Politecnico di Milano
Dipartimento di Elettronica, Informazione e Bioingegneria
LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

BitIodine: Extracting Intelligence from the Bitcoin Network

Thesis of:
Michele Spagnuolo

Matricola:
778935

Advisor (Politecnico di Milano):
Prof. Stefano Zanero

Advisor (Politecnico di Torino):
Prof. Antonio Lioy

A.A. 2012 – 2013

My first debt of gratitude is due to my advisor Stefano Zanero for his help, guidance and general kindness. I owe him my heartfelt appreciation. Discussions with Stefano and Federico Maggi greatly contributed to the exposition and development of this thesis and material submitted to conferences and workshops.

I would like to thank the following friends, fellow students and teachers who have been source of thoughts, inspiration and smiles. In random order (the list was scrambled using randomness coming from atmospheric noise): Francesca Prosperuzzi, Edoardo Colombo, Martina Cividini, Elena Porta, Alessandra Piatti, Francesca Elisa Diletta Raimondi, Francesca Ragni, Matteo Paglino, Mario Sangiorgio, Diego Martinoia, Stefano Schiavoni, Gabriele Petronella, Matteo Serva, Marco D. Santambrogio, Daniele Gallingani, Cristina Palamini, Alessandro Barenghi, Matteo Turri, Luciano Righi, Giuseppina Ferolo, Alberto Scolari, Giovanni Gonzaga, Luca Cioria.

Special thanks to Giorgio Cavaggion, with whom I shared the Chicago experience, for his amazing awesomeness.

Thanks to the awesome people at NECSTLab in Milan, and to the Bitcoin, Gephi and Python communities.

A sincere thank you to Giuseppe Galano and his colleagues at Banca d'Italia for allowing me to present my work in Rome and their particular kindness.

A *thank you* to the mysterious Satoshi Nakamoto, the mastermind behind Bitcoin.

I wish to thank my parents, Laura and Pasquale, and a human being called Liz, who simply is the best thing that ever happened to me.

Finally, I would like to dedicate this work to my grandma Carla.

MS

To anyone who ever taught me,
and to anyone I ever learned from.

Abstract

Bitcoin, the famous peer-to-peer, decentralized electronic currency system, allows users to benefit from pseudonymity, by generating an arbitrary number of aliases (or addresses) to move funds. However, the complete history of all transactions ever performed in the Bitcoin network, called "blockchain", is public and replicated on each node. The data contained into the network is difficult to analyze manually, but can yield a high number of relevant information.

In this thesis we present a modular framework, BitIodine, which parses the blockchain, clusters addresses that are likely to belong to a same user or group of users, classifies such users and labels them, and finally visualizes complex information extracted from the Bitcoin network.

BitIodine allows to label users automatically or semi-automatically with information on who they are and what they do, thanks to several web scrapers that incrementally update lists of addresses belonging to known identities, and that connect information from trades recorded in exchanges, thus allowing to trace money entering and exiting the Bitcoin economy. BitIodine also supports manual investigation by finding paths and reverse paths between two addresses or a user and an address.

We test BitIodine on several real-world use cases. For instance, we find a connection between the founder of the Silk Road, the famous black market operating in Bitcoin, and an address with a balance exceeding 111,114 BTC, likely belonging to the encrypted Silk Road cold wallet. In another example, we investigate the CryptoLocker ransomware, a malware that encrypts the victim's personal files with strong encryption, asking for a ransom to be paid in order to release the files. Starting by an address posted on a forum by a victim, we accurately quantify the number of ransoms paid and get information about the victims.

We release BitIodine to allow the community of researchers to enhance it, thanks to its modular infrastructure. Our hope is that it can become the skeleton for building more complex frameworks for Bitcoin forensic analysis.

A publication based on this work is currently under review by the program committee of an international conference about security, cryptography and finance[1].

[1] We can not disclose the name of the conference until the review phase is over.

Ampio estratto

Bitcoin, una moneta elettronica decentralizzata basata su una rete peer-to-peer e un protocollo open source, consente di offrire un certo grado di anonimia ai suoi utenti attraverso la *pseudonimia*, ossia generando un numero arbitrario di *alias* (o *indirizzi*) per effettuare pagamenti. Tuttavia, la storia completa di tutte le transazioni effettuate nella rete Bitcoin, chiamata "blockchain", è pubblica ed è completamente replicata su ogni nodo della rete. È difficile analizzare manualmente la mole di informazioni contenuta in questo grande registro elettronico distribuito, ma effettuare *mining* automatico o semi-automatico della blockchain consente di estrapolare informazioni interessanti.

In questa tesi presentiamo BitIodine, un framework modulare ed estensibile che analizza la blockchain, raggruppa indirizzi che potrebbero appartenere ad uno stesso utente o gruppo di utenti in *cluster*, li classifica ed etichetta, ed infine visualizza informazioni elaborate.

BitIodine permette di etichettare le entità della rete Bitcoin in modo automatico o semi-automatico con informazioni su chi sono e che cosa fanno, grazie a diversi *web scrapers* che in modo incrementale aggiornano elenchi di indirizzi appartenenti a identità conosciute, registrando anche transazioni in siti che consentono di acquistare o vendere Bitcoin in valuta reale (*exchange*), permettendo così di rintracciare fondi in entrata e in uscita dall'economia Bitcoin. BitIodine supporta anche l'investigazione manuale da parte di un utente esperto, trovando percorsi, diretti o inversi, fra utenti e indirizzi (e viceversa).

Testiamo BitIodine in diversi casi d'uso del mondo reale. Per esempio, troviamo una connessione tra Dread Pirate Roberts, creatore della Silk Road, il più grande mercato nero operante in Bitcoin, e un indirizzo con un saldo superiore a 111.114 BTC, probabilmente appartenenti al *cold wallet* cifrato della Silk Road stessa. BitIodine ci consente di analizzare l'attività riguardante il malware CryptoLocker, che cifra i file personali e chiede alle vittime un riscatto in Bitcoin. A partire da un indirizzo postato su un forum di una vittima, riusciamo a quantificare con precisione il numero di riscatti pagati, ottenendo informazioni sulle persone infettate dal malware.

Rilasciamo il codice sorgente di BitIodine per consentire alla comunità dei ricercatori di migliorarlo, grazie alla sua infrastruttura modulare, credendo possa

diventare una base per la costruzione di strutture più complesse per l'analisi forense della rete Bitcoin.

In sintesi, i nostri contributi originali sono:

- Forniamo un framework modulare ed estensibile che supporti applicazioni complesse per l'analisi forense della rete Bitcoin.
- Etichettiamo automaticamente cluster ed entità della rete, con supervisione limitata o nulla.
- Testiamo BitIodine in casi d'uso del mondo reale, come ad esempio investigazioni della Silk Road e pagamenti legati a malware come CryptoLocker.

Una pubblicazione basata su questo lavoro è attualmente in fase di revisione da parte del *program committee* di una conferenza internazionale in materia di sicurezza, crittografia ed aspetti finanziari[2].

Dopo l'introduzione, nel Chapter 2 illustriamo lo stato dell'arte per quanto riguarda Bitcoin e la sua adozione nel mercato e per usi illeciti, come la monetizzazione di malware. Analizziamo inoltre il complesso problema della privacy in Bitcoin, evidenziandone i punti deboli.

Nel Chapter 3, dopo aver presentato la terminologia caratteristica di Bitcoin, illustriamo l'architettura del nostro framework, BitIodine, descrivendone in dettaglio i moduli che lo compongono.

Il Chapter 4 è dedicato all'implementazione, alla definizione formale delle euristiche utilizzate da BitIodine e da altri aspetti tecnici, come la struttura del database e dei grafi.

I casi d'uso sono esposti nel Chapter 5.

Nel primo dimostriamo che un indirizzo appartiene a Silk Road. Nel secondo mostriamo una connessione fra il suo fondatore e un indirizzo contenente alcune decine di milioni di dollari, verosimilmente appartenenti al *cold wallet*

[2]Non ci è consentito specificare il nome della conferenza durante la fase di revisione dei paper.

cifrato della Silk Road stessa. Nel terzo individuiamo la transazione che, secondo l'FBI, sarebbe un pagamento per un assassinio su commissione. Il quarto è dedicato al primo acquisto di pizze in Bitcoin nel 2010, una transazione storicamente rilevante. Nel quinto ed ultimo investighiamo l'attività riguardante il malware CryptoLocker.

Le limitazioni del nostro approccio sono delineate nel Chapter 6, seguite dalle conclusioni.

Contents

Introduction 1

Bitcoin is a decentralized monetary system that aims to become the digital equivalent of cash. Like cash, Bitcoin transactions do not explicitly identify the payer nor the payee: a transaction is just a cryptographically signed message that embodies a transfer of funds from one public key to another. The corresponding private keys are needed to authorize a fund transfer. Based on an open source protocol and a peer-to-peer network of participants that validates and certifies all transactions, Bitcoin has recently received important media coverage [13, 11, 12], mostly due to its conversion rate to dollars surging [13, 12], and to the Bitcoin economy expanding at unprecedented pace [13], with the economic crisis dominating the global scenario. Some features of Bitcoin, such as cryptographically guaranteed security of transactions, negligible transaction fees, no set-up costs and no risk of charge-back convinced several businesses around the world (such as Wordpress, Foodler, Howard Johnson, OkCupid, Namecheap and Private Internet Access, to name a few) to adopt it as an alternative payment method. At the same time, its apparent anonymity, privacy features, the fact transactions are irrevocable and its ease of use attracted also cybercriminals [18], who use Bitcoin as a way of monetizing botnets and extorting money with malware such as CryptoLocker, that encrypts personal files and asks for a ransom to be paid in order to receive the decryption key, which is saved on a remote server.

The decentralized accounting paradigm typical of Bitcoin requires each node of the network to keep in memory the entire transaction history, a public ledger of every transaction ever happened, called *blockchain*. While Bitcoin identities are not explicitly tied to real-world individuals or organizations, all transactions are public and transparent. This is a problem, when it comes to anonymity: anyone can see the flow of Bitcoin from address to address. This means all transactions are conducted in public, and each one is tied to the preceding one. In a sense, this makes Bitcoin much less private than cash. If a user chooses to engage in sensitive transactions on Bitcoin, (s)he should be aware that a public record will be preserved forever.

There is a lot of interesting information to be mined out of the blockchain. Some addresses are known and tied to entities such as gambling sites, users of the main Bitcoin-related forum, Bitcoin Talk, or Bitcoin-OTC marketplace. By analyzing the blockchain and correlating it with this publicly available meta data, it is possible to find out how much an address is used for gambling activities or mining, if it was used for scamming users in the past, if and how it is related to other addresses and entities. Addresses can be algorithmically grouped in clusters that correspond with entities that control them (but do not necessarily *own* them) [2, 4, 18, 28]. We will hereinafter refer to such clusters and entities interchangeably as *users*. The interesting outcome for investigators is that it is possible to retrieve valuable information about an entity by just knowing one of its addresses. Collapsing addresses into clusters compacts and simplifies the huge transaction graph, creating edges between users that correspond to aggregate transactions. In other words, with this approach it is possible to move out of the way the complexity of apparently anonymous transactions between meaningless addresses, and make money exchanges between entities visible. In summary, in existing approaches clusters are labeled mostly manually, and the whole process is not automated.

In this thesis we propose BitIodine, a collection of modules to automatically parse the blockchain, cluster addresses, classify addresses and users, graph, export and visualize elaborated information from the Bitcoin network. In particular, we devise and implement a *Classifier* module that labels the clusters in an automated or semi-automated way, by using several web scrapers that incrementally update lists of addresses belonging to known identities. We create a feature-oriented database that allows fast queries about any particular address to retrieve balance, number of transactions, amount received, amount sent, and ratio of activity concerning labels (e.g., gambling, mining, exchanges, donations, freebies, malware, FBI, Silk Road), or, in an aggregated form, for clusters. It is possible to query for recently active addresses, and filter results using cross filters in an efficient way.

BitIodine has been tested on several real-world use cases. In this thesis, we describe how we used BitIodine to find the transaction that, according to the FBI, was a payment by Dread Pirate Roberts, founder of the Silk Road, to a hitman to have a person killed [25]. We find a connection between Dread Pirate Roberts and an address with a balance exceeding 111,114 BTC[1], likely belonging to the encrypted Silk Road cold wallet. Finally, we investigate the CryptoLocker ransomware, and, starting by an address posted on a forum by a victim, we accurately quantify the number of ransoms paid (around 375.93 BTC as of November 1, 2013), and get information about the victims.

In summary, our contributions are:

- We provide a modular framework for building complex applications for forensic analysis of the Bitcoin blockchain, which is easily expandable and future-proof.
- We automatically label clusters/users with limited to no supervision.
- We test our framework on real-world use cases that include investigations on the Silk Road and on malware such as CryptoLocker.

