual arms race," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC '19. New York, NY, USA: ACM, 2019, pp. 1916–1923. [Online]. Available: http://doi.acm.org/10.1145/3297280.3297467

[15] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon, "From throw-away traffic to bots: detecting the rise of dga-based malware," in *Presented as part of the 21st USENIX Security Symposium USENIX Security 12)*, 2012, pp. 491–506.

[16] T. Curran and D. Geist, "Using the bitcoin blockchain as a botnet resilience mechanism," 2016. [Online]. Available: https://www.os3.nl/_media/2016-2017/courses/ot/dana_tom.pdf

[17] G. J. Simmons, "The prisoners' problem and the subliminal channel," in *Advances in Cryptology*. Springer, 1984, pp. 51–67.

[18] ——, "The subliminal channel and digital signatures," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1984, pp. 364–378.

[19] D. Frkat, R. Annessi, and T. Zseby, "Chainchannels: Private botnet communication over public blockchains," 2018. [Online]. Available: https://www.annessi.net/data/2018-subliminalblockchain_preprint.pdf

[20] G. Falco, C. Li, P. Fedorov, C. Caldera, R. Arora, and K. Jackson, "Neuromesh: Iot security enabled by a blockchain powered botnet vaccine," in *Proceedings of the International Conference on Omni-Layer Intelligent Systems*, ser. COINS '19. New York, NY, USA: ACM, 2019, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/3312614.3312615

[21] N. A. I. Majid A. Malaika, Omar Al Ibrahim, "Botract: Abusing Smart Contracts and Blockchains for Botnet Command and Control," https://www.omprotect.com/wp-content/uploads/2017/12/BotDraftPaper-v1.pdf, 2017, accessed: 2019-02-13.

[22] O. Zohar, "Unblockable Chains," accessed: 2019-02-13. [Online]. Available: https://github.com/platdrag/UnblockableChains

[23] J. Sweeny, "Botnet resiliency via private blockchains," SANS Institute Information Security Reading Group, 2017. [Online]. Available: https://www.sans.org/reading-room/whitepapers/covert/paper/38050

[24] "The Bitocoin Wiki," https://en.bitcoin.it/wiki, accessed: 2019-02-20.

[25] N. Van Saberhagen, "Cryptonote v 2.0," 2013. [Online]. Available: https://whitepaperdatabase.com/wp-content/uploads/2017/09/Monero-whitepaper.pdf

[26] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin,"

in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 459–474.

[27] M. Goncharov, "Russian underground revisited," *Cybercriminal Underground Economy Series*, 2014.

[28] S. Haas, S. Karuppayah, S. Manickam, M. Mühlhäuser, and M. Fischer, "On the resilience of p2p-based botnet graphs," in *2016 IEEE Conference on Communications and Network Security (CNS)*, Oct 2016, pp. 225–233.

[29] B. Krebs, *Spam nation: The inside story of organized cybercrime-from global epidemic to your front door*. Sourcebooks, Inc., 2014.

[30] "U.S. Leads Multi-National Action Against "Gameover Zeus" Botnet and "Cryptolocker" Ransomware, Charges Botnet Administrator," http://www.justice.gov/opa/pr/us-leads-multi-national-action-against-gameover-zeus-botnet-and-cryptolocker-ransomware, 2014, accessed: 2019-02-21.

[31] K. Chiang and L. Lloyd, "A Case Study of the Rustock Rootkit and Spam Bot."

[32] D. Andriesse, C. Rossow, B. Stone-Gross, D. Plohmann, and H. Bos, "Highly resilient peer-to-peer botnets are here: An analysis of gameover zeus," in *2013 8th International Conference on Malicious and Unwanted Software: "The Americas" (MALWARE)*, vol. 00, Oct. 2014, pp. 116–123.

[33] B. Prince, "Flashback botnet updated to include twitter as c&c," 2012. [Online]. Available: http://www.securityweek.com/flashback-botnet-updated-include-twitter-cc

[34] H. Hang, X. Wei, M. Faloutsos, and T. Eliassi-Rad, "Entelecheia: Detecting p2p botnets in their waiting stage," in *2013 IFIP Networking Conference*. IEEE, 2013, pp. 1–9.