[1] The common shorthand currency notation for Bitcoin(s)

State of the art and motivation 2

2.1 Background

2.1.1 An overview of Bitcoin

While Bitcoin can be seen as a *trust-no-one* currency that isn't subject to manipulation by central banks or corporations, from a more technical point of view it is a payment system written from scratch and based on the very best cryptography, designed for security. The Bitcoin protocol has been designed to be forward compatible and *future proof* (more details in the following subsection).

Open source and resilient to attacks, having no single point of failure, it is very difficult to take down.

There are several advantages of using Bitcoin as a means of exchange:

Speed and price It is possible to transfer money anywhere in the world within minutes with negligible fees.

No central authority Bitcoin is not dependent on any company or government to maintain its value.

No setup Merchants can start accepting bitcoins instantaneously, without setting up merchant accounts, buying credit card processing hardware, etc.

Better privacy Bitcoins are less traceable than many types of monetary transactions (though not anonymous, as we discuss in this thesis).

No counterfeit, no chargebacks Bitcoins can not be counterfeited and transactions can not be reversed.

No account freezing No transaction blocking or account freezing. Governments can freeze bank accounts of dissidents and payment processors refuse to process certain types of transactions (for example, PayPal froze WikiLeaks' account [10]).

Algorithmically known inflation Bitcoin is seen as a store of value because the total number of bitcoins that will ever be created is known in advance and it is impossible to create more than that.

Bitcoin makes use of several cryptographic technologies.

First is public key cryptography. Technically, bitcoin addresses are nothing more than hashed public keys. Each coin is associated with its current owner's public ECDSA (Elliptic Curve Digital Signature Algorithm, based on calculations of elliptical curves over finite space) key.

When a user sends bitcoins to someone, a message (*transaction*) is created, attaching the new owner's public key to this amount of coins, and signing it with the payer's private key. When the transaction is broadcast to the bitcoin network, the signature on the message verifies for everyone that the transaction is authentic. The complete history of transactions is kept by everyone, so anyone can verify who is the current owner of any particular group of coins.

This complete record of transactions is kept in the *block chain*, which is a sequence of records called *blocks*. All computers in the peer-to-peer network have a copy of the block chain, which they keep updated by passing along new blocks to each other. Each block contains a group of transactions that have been sent since the previous

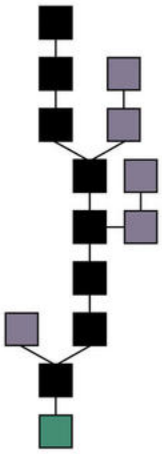


FIGURE 2.1: A tree representation of the Bitcoin blockchain

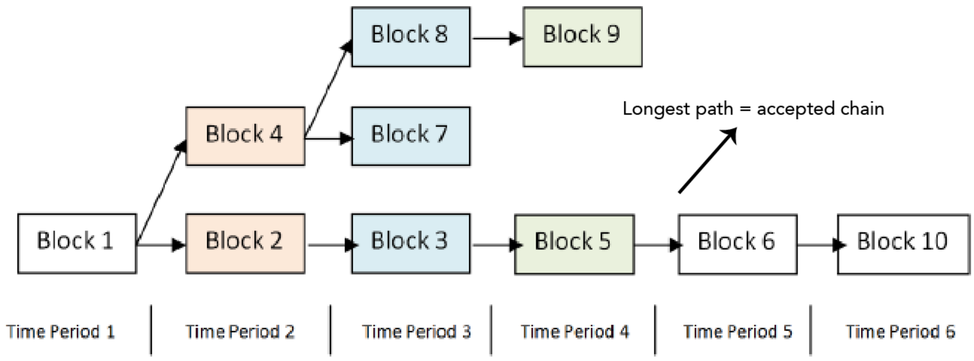


FIGURE 2.2: Visual representation of how Bitcoin deals with the distributed consensus problem

block. In order to preserve the integrity of the block chain, each block in the chain confirms the integrity of the previous one, all the way back to the first one, the genesis block.

For any block on the chain, there is only one path to the genesis block. Coming from the genesis block, however, there can be forks. One-block forks are created from time to time when two blocks are created just a few seconds apart. When that happens, generating nodes build onto whichever one of the blocks they received first. When miners generate blocks at the same time or too close together, the protocol has to deal with a *distributed consensus problem*. A participant of the network choosing to extend an existing path in the block chain indicates a vote towards consensus on that path (see Figure 2.2). The longer the path, the more computation was expended building it. This is interesting, because Bitcoin offers a unique solution to the consensus problem in distributed systems since voting power is directly proportional to computing power.

The generation of a new block is costly because each block must meet certain requirements that make it difficult to generate a valid block. To make generating bitcoins difficult a *cost-function* is used. Integrity, block-chaining, and the cost-function rely on SHA (SHA-256 and SHA-1) as the underlying cryptographic hash function, and blocks can be identified by their hash, which serves the dual purpose of identification as well as integrity verification. The cost-function difficulty factor is achieved by requiring that the hash output has a number of leading zeros, an easily-verifiable proof of work – every node on the network can instantly verify that a block meets the required criteria. This framework guarantees the essential features of the Bitcoin system: verifiable ownership of bitcoins and a distributed database of all transactions, which prevents *double spending*.

Users enter the Bitcoin system by trading non-digital currencies for the Bitcoin currency at a number of currency exchange marketplaces, or by *mining* coins, where miners computationally compete to solve a cryptographically hard problem; the solver is then rewarded a fixed number of bitcoins for cryptographically validating transactions on the network. Mining can be seen as the process of securing transactions and committing them into the bitcoin public chain. The Bitcoin protocol allows the miner who generates a block to claim a fixed number of bitcoin (decreasing with time) as well as any transaction fees for the transactions that the miner chooses to include in the block.

This *mining* is somehow analogous to physical mining of gold. Rather than being backed by gold or other precious metals, the value of bitcoins is derived from computation expended.

Nowadays mining with GPUs is no longer profitable, and ASICs are used. Companies such as Butterfly Labs[1] manufacture high speed processors and boards dedicated to bitcoin mining.

Finally, Bitcoin protocol has been designed to be forward compatible. While ECDSA is not secure under quantum computing, quantum computers of a kind that could be used for cryptography do not exist yet. Bitcoin's security, when used properly with a new address on each transaction, depends on more than just ECDSA: cryptographic hashes such as SHA-256 are much stronger than ECDSA under a quantum computing paradigm. Bitcoin's security was designed to be upgraded in a forward compatible way and provides a scripting language layer that makes switching to other algorithms transparent to users, if this were considered an imminent threat.

2.1.2 Bitcoin in economic theories

Will Bitcoin ever become mainstream money?

While this is an important question, it is also not relevant for the broader question of the future of Bitcoin.

The *money or nothing fallacy* argues that if we can refute that Bitcoin is or will ever be *money*, it follows that it is *nothing*.

As long as it provides a significant advantage in transaction costs, Bitcoin will be competitive – even if it will never replace fiat currencies, whether that is due to too much leverage the states have over fiat money, or due to inertia. Non-Austrians economists have called such a medium of exchange *metacurrency*, for example, Krugman calls it *vehicle currency* [16]. Austrians also have a name suitable for such class of media of exchange, for example *secondary media of exchange* [32] (Mises) or *quasi-monies* [21] (Rothbard).

From an economic point of view, Bitcoin has the following features:

- **immaterial good**
- with **ultra low transaction costs**, and

[1] `http://www.butterflylabs.com`

- **inelastic supply** (regardless of price or demand, a fixed amount of bitcoins are generated every ten minutes).

Since Bitcoin is not a *claim* (nobody is obligated to redeem it), nor is it treated at par with anything else, it should not be considered a *money substitute*, but it is closer to *commodity money*. Another reason for classifying Bitcoin as commodity money is the inelasticity of supply.

2.1.3 Regulation in the United States and Germany

On March 18th, 2013, the United States Financial Crimes Enforcement Network issued a clarification [8] to the US regulation regarding virtual currencies.

The statement, without explicitly addressing Bitcoin, stipulates that digital currencies are to be treated essentially as foreign currencies, and not as tender money. This clarifies that Bitcoin will not be treated as illegal tender in the US, because, according to FinCEN, it lacks all the real attributes of real currency. Companies that exchange bitcoins for real money will have to comply with the same money laundering regulations as traditional currency exchangers – namely, they must verify the identity of anyone exchanging money for bitcoins and report large transactions to the government.

Using a digital currency to purchase goods, however, is specifically exempted.

In August of 2013, Bitcoin has been recognized by the German Finance Ministry as a *unit of account*, meaning it is can be used for tax and trading purposes in the country. It is thus not classified as e-money or a foreign currency, but is rather a financial instrument under German banking rules [6].

On November 18, 2013, the United States Senate Committee on Homeland Security and Governmental Affairs heard testimony from various government officials, academics, and Bitcoin proponents to discuss virtual currencies [27]. As the committee was meeting, the current exchange rate of bitcoins to dollars was skyrocketing, breaking 900 USD per bitcoin (in February of the same year, one bitcoin was trading for 30 USD).

In their written testimonies released prior to the hearing, various government officials detailed their attitude and policies toward Bitcoin in particular. They noted that while such virtual currencies may be *legitimate*, they pose potential issues for law enforcement. Peter Kadzik, the Principal Deputy Assistant Attorney General, wrote in his letter to the committee that the FBI has "founded and chairs the Virtual Currency Emerging Threats Working Group".

Notably, outgoing Federal Reserve Chairman Ben Bernanke wrote "[T]here are also areas in which [such currencies] may hold long-term promise, particularly if the innovations promote a faster, more secure, and more efficient payment system" [29].

Bernanke said that while the Federal Reserve does monitor the evolution of Bitcoin and other related currencies, "it does not have authority to directly supervise or regulate these innovations or the entities that provide them to the market".

2.1.4 Privacy in Bitcoin

In 2012, *Androulaki, Elli et alii* [2] explored the privacy implications of Bitcoin. Their findings show that the current measures adopted by Bitcoin are not enough to protect the privacy of users if Bitcoin were to be used as a digital currency in realistic settings. More specifically, in a small controlled environment, clustering techniques were found suitable to unveil the profiles of Bitcoin users, even if these users try to enhance their privacy by manually creating new addresses.

The main problem, when it comes to anonymity, is that the history of a coin is publicly available. Anyone can see the flow of bitcoins from address to address. Common anonymization techniques are:

- Randomly sending coins to new addresses generated just for this purpose. The coins are still part of the owner's balance, but it is very difficult for an outsider to prove that the owner sent the coins to himself instead of another person. However, the transaction chain still has the owner's identity in it.

- Using a *mixer*, that takes the coins of many different people, mix them up, and send similar amounts back to those peoples' addresses. If the mixer keeps no logs of who gets which coins, investigations are unfeasible.

To be truly anonymous, Bitcoin should have, by default, the capability to automatically send coins through several external mixers.

Bitcoin is also vulnerable to network analysis attacks: if an attacker is able to watch all of the victim's incoming and outgoing traffic, (s)he can easily see which transactions are initiated by the victim. Since the connection is not encrypted, transactions broadcast (and not received) by the victim are the ones originated by the victim.

Records of every single Bitcoin transaction that is ever been conducted are public. From a privacy perspective, this is a weakness. Due to the way Bitcoin works, this information can not be limited to just a few trustworthy parties, since there are *no* trustworthy parties. This means all of your transactions are conducted in public, and each transaction is tied to the one that precedes it. In a sense this makes Bitcoin much less private than *cash*, and even worse than *credit cards*. If an user chooses to engage in sensitive transactions on Bitcoin, (s)he should be aware that a public record will be preserved forever.

Since every user can generate as many addresses as (s)he wants, Bitcoin offers privacy through *pseudonymity* only, and in this thesis we show how it is possible to de-anonymize Bitcoin transactions.

Recently, in 2013, Ian Miers, Christina Garman, Matthew Green and Aviel D. Rubin proposed a cryptographic extension to Bitcoin that augments the protocol to allow for fully anonymous transactions called Zerocoin [15]. To achieve the goal, Zerocoin adds extensions to the existing Bitcoin protocol, such as digital commitments (that allow one to commit to a chosen statement while keeping it hidden to others, with the ability to reveal the committed statement later), one-way accumulators (a decentralized alternative to digital signatures) and zero-knowledge proofs (a method by which one party, the *prover*, can prove to another party, the *verifier*, that a given statement is true, without conveying any additional information).

2.2 Reception, usage and abuses

2.2.1 Spending Bitcoin

Bitcoin, especially in recent months, is being used by individuals to trade goods.

For example, Bitmit[2] is an *eBay*-like shopping platform on which people from all over the world can trade their goods using Bitcoin. It is also simple to buy gift cards for Amazon or other big e-commerce companies.

We would like to report two interesting examples of trading between individuals: the first takes place in May 2010, the second in April 2013.

In May of 2010, a BitcoinTalk user called *laszlo* from Jacksonville, Florida, bought two pizzas for 10,000 BTC[3]. Another user, *jercos*, bought two pizzas to be delivered to him and posted photos as proof[4][5].

10,000 BTC were valued $41 at the time of the trade. In November 2013, they can be sold for 8 million USD.

We will analyze this trade with BitIodine in Section 5.4.

On April 2, 2013 a family in Austin sold a 2007 Porsche Cayman S for 300 BTC[6]. The buyer reportedly purchased his initial Bitcoin investment years ago, at around $4 a piece, thus actually paying the car $1200 only (300 BTC were valued around 42,000 USD at the time of the trade).

A number of online businesses and non-profit organizations accept Bitcoins, most notably Wordpress [34], 4chan [1], Wikileaks [33], Reddit [26], domain registrar and hosting provider Namecheap [23] and one of the biggest online dating communities, OkCupid (). Additionally, the Internet Archive has offered their employees an option to receive a portion of their paychecks in Bitcoins [3].

BitElectronics[7] is an e-commerce website that sells consumer electronics to all EU countries in Bitcoin, with free shipping.

[2] `http://bitmit.net`

[3] `https://bitcointalk.org/index.php?topic=137.0`

[4] `http://heliacal.net/~solar/bitcoin/pizza/`

[5] `http://blockchain.info/tx/a1075db55d416d3ca199f55b6084e2115b9345e16c5cf302fc80e9d5fbf5d48d`

[6] `https://bitcointalk.org/index.php?topic=143722.0`

[7] `http://bitelectronics.net`

Figure 2.3: Bitcoin is getting mainstream business acceptance.

Howard Johnson, a chain of hotels and restaurants located primarily throughout the United States and Canada, accepts Bitcoin.

Online food delivery and takeout portal Foodler is accepting Bitcoin alongside credit cards and cash-on-delivery for orders from more than 17,000 restaurants in the US.

There are also a handful of Bitcoin casinos and gambling sites. The transparent nature of Bitcoin, where every transaction is public, revolutionized the online gambling world, since owners can provide cryptographically provable fairness and publicly display proof of payment of winnings. Finally, a number of truly zero-sum games where players compete against each other, and not against the gambling site, recently appeared.

2.2.2 Trading and exchanges

On July 18th, 2010, the Japan-based exchange market *Mt. Gox* launched, allowing people to buy and sell Bitcoins in exchange for real currencies such as US dollars or Euro, as well as providing a simple way for merchants to accept Bitcoins as payment on their websites. In July 2011, they facilitated more than 80% of all Bitcoin trading, but nowadays other exchanges such as *Bitstamp* and *BTC-e* are gaining popularity. Recently, Mt. Gox joined forces with Coinlab - the world's first US-venture backed Bitcoin company - to cater to their customer base in the US and Canada.

There are several other minor exchanges, mostly based in Europe and the US. Bitcoins can also be traded locally from person to person, using services such as LocalBitcoins[8].

At the time of writing (November 2013), Bitcoin is in high demand, one Bitcoin trades for approximately 800 USD (more than 20 times the price in January of the same year), and its value increased sixfold in a month. The network has a market capitalization of more than 9 billion USD.

2.2.3 The Silk Road

Bitcoins are the only currency accepted on Silk Road, a now defunct online black market in the *deep web* that could only be accessed via TOR. Even though the site launched in February 2011, the site did not receive mainstream attention until Gawker published an expose on it in June of that year. Silk Road allowed people to buy a number of items including drugs, apparel, books, digital goods, drug paraphernalia, erotica and forgeries [5].

On October 1st 2013, Ross William Ulbricht, a 29-year-old man, was arrested in a joint operation run by the cybercrime squad within the FBI's New York field office involving the FBI, DEA, IRS and Homeland Security's investigative unit. According to the allegations [25], he is the creator and operator of the infamous "Silk Road" black market, under the alias of "Dread Pirate Roberts" (DPR). From February 6, 2011 to July 23 2013, sales through the market amounted to 9,519,664 BTC (spread across 1,229,465 transactions), 614,305 BTC of which went directly to the accused as commissions. Prosecutors said they seized approximately 173,600 BTC, at date around USD 138,000,000, in the largest seizure of the digital currency ever [25].

[8] `https://localbitcoins.com`

2.2.4 Bitcoin hedge fund

In 2013, Exante Ltd., a Malta-based investment firm, opened to the public a bitcoin hedge fund marketed towards institutional investors and high net-worth individuals. Bitcoin shares are currently traded through the Exante Hedge Fund Marketplace platform and authorized and regulated by the Malta Financial Services Authority. As of March 2013, Exante holds 3.2 million USD in bitcoin assets.

2.2.5 Bitcoin communities

Bitcoin users and miners congregate on Reddit and the *Bitcoin Talk* forums, among numerous other smaller local and regional groups. There is also a Wiki[9] and the online publication Bitcoin Magazine[10] that gathers information about the currency. Since 2011, Bitcoin conferences have been held annually throughout Europe (2011, Prague and 2012, London), with the first US conference scheduled for May 2013 in San Jose, California.

2.2.6 Bitcoin in cybercrime

In July 19, 2011, the first known Bitcoin miner trojan was found in the wild by Symantec, and named *Trojan.Coinbitminer*[11]. A month after, another Bitcoin mining bot, controlled via Twitter, was found by F-Secure [14].

In April 2013, according to antivirus seller Kaspersky Lab, a new trojan that takes control of infected machines and forces them to mine Bitcoin is spreading via Skype. With the trojan, cybercriminals aim at creating a botnet of infected machines, called *zombies*, forced to perform complex calculations to earn them money, putting the machines under heavy CPU and GPU load.

In September 2013, a malware called CryptoLocker was discovered in the wild. CryptoLocker is a malicious program belonging to the category known as *ransomware*.

[9] `https://en.bitcoin.it/wiki/Main_Page`

[10] `http://bitcoinmagazine.com/`

[11] `https://www.symantec.com/security_response/writeup.jsp?docid=2011-072002-1302-99`

It encrypts the victim's personal files with strong encryption, and the criminals retain the only copy of the decryption key on their server. The malware asks for a ransom to be paid with MoneyPak or Bitcoin within 72 hours in order to release the files. We use BitIodine to detect CryptoLocker clusters, belonging to the malware authors, and compute some statistics about ransoms paid by the victims.

2.3 State of the art, goals and challenges

The need for a digital currency based in cryptography was discussed in two separate academic papers published in 1993 by researchers at Carnegie Mellon University [31] and the University of Southern California [17]. Five years later, cryptography advocate Wei Dai suggested a system in which the currency would be both regulated and created through crowdsourced cryptography, thus eliminating the risk of double-spending altogether [7].

On November 1st, 2008, a person or group of people under the pseudonym Satoshi Nakamoto distributed a paper [22] solidifying this idea into a proposal for peer-to-peer electronic cash system called *Bitcoin*. Beyond Bitcoin, no other links to this identity have been found, and his involvement in the original bitcoin protocol does not appear to extend past mid-2010. The blockchain was started on or after January 3rd, 2009, as its first block, called *genesis block*, references the title of an article published that day in the UK newspaper *The Time*s about a bank bailout [9]. The announcement of the system and its open source client was posted on the Cryptography Mailing List[12] on January 9th.

Bitcoin transactions do not explicitly identify the payer, nor the payee, as transactions are just cryptographically signed messages embodying a transfer of funds from one public key to another. The corresponding private keys are needed to authorize a fund transfer.

The decentralized paradigm of Bitcoin requires each node of the network to keep in memory the entire transaction history, called the *blockchain*. Bitcoin keys are not

[12] http://www.mail-archive.com/cryptography@metzdowd.com/msg10142.html

explicitly tied to real-world individuals or organizations. All transactions are public, transparent, and permanently recorded since the origin. This means that anyone can see the flow of Bitcoin from address to address.

There is a lot of potentially interesting information to be mined out of the blockchain. Some addresses are known and tied to entities, such as for instance gambling sites, users of the main Bitcoin-related forum, Bitcoin Talk, or Bitcoin-OTC marketplace. By analyzing the blockchain, it is possible to automatically find out how much an address is used for gambling activities or mining, if it was used for scamming users in the past, if and how it is related to other addresses and entities. The idea of algorithmically associating Bitcoin addresses to entities controlling them is described by *Androulaki et al.* [2] and *Ivan Brugere* [4]. They both share the idea of grouping addresses into entities that control them. The first work investigates Bitcoin privacy provisions in a simulated setting where Bitcoin is used for daily payments, and comes to the conclusion that the current implementation of Bitcoin would enable the recovery of user transaction profiles to a large extent. The second work is an investigation of the Bitcoin digital currency network from a data mining perspective focused on anomaly detection, using simple network features, in an attempt to monitor the network to identify thefts.

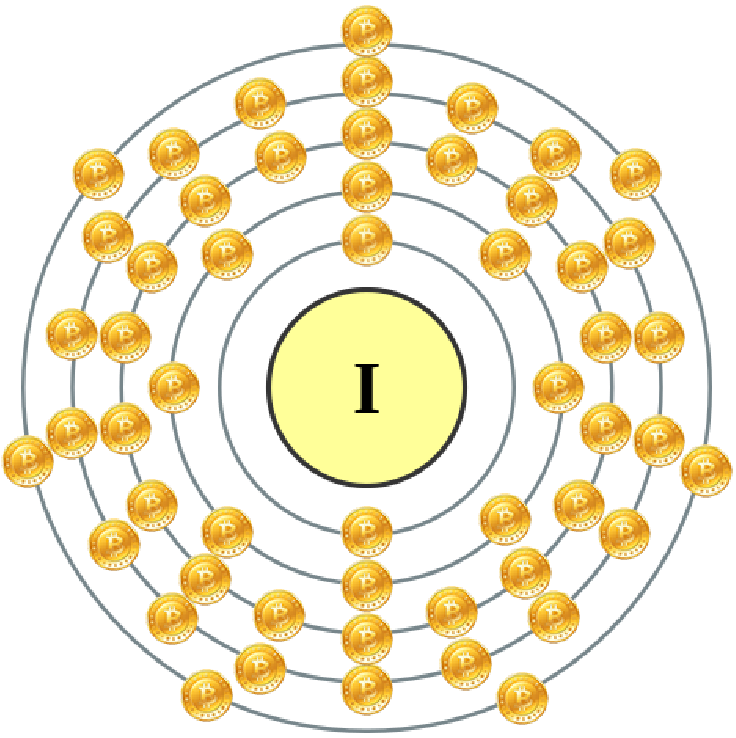
Reid and Harrigan also carried out an important analysis of anonymity in Bitcoin [28], advocating the creation of appropriate tools to associate many public-keys with each other, and with external identifying information. The activity of known users can be observed in detail using passive analysis only, but the authors take into consideration also active analysis, where an interested party can potentially deploy *marked* Bitcoins and collaborate with other users to discover even more information.

Möser focused on analyzing mixing services [20], such as Bitcoin Fog, that claim to obfuscate the origin of transactions, thus increasing the anonymity of its users. *Ober, Katzenbeisser, Hamacher* analyzed the topology and dynamics of the Bitcoin transaction graph, detecting structural patterns that have implications for the anonymity of users in [24]. *Christin* collected precious information about the Silk Road [5] before the seizure by the FBI.

A forensic approach to the problem is present in *Meiklejohn et al.* [18], with a stress on investigating the use of Bitcoin for criminal or fraudulent purposes at scale. Using a small number of manually labeled transactions, the authors were able to identify major institutions and the interactions between them and demonstrated that this approach can shed considerable light on the structure of the Bitcoin economy and how Bitcoin is used.

We believe that limited or no supervision for labeling addresses and users will become a necessity as the Bitcoin network grows. A more automated and scalable approach to Bitcoin forensics is needed, and we aim to develop it in this work.

BitIodine: system overview 3



3.1 Bitcoin terminology

3.1.1 Address

A Bitcoin *address* is a string like `1MikiSPbrhCFk7S4wzZP7gQqhwWH866DCb` generated by a Bitcoin client together with the private key needed to redeem the coins sent to it. It is public, and can be posted everywhere in order to receive payments.

3.1.2 Wallet

A *wallet* is a file which stores addresses and the private keys needed to use them.

3.1.3 Blockchain

The *blockchain* is a shared public transaction log on which the entire Bitcoin network relies. All confirmed transactions are included in the blockchain with no exception. This way, new transactions can be verified to be spending bitcoins that are actually owned by the spender. The integrity and the chronological order of the blockchain are enforced with cryptography[1].

3.1.4 Block

A *block* is an individual unit of a blockchain. In order to guarantee integrity, each block contains the hash of the previous block and as many unconfirmed (not embedded in previous blocks) transactions as can be found in the network.

3.1.5 Transaction

A *transaction* is a transfer of value between Bitcoin addresses that gets included in the *blockchain* and broadcast by the network. Transactions are *signed* by private keys of owners of the input addresses, providing a mathematical proof that they come from the owner of the addresses. The signature also prevents the transaction from being altered by anybody once it has been issued.

[1] `http://bitcoin.org/en/how-it-works`

3.2 BitIodine terminology

3.2.1 Transaction and User graphs

In order to analyze several transactions between seemingly meaningless addresses, it is important to build a graph of them.