[35] R. Matzutt, J. Hiller, M. Henze, J. H. Ziegeldorf, D. Müllmann, O. Hohlfeld, and K. Wehrle, "A quantitative analysis of the impact of arbitrary blockchain content on bitcoin," in *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC). Springer*, 2018.

[36] M. Bartoletti and L. Pompianu, "An analysis of bitcoin op_return metadata," in *Financial Cryptography and Data Security - FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers*, 2017, pp. 218–230. [Online]. Available: https://doi.org/10.1007/978-3-319-70278-0_14

[37] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts (sok)," in *International Conference on Principles of Security and Trust*. Springer, 2017, pp. 164–186.

[38] B. Irinco, "The dorkbot rises," http://blog.trendmicro.com/trendlabs-security-intelligence/the-dorkbot-rises, Oct. 2012.

[39] M. J. v. E. Hadi Asghari, Michael Ciere, "Post-mortem of a zombie: Conficker cleanup after six years," https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-asghari.pdf, Aug. 2015.

[40] S. Greengard, "The war against botnets," *Communications of the ACM*, vol. 55, no. 2, p. 16, feb 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2076450.2076456

[41] J. Nazario and T. Holz, "As the net churns: Fast-flux botnet observations," in *2008 3rd International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE, 2008, pp. 24–31.

[42] N. Falliere, "Sality: Story of a peer-to-peer viral network," Symantec Corporation, Tech. Rep., 2011.

[43] A. Neville and R. Gibb, "ZeroAccess Indepth," Symantec Security, Tech. Rep., 2013.

[44] S. Karuppayah, E. Vasilomanolakis, S. Haas, M. Muhlhauser, and M. Fischer, "BoobyTrap: On autonomously detecting and characterizing crawlers in P2P botnets," *IEEE International Conference on Communications, ICC*, 2016.

[45] E. Vasilomanolakis, J. H. Wolf, L. Böck, S. Karuppayah, and M. Mühlhäuser, "I trust my zombies: A trust-enabled botnet," *arXiv preprint arXiv:1712.03713*, 2017.

[46] D. Andriesse, C. Rossow, and H. Bos, "Reliable recon in adversarial peer-to-peer botnets," in *Proceedings of the 2015 Internet Measurement Conference*, ser. IMC '15. New York, NY, USA: ACM, 2015, pp. 129–140.

[47] L. Böck, E. Vasilomanolakis, J. H. Wolf, and M. Mühlhäuser, "Autonomously detecting sensors in fully distributed botnets," *Computers & Security*, vol. 83, pp. 1 – 13, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404818312094

[48] P. Paganini, "F-secure has discovered miniduke malware samples in the wild," http://securityaffairs.co/wordpress/23658/cyber-crime/f-secure-new-miniduke-atp.html, Apr. 2014, accessed: 2019-02-22.

[49] A. Lelli, "Trojan.whitewell: what's your (bot) facebook status today?" 2009, accessed: 2019-02-22. [Online]. Available: http://www.symantec.com/connect/blogs/trojanwhitewell-what-s-your-bot-facebook-status-today

[50] C. Guarnieri and M. Schloesser, "A Tor-powered botnet straight from Reddit," https://blog.rapid7.com/2012/12/06/skynet-a-tor-powered-botnet-straight-from-reddit/, accessed: 2019-02-22.

[51] A. Matrosov, "The rise of tor-based botnets," http://www.welivesecurity.com/2013/07/24/the-rise-of-tor-based-botnets/, Jul. 2013, accessed: 2019-02-22.

[52] D. Brown, "Resilient botnet command and control with tor," *DEF CON*, vol. 18, 2010.

[53] M. Bartoletti, B. Bellomy, and L. Pompianu, "A journey into bitcoin metadata," *Journal of Grid Computing*, Jan 2019. [Online]. Available: https://doi.org/10.1007/s10723-019-09473-3

[54] R. Matzutt, M. Henze, J. H. Ziegeldorf, J. Hiller, and K. Wehrle, "Thwarting unwanted blockchain content insertion," in *2018 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 2018, pp. 364–370.

[55] H. Kuzuno and C. Karam, "Blockchain explorer: An analytical process and investigation environment for bitcoin," in *2017 APWG Symposium on Electronic Crime Research, eCrime 2017, Phoenix, AZ, USA, April 25-27, 2017*, 2017, pp. 9–16. [Online]. Available: https://doi.org/10.1109/ECRIME.2017.7945049