In *transaction graphs*, *nodes* are *addresses* and *edges* are single *transactions* between them.

For instance, in Figure 3.1 there is a visualization of a transaction graph related to a Bitcoin *faucet* (a website that gives away a very little amount of bitcoins to any visitor that inputs an address). The address of the faucet is `15Art…`. Other important addresses in the graph are `1sLiM…`, which is owned by Bitcoin Talk user *IXIslimIXI*, and `14wQQ…`, which, as explained later when presenting the tool, is classified by BitIodine as an empty, old address, used for gambling.

BitIodine de-anonymizes addresses by grouping them into *users* or *clusters*. We will hereinafter refer to such clusters and entities interchangeably as *users*.

This is to create an *user graph*, compacting the huge transaction graph and giving a new meaning to transactions: edges connecting users are aggregate transactions – transfer of money between precise entities.

In *user graphs*, *nodes* are *users* and *edges* are *macro-transactions* (i.e., flows of coins) between them.

The interesting outcome for investigators is that it is possible to retrieve valuable information about an entity by just knowing one of its addresses. Collapsing addresses into clusters compacts and simplifies the huge transaction graph, creating edges between users that correspond to aggregate transactions. In other words, with this approach it is possible to move out of the way the complexity of apparently anonymous transactions between meaningless addresses, and make money exchanges between entities visible.

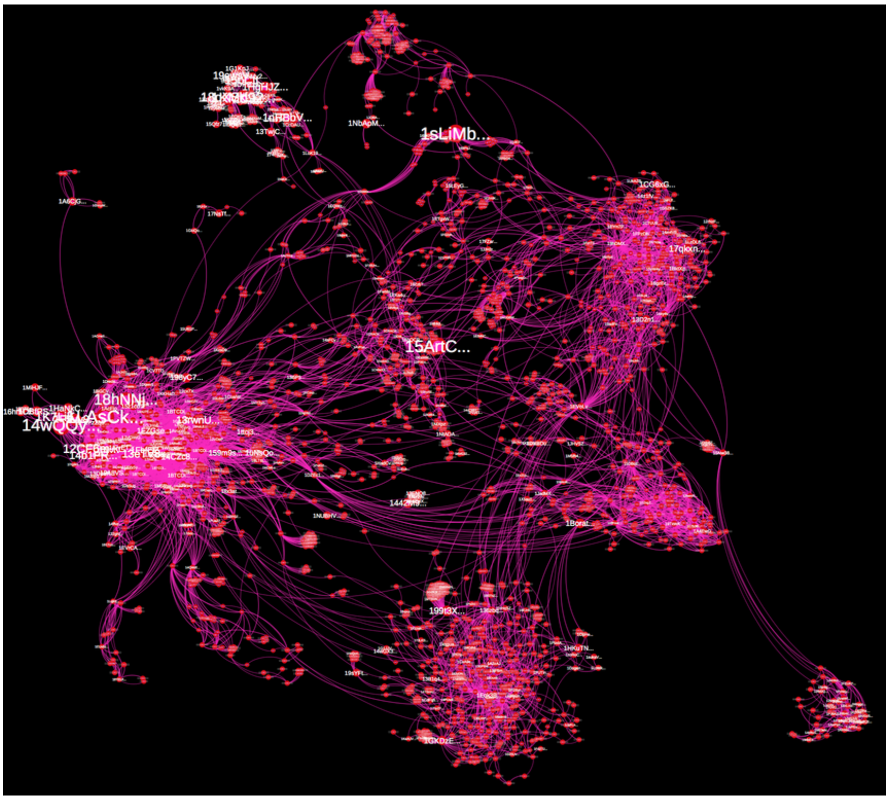


Figure 3.1: A transaction graph – transactions related to a Bitcoin faucet

3.2.2 Heuristics

Two heuristics are used to group addresses into *users*.

3.2.2.1 First heuristic

The first heuristic exploits multi-input transactions. Multi-input transactions occur when a user wishes to perform a payment, and the payment amount exceeds the value of each of the available Bitcoin in the user's wallet. In order to avoid performing multiple transactions to complete the payment, enduring losses in terms of transaction fees, Bitcoin clients choose a set of Bitcoin from the user's wallet such that their aggregate value matches the payment and perform the payment through multi-input transactions. This means that whenever a transaction has multiple input addresses, we can safely assume that those addresses belong to the same wallet, thus to the same user.

3.2.2.2 Second heuristic

The second heuristic has to do with *change* in transactions. The Bitcoin protocol forces each transaction to spend, as output, the whole input. This means that the "unspent" output of a transaction must be used as input for a new transaction, which will deliver "change" back to the user. In order to improve anonymity, a *shadow address* is automatically created and used to collect the change that results from any transaction issued by the user. The heuristic tries to predict which one of the output addresses is actually belonging to the same user who initiated the transaction, and it does so in two possible ways: the first one is completely deterministic, the second one exploits a (recently fixed) flaw in the official Bitcoin client.

In Figure 3.2, big red nodes are *users*, big blue nodes are addresses of SatoshiDice, a popular Bitcoin gambling site, and the rest are normal, ungrouped addresses.

By just looking at the graph, we can easily spot big players, and the *satellite* structure generated by gamblers moving coins to other addresses not known to be owned by the same person.

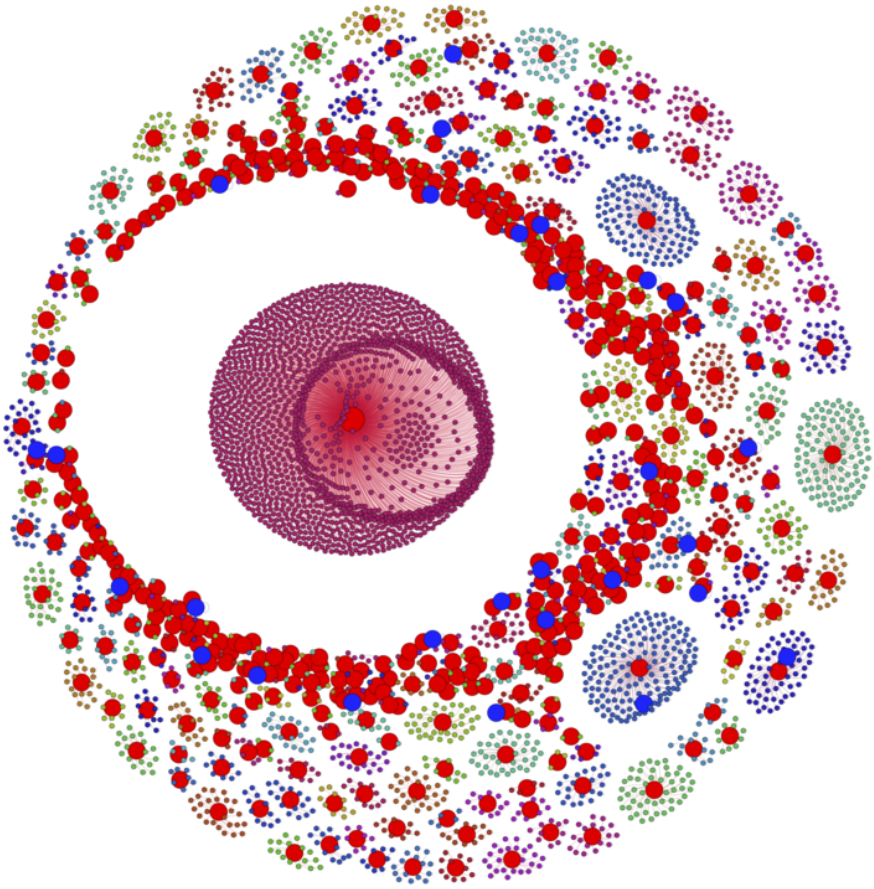


FIGURE 3.2: A graph with addresses grouped in users – SatoshiDice gamblers

3.3 Architecture and data flow overview

BitIodine is meant to be a modular, expandable and easily deployable framework to build complex applications for forensic analysis of the Bitcoin blockchain.

Figure 3.3 describes in a simplified way the building blocks of BitIodine and the interactions between different modules.

3.3.1 Block Parser

The block parser reads blocks and transactions from the local `.bitcoin` folder populated by the official *bitcoind* client and exports the blockchain data to the *blockchain DB*, which uses a custom relational schema we designed in order to obtain good per-

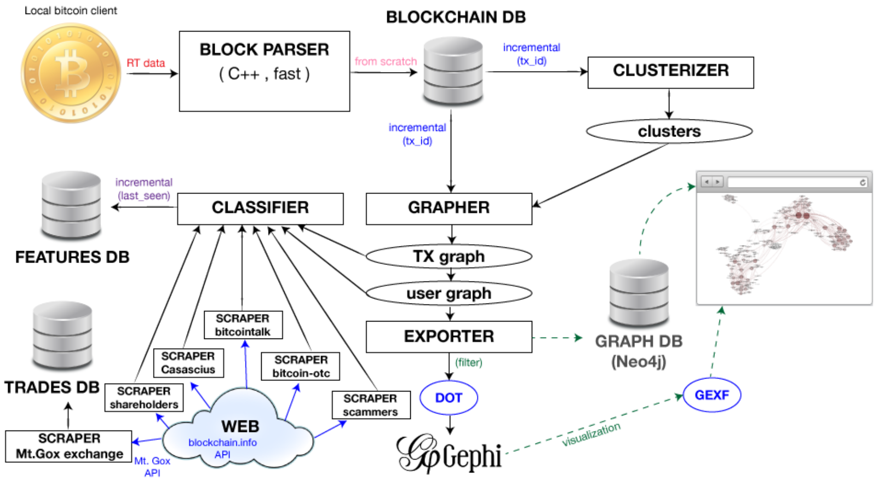


Figure 3.3: Building blocks of BitIodine

formance (see Section 4.2) when queried by other modules. This allows for a fast updating of data from the Bitcoin network.

3.3.2 Clusterizer

The goal of the clusterizer is to find groups of addresses that belong to the same user. It incrementally reads the blockchain DB and generates/updates clusters of addresses using two heuristics, detailed in Section 4.1. The first heuristic exploits transactions with multiple inputs, while the second leverages the concept of "change" in transactions (see Section 4.1). These clusters are stored in *cluster files*.

3.3.3 Scrapers

A set of scrapers crawl the web for Bitcoin addresses to be associated to real users, automatically collecting, generating and updating lists of:

- *usernames* on platforms, namely *Bitcoin Talk* forum and *Bitcoin-OTC* marketplace (from forum signatures and databases)
- *physical coins* created by Casascius[2] along with their Bitcoin value and status

[2] `https://www.casascius.com`

(opened, untouched)

- known *scammers*, by automatically identifying users that have significant negative feedback on the Bitcoin-OTC and Bitcoin Talk trust system.

- *shareholders* in stock exchanges (currently limited to *BitFunder*)

Additional lists can be built with a semi-automatic approach which requires user intervention. In particular, by downloading tagged data via Blockchain.info[3], the tool helps user build lists of *gambling* addresses, *online wallet* addresses, *mining pool*[4] addresses and addresses which were subject to *seizure* by law enforcement authorities. The user can verify tags and decide to put the most relevant ones in the correct lists.

Finally, a scraper uses Mt. Gox trading APIs to get historical data about trades of Bitcoin for US dollars, and saves them in a database called *trades DB*. This module is useful to detect interesting flows of coins that enter and exit the Bitcoin economy.

The interface is easily expandable, and adding scrapers for new services and websites is easy.

3.3.4 Grapher

The grapher incrementally reads the *blockchain DB* and the *cluster file* to generate, respectively, a *transaction graph* and a *user graph*. In a transaction graph, addresses are nodes and single transactions are edges. It is useful for several applications, such as finding successors and predecessors of an address. In a user graph, users (i.e., clusters) are nodes and aggregate transactions between them are edges.

3.3.5 Classifier

The classifier reads the *transaction graph* and the *user graph* generated by the *grapher*, and proceeds to automatically label both single addresses and clusters with specific

[3] `https://blockchain.info/tags`

[4] Pooled mining is a Bitcoin generation (mining) approach where multiple generating clients contribute to the generation of a block, and then split the block reward according the contributed processing power.

annotations. Examples of labels are Bitcoin Talk and Bitcoin-OTC usernames, the ratio of transactions coming from direct or pooled mining, to/from gambling sites, exchanges, web wallets, other known BitcoinTalk or Bitcoin-OTC users, freebies and donation addresses. There are also boolean flags, such as *one-time address*, *disposable*, *old*, *new*, *empty*, *disposable*, *scammer*, *miner*, *shareholder*, *FBI*, *Silk Road*, *killer* and *malware*. The complete list of labels for addresses is presented in Figure 1, and the one for clusters is in Figure 2, in Chapter 7.

Classification can take place globally on the whole blockchain, or selectively on a list of specified addresses and clusters of interest. The results are stored in a database and can be updated incrementally.

3.3.6 Exporters

A module allows to export and filter (portions of) the *transaction graph* and the *user graph* in several formats, and support manual analysis by finding *simple paths* (i.e., paths with no repeated nodes) on such graphs. More precisely, we allow the user to *export* transactions which happened *inside* a cluster, or *originating* from a cluster. We also allow the user to find either the shortest, or all of the *simple paths* from an address to another address, from an address to a cluster, from a cluster to an address, or between two clusters. Moreover, we allow users to find all simple paths originating from an address or a cluster (i.e., the subgraph of *successors*, as defined in graph theory), or to reverse such search, by identifying the subgraph of *predecessors* of an address or cluster. Subgraphs of successors or predecessors can be useful, for instance, in taint analysis, and can assist manual investigation of mixing services, as we do in Section 5.2.

System details 4

In order to formalize algorithms and heuristics used by BitIodine, we need to define a few concepts.

Let N denote the whole Bitcoin network. We denote with n_B, n_U, n_A, respectively, the total number of blocks, users and addresses in the network. We also denote as $B = \{b_1, b_2, \ldots, b_{n_B}\}$ the set of blocks in the network N, and similarly as $U = \{u_1, u_2, \ldots, u_{n_U}\}$ the set of users and as $A = \{a_1, a_2, \ldots, a_{n_A}\}$ the set of addresses.

We also denote with $\tau_i(S_i \to R_i)$ a transaction with a unique index i, and $S_i \subseteq A$ and $R_i \subseteq A$ denote the sets of senders' addresses and recipients' addresses, respectively. We define $T = \{\tau_1(S_1 \to R_1), \tau_2(S_2 \to R_2), \ldots, \tau_{n_T}(S_{n_T} \to R_{n_T})\}$ as the set of all n_T transactions which took place. We also define $T|_{b_i} \subset T$ as the subset of all the transactions contained in blocks with index $k \leq i$ (blocks are uniquely identified by indexes starting from 0, for the *genesis block*, sequentially increasing as they are appended to the blockchain).

We also define two functions. The first is $lastblock : T \mapsto B$ (a function that maps the set of transactions to the set of blocks): $lastblock(\tau_i) = b_i$ if and only if b_i is the last block relayed by the network N as the transaction τ_i is broadcast. The second is $owns : A \mapsto U$ (a function that maps the set of addresses to the set of users): $owns(a_i) = u_k$ if and only if u_k owns the private key of a_i.

Listing 4.1: Fragment of code in `src/wallet.cpp`

```
// Insert change txn at random position
int list_begin = wtxNew.vout.begin();
int n_of_payees = wtxNew.vout.size();
vector<CTxOut>::iterator position = list_begin + GetRandInt(n_of_payees);
wtxNew.vout.insert(position, CTxOut(nChange, scriptChange));
```

4.1 Formal definition of heuristics

4.1.1 First heuristic: multi-input transactions grouping

The first heuristic exploits multi-input transactions. Multi-input transactions occur when a user u wishes to perform a payment, and the payment amount exceeds the value of each of the available Bitcoin in u's wallet. In order to avoid performing multiple transactions to complete the payment, enduring losses in terms of transaction fees, Bitcoin clients choose a set of Bitcoin from u's wallet such that their aggregate value matches the payment and perform the payment through multi-input transactions. This means that whenever a transaction has multiple input addresses, we can safely assume that those addresses belong to the same wallet, thus to the same user.

More formally, let $\tau_i(S_i \rightarrow R_i) \in T$ be a transaction, and $S_i = \left\{a_1, a_2, \ldots, a_{n_{S_i}}\right\}$ the set of input addresses. Let also $|S_i| = n_{S_i}$ be the cardinality of the set. If $n_{S_i} > 1$, then all input addresses belong to the same (previously known or unknown) user: $owns(a_i) \triangleq u_k \quad \forall i \in S_i$.

4.1.2 Second heuristic: shadow address guessing

The second heuristic has to do with *change* in transactions. The Bitcoin protocol forces each transaction to spend, as output, the whole input. This means that the "unspent" output of a transaction must be used as input for a new transaction, which will deliver "change" back to the user. In order to improve anonymity, a *shadow address* is automatically created and used to collect the change that results from any transaction issued by the user. The heuristic tries to predict which one of the output addresses is actually belonging to the same user who initiated the transaction, and it does so in two possible ways: the first one is completely deterministic, the second one exploits a (recently fixed) flaw in the official Bitcoin client.

The completely deterministic and conservative variant works as follows: If there are two output addresses (one payee and one change address, which is true for the vast majority of transactions), and one of the two has never appeared before in the blockchain, while the other has, then we can safely assume that the one that never appeared before is the shadow address generated by the client to collect change back.

More formally, let $\tau_i(S_i \rightarrow R_i) \in T$ be a transaction, and $R_i = \left\{a_1, a_2, \ldots, a_{n_{R_i}}\right\}$ be the set of output addresses (with $|R_i| = n_{R_i}$ being the cardinality of the set), and let us consider $T|_{lastblock(\tau_i)}$, that is, the set T limited to the last block at the time of transaction τ_i. If $n_{R_i} = 2$, then the output addresses are a_1 and a_2. If $a_1 \notin T|_{lastblock(\tau_i)}$ and $a_2 \in T|_{lastblock(\tau_i)}$, then a_1 is the shadow address, and belongs to the same user u_k who owns the input address(es): $owns(a_1) \triangleq u_k$.

A bug in the official Bitcoin client allows to improve upon this heuristic. In the previous page, we can see a fragment of file `src/wallet.cpp`[1]. When the client chooses in which slot to put the shadow address, it passes to `GetRandInt` the number of payees. However, thanks to an off-by-one error, in the common case of one payee, `GetRandInt` will always return 0, and the change always ends up in the first output. It is an off-by-one error, and the code in line 3 should be corrected as follows:

Listing 4.2: Fix to correct an off-by-one error in `wallet.cpp`

```
int n_of_payees = wtxNew.vout.size() + 1;
```

This bug has been fixed only on 30 January 2013 (commit `ac7b8ea`), and the first version without this bug is 0.8.0 RC1, released on 9 February 2013. For transactions occurred before that date, just 6.8% have the shadow address provably in the second slot of two-outputs transactions. Due to this, for transactions before this date we can relax the heuristic, and consider a first output that was previously unseen in any

[1] `https://github.com/Bitcoin/Bitcoin/blob/master/src/wallet.cpp`

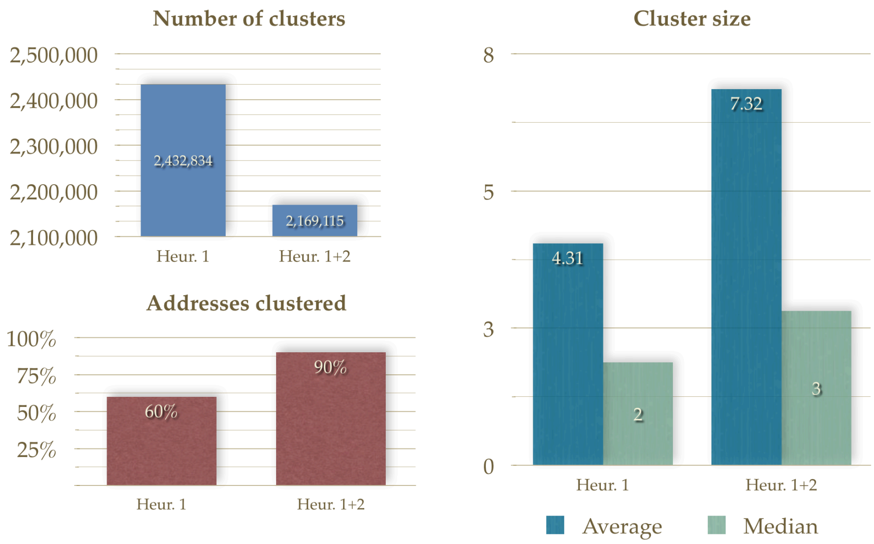


Figure 4.1: Statistics about clusters obtained with different heuristics

two-output transaction as a shadow address, regardless of the second one. This allows for a much better coverage, and generates much more compact clusters of users, as shown in Figure 4.1.

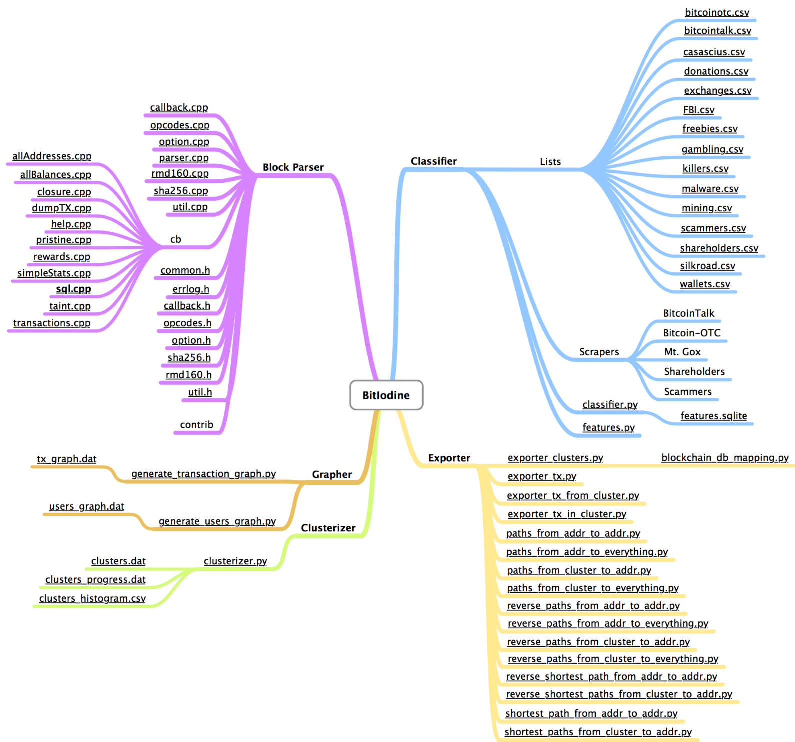


FIGURE 4.2: Code structure of BitIodine

4.2 Implementation details

We hereby describe BitIodine's implementation, and the challenges we faced in order to achieve good performance. Since we are dealing with several gigabytes of data, and graphs with millions of nodes and tens of millions of edges, there are significant scalability and performance issues to overcome. FigureFigure 4.2 shows a high-level overview of the code structure of BitIodine. Libraries and utility modules for Python modules are not included here for simplicity.

We used Python 3.3.3rc1 for every module, except the *Block parser*, which is written in C++ for performance reasons. The block parser is a modified version of the *blockparser*[2] tool by *znort987*, to which we added several custom callbacks: our modi-

[2] `http://github.com/znort987/blockparser`

fied version is highly efficient in exporting all addresses on the network, in performing taint analysis on an address, and in exporting to SQLite.

We opted for the use of embedded SQLite databases for storing the blockchain and the features database because it is a zero-configuration, server-less, embedded, stable and compact cross-platform solution. We do not need concurrency while writing to database files, so the only possible disadvantage does not affect its use in BitIodine. In designing the custom database schema for BitIodine we had to find a good balance between size and performance, weighing the use of indexes (see Section 5.6).

The clusterizer is designed to be incremental, and it is also possible to pause the generation of clusters at any time, and resume it from where it stopped.

Internally, graphs are handled by NetworkX, a Python language software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks[3], for the internal representation of users and transactions graphs.

NetworkX objects can be serialized and written to a file with ease, and in-memory querying for successors and predecessors of nodes is efficient. Is it also possible to embed an arbitrary number of additional data labels to nodes and edges (for instance, we added transaction hashes).

Exporters are implemented to support a multiplicity of output formats, allowing results to be fed into visualization software such as Gephi or exported to a graph database such as Neo4j.

4.2.1 Database schemas

In Figure 4.3 is the relational schema we designed to contain the blockchain. As mentioned earlier, the blockchain DB is populated by the *block parser* module.

The full SQL schema can be found in appendix.

In Figure 4.4 is the relational schema we designed to store features of classified addresses and clusters. This database is populated by the *classifier* module.

The full SQL schema can be found in appendix.

[3] `http://networkx.github.io`

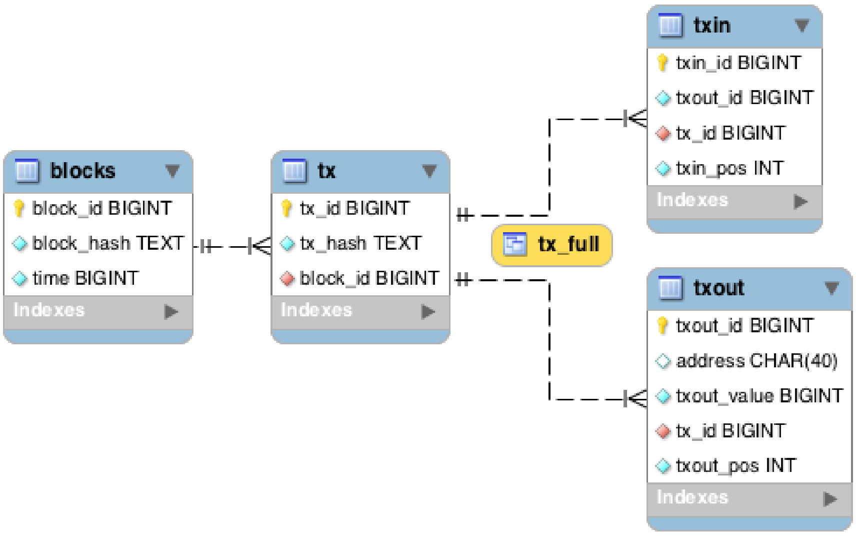


FIGURE 4.3: SQL schema diagram for the blockchain representation

In Figure 4.5 is the relational schema we designed to store trades in the Mt. Gox exchange. This database is populated by the *Mt. Gox scraper* module.

The full SQL schema can be found in appendix.

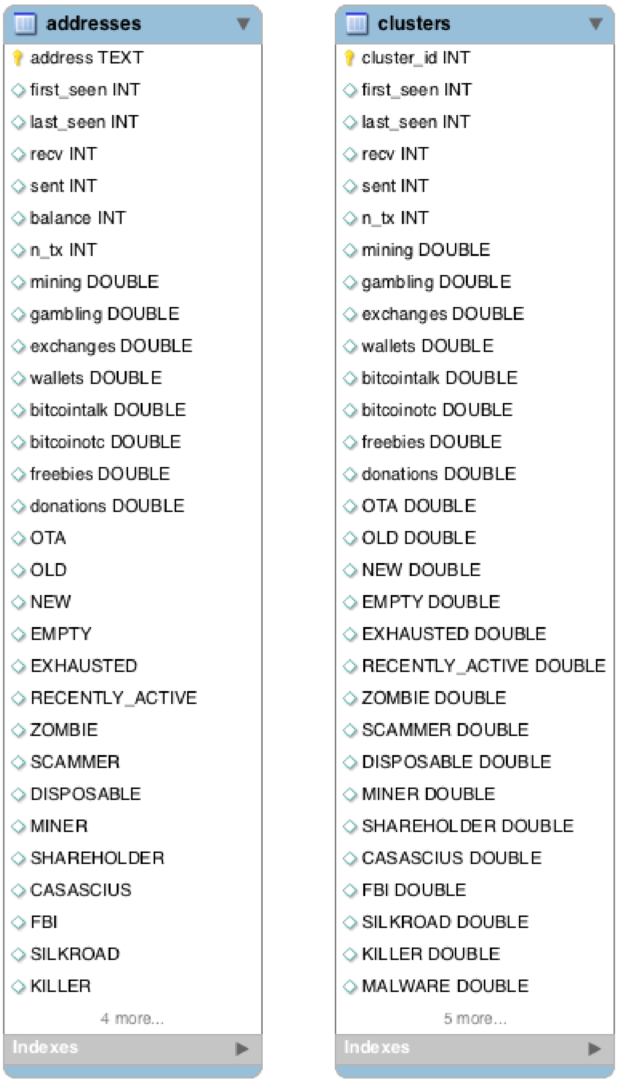


FIGURE 4.4: SQL schema diagram for the features DB

FIGURE 4.5: SQL schema diagram for the trades DB

5 Experiments and case studies

The goal of our experiments is to evaluate the correctness (Section 5.2, Section 5.3, Section 5.5) and the performance (Section 5.6) of BitIodine. Since BitIodine builds novel knowledge, there is no ground truth data available to validate our findings. However, we were able to confirm our findings thanks to contextual information found on the web resources cited in each case study.

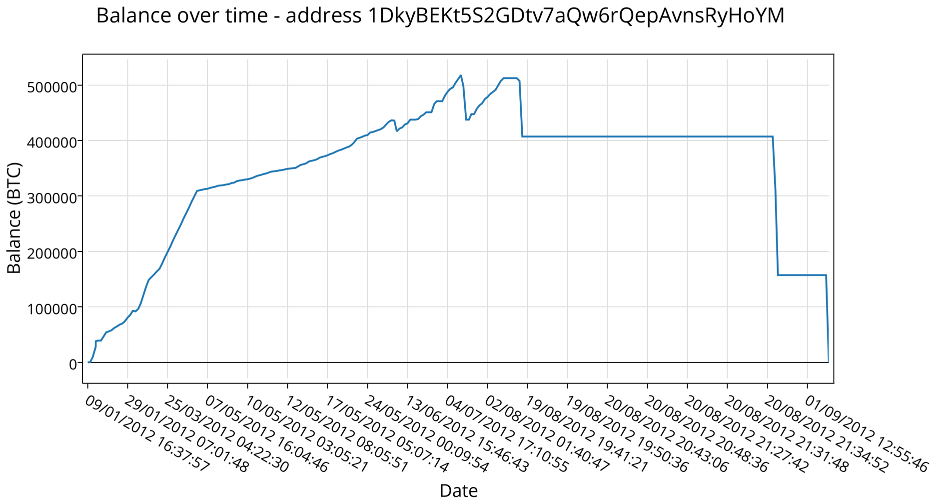


Figure 5.1: Plot of balance over time of a Silk Road-owned address

5.1 Investigating the Silk Road

We would like to identify the addresses owned by the Silk Road, the large Bitcoin black market.

The *block parser* module allows us to find the top N addresses ordered by balance passing `balances -l N` to its command-line interface.

Using this functionality, we see that one of the addresses that moved most funds on the network in 2012 is `1DkyBEKt5S2GDtv7aQw6rQepAvnsRyHoYM`.

As we can see in the plot in Figure 5.1, it has been accumulating coins steadily since April 2012, becoming completely empty in September of the same year. In total, the address has received 613,326 BTC. The address belongs to a cluster of 7 addresses, most of them input of very large transactions.

At the time, the Bitcoin market was not very liquid, and it was simpler to detect single trades using the *Mt. Gox scraper*.

By analyzing the activity of the address with the *block parser* module and by querying the *trades DB* populated by the *Mt. Gox scraper*, we find that:

- On July 17, 2012 this address had a balance of 517,825 BTC.

- On the same day Bitcoin looked on the verge of breaking 10 USD/BTC on Mt. Gox.
- At 02:00 AM, someone sold 10,000 BTC at 9 USD/BTC, driving the price down. This is indicated in Figure 5.2 with a red arrow.
- At 02:29, two large withdrawals of respectively 20,000 BTC[1] and 60,000 BTC[2] were made from this address one after the other (11 seconds between them), and included in a block at 02:32.
- Mt. Gox, at the time, needed 6 confirmations in order to allow the user to spend deposited bitcoins. 6 confirmations were matured with block at height `189421`, relayed at 02:47:24 AM.
- A few minutes after that, at 02:52 and 02:53, someone sold approximately 15,000 BTC at market price in several batches (the only two trades for more than 1,000 BTC are shown in Figure 5.3), causing the price to drop below 7.5 USD/BTC. This is highlighted in Figure 5.2 with a yellow background.

[1] https://blockchain.info/tx/3cb4f452fd0e5719391a6f1b82cd80a329a86734350ca04cab81d0e94f94e709
[2] https://blockchain.info/tx/cea22747487e8a2824566fa362981782871fea50fdfb690a3b63d85bd3189593

Figure 5.2: Bitcoin price chart for 17 Jul 2012

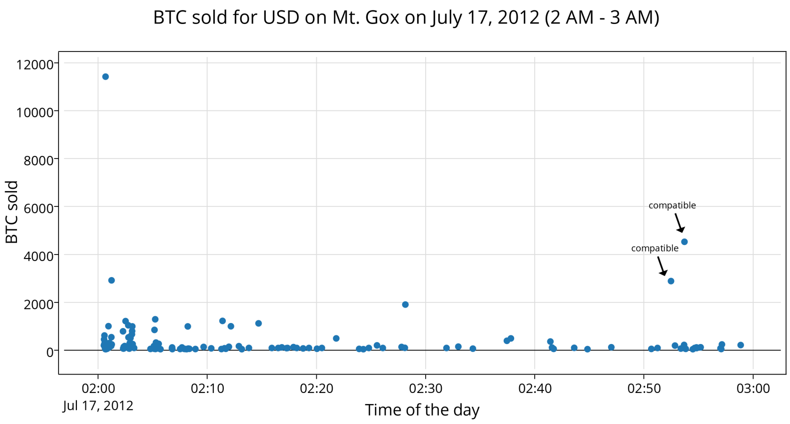


FIGURE 5.3: Two big trades on the Mt. Gox exchange

We can speculate that the owner of so many bitcoins was scared by the move down in price (or, (s)he caused that with existing funds on the exchange), and sold a bunch at market price.

Thanks to our tool, we are also able to prove that the address was actually owned by the Silk Road.

A BitcoinTalk user, on July 29, 2012, posted an important piece of information: the fact he deposited 0.001 BTC to his Silk Road account, and the address he was given as a deposit address.

According to the *Clusterizer*, the deposit address (`1Q6nyjSQ79AAw67xAGHgXxXHRj9erLLqhD`) is provably in the same wallet as more than 25,000 other addresses. This is because the Silk Road scrambles addresses, mixing funds and splitting them in thousands of one-time addresses in order to make investigations more difficult.

We can use the list of addresses in the cluster to find whether there was any connection between these addresses and the large `1Dky…` address.

The cluster is active since June 18, 2012, and there have been more than 80,000 inputs and outputs to/from these addresses. Since we do not see older transactions in the list, it seems the Silk Road wallet must have been reset around June 18, 2012.

By following the flow of coins, we notice that the 0.001 BTC is being grouped

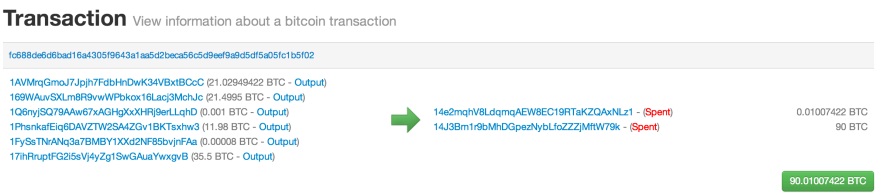


Figure 5.4: An important transaction in our Silk Road investigation

Figure 5.5: The transaction that links the address to Silk Road

with a few other Silk Road-owned addresses[3].

In the transaction in Figure 5.4, we notice that the deposit address and `1AVM...` are inputs to the same transaction. In fact, our huge cluster of more than 25,000 addresses confirms that both the deposit address and `1AVM...` are in the same wallet.

As we can see in Figure 5.5, a big multi-input payment to our address `1Dky...` has `1AVM...` as one of the payers[4] – this means we have spotted a multi-hop connection from our deposit address to the big address we were wondering whether it belonged to Silk Road.

Also, we can spot other addresses in the same cluster as input addresses to `1Dky...`.

Thanks to our tools, we have concluded that the large address was indeed related to the Silk Road.

[3] https://blockchain.info/tx/fc688de6d6bad16a4305f9643a1aa5d2beca56c5d9eef9a9d5df5a05fc1b5f02

[4] http://blockchain.info/tx/7c43eba80f90c770b8a5e3d196df7138fadbc62b2c81fc234fa48023bc23a8b2

5.2 Investigating activity involving Dread Pirate Roberts

On October 1st 2013, Ross William Ulbricht, a 29-year-old man, was arrested in a joint operation run by the cybercrime squad within the FBI's New York field office involving the FBI, DEA, IRS and Homeland Security's investigative unit. According to the allegations[5], he is the creator and operator of the infamous "Silk Road" black market, under the alias of "Dread Pirate Roberts" (DPR). From February 6, 2011 to July 23 2013, sales through the market amounted to 9,519,664 BTC (spread across 1,229,465 transactions), 614,305 BTC of which went directly to the accused as commissions. Prosecutors said they seized approximately 173,600 BTC, at date around USD 30,000,000, in the largest seizure of the digital currency ever.

The 29,600 coins the FBI accessed first were held by Silk Road in a so called *hot wallet*, which means they were used as an operating pool by the site, but the majority of other funds were held separately by Ulbricht in an encrypted "cold wallet", and should be worth around USD 120,000,000. On October 25, the FBI seized another 144,000 BTC. The seizure was operated by transferring the seized coins to two addresses controlled by the FBI. These addresses are publicly known[6]. On the other hand, the addresses which formed the cold wallet are not yet identified with certainty (or at least, this information is not public at the time of our writing). Using BitIodine alone, we are able to find a very promising connection between an address known to belong to DPR and `1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a`, an address with a balance exceeding 111,114 BTC (more than USD 22,000,000), likely belonging to the Silk Road *cold wallet*.

Ulbricht used to post on the Bitcoin Talk forum as *altoid*. Proof is that in the end of 2011, DPR was looking for a "*lead developer in a venture backed Bitcoin startup company*", and posted as *altoid* asking people to refer to his email address `rossul-`

[5] `http://krebsonsecurity.com/wp-content/uploads/2013/10/UlbrichtCriminalComplaint.pdf`

[6] 1F1tAaz5x1HUXrCNLbtMDqcw6o5GNn4xqX and 1FfmbHfnpaZjKFvyi1okTjJJusN455paPH

Author | Topic: help with Bitcoin development in php (variable parameters) (Read 1749 times)

altoid
Member

Activity: 48

Ignore

help with Bitcoin development in php (variable parameters)
April 25, 2011, 02:17:14 AM

quote #1

Hi all, I have run into some trouble using the bitcoin api with php. When I issue a command like:

$bitcoin->sendfrom($userid, $receiving_address, $amount);

I get an error like:

fopen(http://...@localhost:8332/): failed to open stream: HTTP request failed! HTTP/1.1 500 Internal Server Error

But when I hard code in the parameters:

$bitcoin->sendfrom("1", "1LDNLreKJ6GawBHPgB5yfVLBERi8g3SbQS", 10);

FIGURE 5.6: Forum post by *altoid* a.k.a. Dread Pirate Roberts leaking an address

`bricht@gmail.com`[7]. In another post on the same forum[8], captured in Figure 5.6, he had previously asked for help on the PHP Bitcoin API, pasting one of his addresses, `1LDNLreKJ6GawBHPgB5yfVLBERi8g3SbQS`, as a parameter of `sendfrom` method (it is thus likely he actually owns that address). This far, manual investigation is all that is needed to figure out the connection.

We used BitIodine on these data points. We run the *Classifier* module on that address, and find out it belongs to a cluster of 6 addresses, all empty. Thanks to our *path finders* in the *Exporters* module, we automatically find a very promising connection between the leaked address and a very wealthy address, `1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a`, as in Figure 5.7.

The chain is particularly interesting because every address appears in the blockchain with its first input coming from the previous one in the chain, and often addresses spend all their inputs to addresses on the right exclusively. In our opinion, this is a manual, rudimentary *mixer* or *tumbler*, and BitIodine helped in finding a meaningful connection between the addresses, leading us to speculate with some grounding that `1933` is part of the cold wallet of the Silk Road.

Although in this scenario there is some manual investigation, it would have been

[7] `https://Bitcointalk.org/index.php?topic=47811.0`
[8] `https://Bitcointalk.org/index.php?topic=6460.msg94424`

```
->  first input transaction of the address on the right
->  only input transaction of the address on the right
->  only significative input transaction of the address on the right
->  address on the left spent all its coins to address on the right exclusively

          1LDNLreKJ6GawBHPgB5yfVLBERi8g3SbQS
->->      1BG9jDV3pA1MsJUnvRyWuA2b7PfGd4MZaw
          5000 BTC 2011-04-30 18:32:55
->->      12h6TzwPNBvDnppbsqpyXwW4oo5UUKaKSa
          2000 BTC 2011-05-07 14:12:51 in a multi-input TX for 9067.32 BTC
->->->    1EG9HJG9aGqzgGujfNQMiNbyqpKnFxafvE
          9067.32 BTC 2011-06-19 23:04:29 in a multi-input TX for 37420.09314115 BTC
->->->    1AHki5AbZYiz4fHkGSTVKN3T1Tv5PwZpnh
          37420.09314115 BTC 2011-06-19 23:29:01 in a multi-input TX for 37421.09314115 BTC
->->      15TEAwEMxVS3BK718HhwgJg7nxwyJ2ib9y
          37421.09314115 BTC 2011-06-22 02:48:45
->->      1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a
          37421.09314115 BTC 2011-07-02 02:42:15 in a multi-input TX for 40954.56541907 BTC
```

FIGURE 5.7: Connection between DPR's address and a 111,114 BTC address

difficult to find a significative link manually, given the millions of nodes involved in the graph.

5.3 Payment to a killer?

In March 2013, a Silk Road vendor using the nickname *FriendlyChemist* supposedly attempted to blackmail DPR via Silk Road's private message system, providing proof that he had names and addresses of thousands of vendors. He demanded USD 500,000 for his silence. DPR asked another user, *redandwhite*, to "execute" FriendlyChemist, supplying him/her his full name and address. On March 31st, 2013, after having agreed on terms, DPR sent redandwhite 1,670 BTC to have FriendlyChemist killed.

Using BitIodine, we easily identify the transaction[9] to the alleged hitman, by querying the blockchain DB for transactions of 1,670 BTC on that day. The killer's address is `1MwvS1idEevZ5gd428TjL3hB2kHaBH9WTL`. This 1,670 BTC transaction is the first input it receives. On April 8, 2013 it receives another 3,000 BTC, and on April 12, 2013 another 2,555 BTC. Investigators could not find any record of somebody in that region being killed around that date or matching that description. This possibly implies that DPR was scammed, and that he was not the only one.

In this use case, BitIodine greatly helps manual investigation by allowing to filter transactions by amount and date in an efficient way. We did not have any address or transaction hash, so it would have been hard to spot the transaction manually.

[9] 4a0a5b6036c0da84c3eb9c2a884b6ad72416d1758470e19fb1d2fa2a145b5601

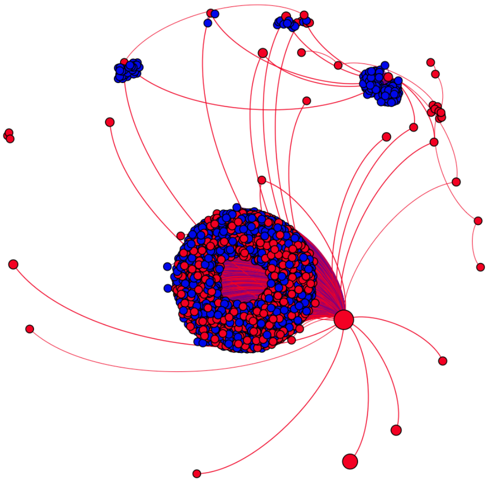


FIGURE 5.8: Graph of transactions inside the *laszlo* cluster

5.4 An expensive pizza

In May 2010, a BitcoinTalk user called *laszlo* from Jacksonville, Florida, bought two pizzas for 10,000 BTC[10]. Another user, *jercos*, bought the two pizzas to be delivered to him and posted photos as proof[11].

10,000 BTC were valued $41 at the time of the trade. In November 2013, they can be sold for more than 4 million USD.

The transaction ID is public[12], so we ask the *Classifier* to profile the cluster to which the input address belongs.

The *Classifier* is indeed able to identify the BitcoinTalk user (*laszlo*), and we also get more information about the user: the cluster is zero-balance, all the addresses are old and the last address used was on 20 Aug 2012. 89,211 BTC were received and sent by addresses in the cluster.

Furthermore, we can get more insight about the nature of the addresses used to perform the payments. About half of them are *mining addresses*, and they all got 50 BTC rewards. In Figure 5.8 we can see in blue mining addresses, in red other

[10] https://bitcointalk.org/index.php?topic=137.0
[11] http://heliacal.net/~solar/bitcoin/pizza/
[12] a1075db55d416d3ca199f55b6084e2115b9345e16c5cf302fc80e9d5fbf5d48d

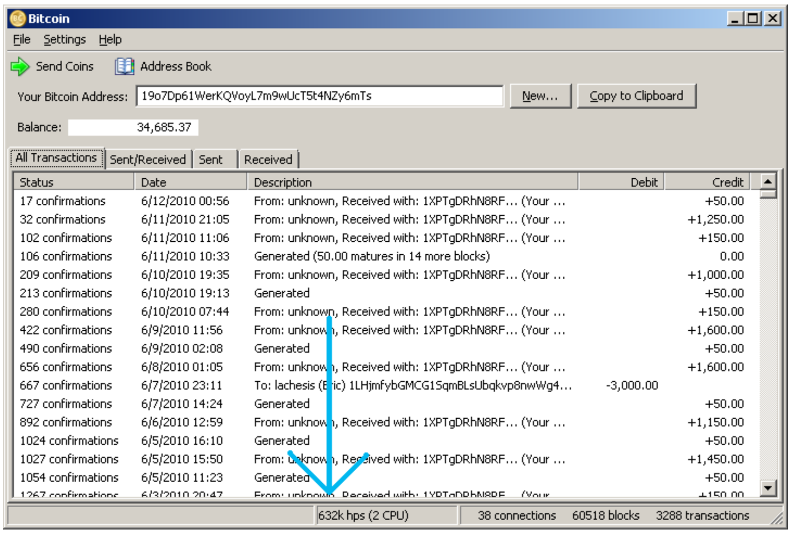


Figure 5.9: Screenshot posted by *lazslo* of his wallet

addresses, and transactions between them as edges.

Most mining addresses rewards are spent directly, some are first transferred to a big red *hub* (with address `1XPTg…`) and the majority of transactions is between other addresses.

This means that, back in 2010, *laszlo* was a miner that decided to give out some bitcoins to BitcoinTalk forum members.

This is confirmed by a screenshot that *laszlo* publicly posted on the forum (Figure 5.9), in which we can see the mining activity (the 50 BTC *Generated* transactions) and the big hub found by BitIodine (with address `1XPTg…`).

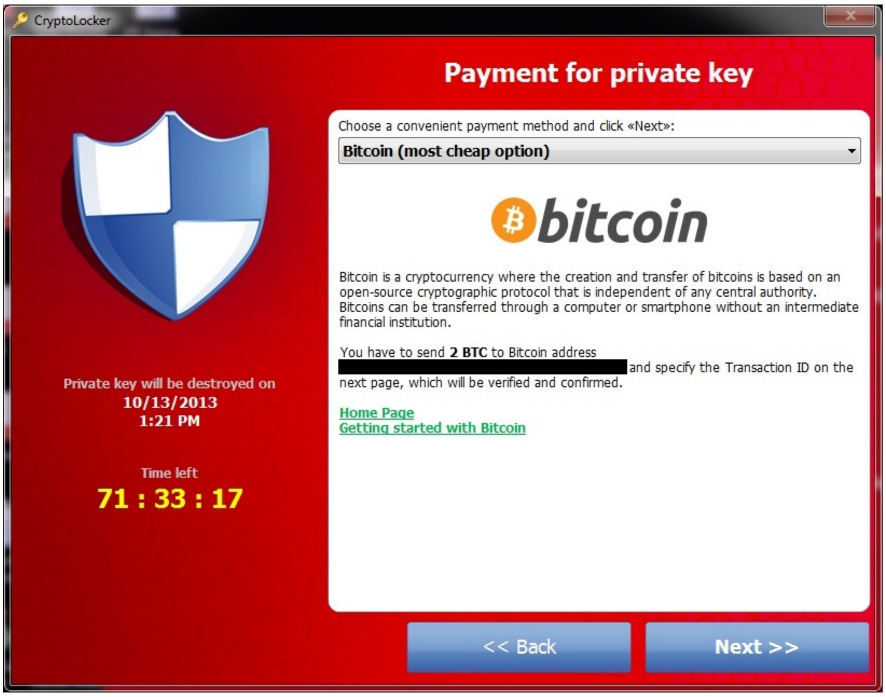


FIGURE 5.10: A screenshot of CryptoLocker asking for a ransom in Bitcoin

5.5 Ransomware investigation with BitIodine

CryptoLocker is a malicious program (malware) belonging to the category known as *ransomware*. It encrypts the victim's personal files with strong encryption, and the criminals retain the only copy of the decryption key on their server. The malware asks for a ransom to be paid with MoneyPak or Bitcoin within 72 hours in order to release the files (see Figure 5.10) [19].

As of November 20, 2013, due to the Bitcoin prices rising, the malware developers lowered the amount requested from 2 BTC to 0.5 BTC, and are also accepting late payments of 10 BTC after the countdown ends [30].

We use BitIodine to detect the CryptoLocker cluster(s), belonging to the malware authors, and compute some statistics about ransoms paid by the victims. By searching on Google for extracts of the text in the request by the malware ("You have to send 2 BTC to Bitcoin address" and "You have to send 0.5 BTC to Bitcoin address") and by reading a Reddit thread where victims and researchers post addresses belonging to the

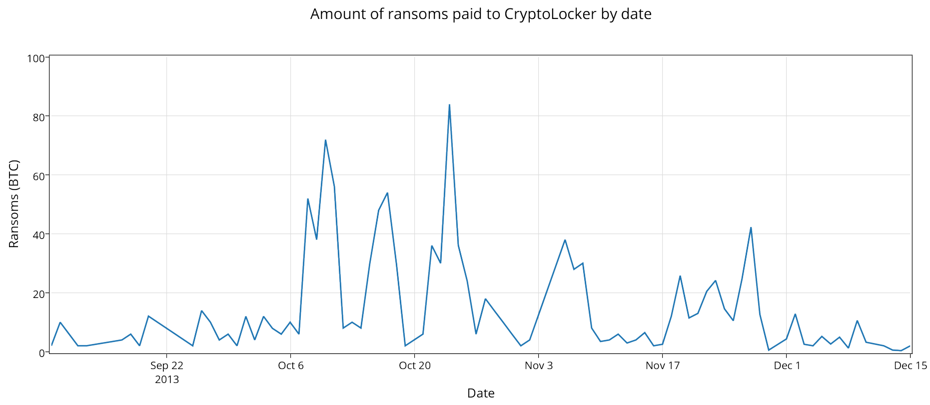


FIGURE 5.11: Amount of ransoms paid to CryptoLocker up to December 15, 2013

malware[13], we collect several addresses that are known to belong to CryptoLocker.

The *Classifer* confirms that they belong to several clusters, with a total of 2118 addresses.

Starting from these addresses, we ask BitIodine to add all other addresses in the same clusters and look for incoming transactions. Some addresses have been reused for several victims, while others are one-time only. By considering payments around 2 BTC (*original ransom*), 0.5 BTC (*new ransom*, accounted only if received after November 10 and before November 24, approximate date when the authors decreased the amount again to 0.3 BTC), 10 BTC (*late ransom*, before November 4) or 0.3 BTC (after November 24) (± 0.05 BTC) as *ransoms*, we can compile a table of addresses and number of ransoms paid before November 23, 2013. Due to space limitations, we include the full table in appendix. Suffice it to say here that there is one address with 26 ransoms paid, another with 23, one with 15, two with 4, five with 3, 41 with 2, and the rest with a single payment. In Figure 5.11 we plot the ransoms paid by victims to malware authors by date.

In total, we identified 771 ransoms, for 1226 BTC (approximately USD 1,100,000 on December 15, 2013).

[13] http://www.reddit.com/r/Bitcoin/comments/1o53hl/disturbing_bitcoin_virus_encrypts_instead_of/

By running the *Classifier* on a list of addresses of extorted people, we *automatically* find out, for example, that Bitcoin Talk user `caesar09`[14] is among the victims. BitIodine automatically found that username by analyzing the cluster to which that address belongs. This is an interesting, real-world, applied use of the *Classifier* to associate a username to an address not publicly known to be tied to that identity, thanks to clusters. It could have easily taken weeks of manual investigation to achieve the same results.

It would also be interesting to analyze the cluster related to the very first ransom paid[15], on September 13, four days before the others, because it could be some sort of *test* of the payment mechanism by the malware authors. We were not able to associate that cluster to a known identity due to a lack of useful data for that particular cluster. In particular, our *Classifier* found no known username or other identifying information. Manual analysis confirmed that no known nickname is linked to addresses belonging to that cluster. Therefore, this is not a limitation of our approach: the cluster might get labelled in the future as new transactions are broadcast.

Finally, with manual investigation following the funds from the payment addresses automatically found by BitIodine, we identify an address[16] that aggregates, in an apparently automated way, ransoms and spare change from insufficient payments for a total of 5,332 BTC (on November 23). Most of the funds are transferred to another address[17], that, on November 23, has received 6,757 BTC (approximately 6 millions USD).

This suggests that our estimate of their racket is very conservative.

[14] `https://bitcointalk.org/index.php?action=profile;u=153868`
[15] `https://blockchain.info/tx/285993fbbbe46b2e37090e76af17ea6be91db1c7ede5531056427800cf53251f`
[16] `1AEoiHY23fbBn8QiJ5y6oAjrhRY1Fb85uc`
[17] `1KueBqvskZpVx7zXiS62HFJkTNQdbvWzEL`

5.6 Performance evaluation

The generation of the database takes approximately 30 minutes on a Quadruple Extra Large High-Memory AWS EC2 instance (26 ECU, 68.4 GB of RAM), and its size is around 15GB.

The *Clusterizer* generates 4,077,114 clusters, grouping together 18,153,279 addresses, and takes approximately 45 minutes to process the whole blockchain using the same machine.

Scalability issues may arise as the blockchain grows, in particular for operations involving the transaction graph, which has to be loaded in memory. A solution would be to move the graphs to a graph database such as Neo4j, at the expense of slower queries (because of slower disk I/O with respect to memory) and a space occupation on disk almost five times higher. In our tests, a transaction graph updated to November 1, 2013 is 7 GB in NetworkX format and more than 30 GB with a Neo4j database. While Neo4j, thanks to the Cypher Query Language, allows complex queries that fully exploit graph structures, we opted for a simpler and leaner in memory solution at this stage.

Limitations 6

BitIodine is at an early development stage, but its application to well known cases shows that it already provides a good foundation for building software for forensic analysis of the Bitcoin network. Its limitations are about the first heuristic and scalability. Scalability issues can be mitigated using elastic computing platforms.

Our first heuristic, as presented in Section 4.1.1, works under the assumption that owners do not share private keys. This does not always hold: for example, some web wallets have pools that would be mistakenly grouped as a single user. This is why we defined the *owns* relation as $owns(a_i) = u_k$ if and only if u_k owns the private key of a_i.

The *Classifier* module, in its current implementation, needs to load the transaction graph and the clusters in memory, making classification a memory-intensive task. Also, BitIodine keeps data in two different fashions: in a relational database (the blockchain and features database) and in a graph (transaction and user graphs). This can be seen as redundant. In a future release, a single, efficient graph solution could replace the relational blockchain DB.

In general, we see an on-disk graph database such as Neo4j needed if BitIodine is used in production, even with the drawbacks detailed in Section 4.2. Furthermore, visualization of elaborated information is currently limited to exporting graphs and subgraphs, or even simple paths, by the *Exporters*. Providing an user-friendly web interface would greatly help to make information more accessible and immediate.

Conclusions 7

In this thesis we presented BitIodine, a modular framework that parses the Bitcoin blockchain, clusters addresses that are likely to belong to a same user or group of users, classifies such users and addresses and labels them, and visualizes complex information extracted from the Bitcoin network. Using BitIodine it is possible to label users and addresses automatically or semi-automatically thanks to scrapers that crawl the web and query exchanges for information, thus allowing to attach identities to users and to trace money entering and exiting the Bitcoin economy. This is the major novel contribution of our work. BitIodine also supports manual investigation by finding paths and reverse paths between two addresses or a user and an address.

We tested BitIodine on several real-world use cases. We showed that using a combination of modules it is possible to prove that one bitcoin address actually belongs to Silk Road, the large black market. We discovered a connection between the alleged founder of Silk Road, Dread Pirate Roberts, and an address with a balance exceeding 111,114 BTC, likely belonging to the encrypted Silk Road cold wallet. We found the transaction that, according to the FBI, was a payment by Dread Pirate Roberts to a hitman. Finally, we investigated the CryptoLocker ransomware, and starting by an address posted by a victim, we accurately quantified the number of ransoms paid, and got information about the victims, with very limited manual analysis. Even at this early stage of development, we were able to get valuable information about important addresses and the tool greatly supported investigation of malware activity.

We released BitIodine to allow the community of researchers to enhance it. Our hope is that BitIdoine will become the skeleton for building more complex frameworks for Bitcoin forensic analysis.

For example, Giuseppe Galano, an engineer at Banca d'Italia, Italian central bank, is currently developing, for his graduation thesis, *VIREXBC* (*Visual Interactive REaltime eXplorer*), a real time visualization software of the Bitcoin blockchain, using BitIodine as a base to interactively present complex imagery and infographics generated on the fly.

Finally, we believe that this thesis can raise awareness of the fact that strong anonymity is not a prominent design goal of Bitcoin. We accomplish that by showing that, using external identifying information, it is possible to group many public keys, associate them with identities, and observe the activity of known users in detail, using appropriate techniques.

Bibliography

[1] 4Chan. 4chan Pass, 2013. URL https://www.4chan.org/pass.

[2] Elli Androulaki, GhassanO. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating user privacy in Bitcoin. In Ahmad-Reza Sadeghi, editor, *Financial Cryptography and Data Security*, volume 7859 of *LNCS*, pages 34–51. Springer, 2013.

[3] The Internet Archive. How the Internet Archive is having great time with bitcoin, 2013. URL http://blog.archive.org/2013/04/03/how-the-internet-archive-is-having-great-time-with-bitcoin/.

[4] Ivan Brugere. Anomaly detection in the Bitcoin transaction network. Technical report, ESP-IGERT, 2012.

[5] Nicolas Christin. Traveling the Silk Road: a measurement analysis of a large anonymous online marketplace. In *Proc. of the 22nd int.l conf. on World Wide Web*, WWW '13, pages 213–224, 2013. URL http://dl.acm.org/citation.cfm?id=2488388.2488408.

[6] Matt Clinch. Bitcoin recognized by Germany as "private money". Available online, 2013. URL http://www.cnbc.com/id/100971898.

[7] Wei Dai. B-money. *Cypherpunks*, 1998.

[8] FinCEN. Application of FinCEN's regulations to persons administering, exchanging, or using virtual currencies. *U.S. Department of Treasury*, March 2013.

[9] Gary Duncan Francis Elliott. Chancellor Alistair Darling on brink of second bailout for banks. *The Times*, 2013.

[10] Jonathan Haynes. PayPal freezes WikiLeaks account. *The Guardian*, December 2010.

[11] Alex Hern. Bitcoin: what you need to know. Available online, 2013. URL http://www.theguardian.com/technology/2013/oct/04/bitcoin-what-you-need-to-know-silk-road.

[12] Alex Hern. Bitcoin price surges to post-crash high. Available online, 2013. URL http://www.theguardian.com/technology/2013/oct/21/bitcoin-price-surges-to-post-crash-high.

[13] Kashmir Hill. Five reasons for Bitcoin's most recent price surge. Available online, 2013. URL http://www.forbes.com/sites/kashmirhill/2013/10/23/five-possible-reasons-for-bitcoins-most-recent-surge/.

[14] Mikko Hypponen. Bitcoin mining bot that is controlled via Twitter. Available online, 2011. URL http://www.f-secure.com/weblog/archives/00002207.html.

[15] Matthew Green Aviel D. Rubin Ian Miers, Christina Garman. Zerocoin: anonymous distributed e-Cash from Bitcoin. *IEEE Symposium on Security and Privacy*, 2013.

[16] Paul R. Krugman. Vehicle currencies and the structure of international exchange. *Working Paper Series*, 1979. URL http://www.nber.org/papers/w0333.

[17] Gennady Medvinsky and B Clifford Neumann. *Electronic currency for the Internet*. University of Southern California, Information Sciences Institute, 1993.

[18] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 Internet Measurement Conference*, pages 127–140. ACM, 2013.

[19] Octavian Minea. Cryptolocker ransomware makes a Bitcoin wallet per victim. Available online, October 2013. URL http://labs.bitdefender.com/2013/10/cryptolocker-ransomware-makes-a-bitcoin-wallet-per-victim/.

[20] M Möser. Anonymity of Bitcoin transactions: an analysis of mixing services. In *Proceedings of Münster Bitcoin Conference*, 2013.

[21] Robert P. Murphy Murray N. Rothbard. *Man, Economy, and State: study guide*. Mises Institute, 2006.

[22] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Available online, 2008. URL http://bitcoin.org/bitcoin.pdf.

[23] Namecheap. Namecheap now accepts Bitcoin. Available online, 2013. URL http://community.namecheap.com/blog/2013/03/05/bitcoin/.

[24] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. Structure and anonymity of the Bitcoin transaction graph. *Future Internet*, 5(2):237–250, 2013.

[25] U.S. District Court Southern District of New York. Alleged Silk Road founder Ross Ulbricht criminal complaint. Available online, 2013. URL http://www1.icsi.berkeley.edu/~nweaver/UlbrichtCriminalComplaint.pdf.

[26] Drew Olanoff. Reddit starts accepting Bitcoin for Reddit Gold purchases thanks to partnership with Coinbase, 2013. URL http://tcrn.ch/174kSXw.

[27] U.S. Senate Committee on Homeland Security and Governmental Affairs. Beyond Silk Road: potential risks, threats, and promises of virtual currencies. Available online, November 2013. URL http://www.hsgac.senate.gov/hearings/beyond-silk-road-potential-risks-threats-and-promises-of-virtual-currencies.

[28] Fergal Reid and Martin Harrigan. An analysis of anonymity in the Bitcoin system. *Security and Privacy in Social Networks*, pages 197–223, 2013.

[29] Zachary M. Seward. Ben Bernanke's letter to Congress: Bitcoin and other virtual currencies "may hold long-term promise". Available online, November 2013. URL http://qz.com/148399/ben-bernanke-bitcoin-may-hold-long-term-promise/.

[30] Sean Sullivan. CryptoLocker: Pac-Man fever. Available online, 2013. URL http://www.f-secure.com/weblog/archives/00002642.html.

[31] J Doug Tygar and Bennet Yee. Cryptography: it's not just for electronic mail anymore. Technical report, DTIC Document, 1993.

[32] Ludwig von Mises. *Human action: a treatise on economics*. Yale University Press, 2009.

[33] WikiLeaks. Donate to WikiLeaks, 2012. URL http://shop.wikileaks.org/donate#dbitcoin.

[34] Wordpress. Pay another way: Bitcoin, 2012. URL http://en.blog.wordpress.com/2012/11/15/pay-another-way-bitcoin/.

Appendices

CryptoLocker – Addresses and number of ransoms paid

Malware address	Ransoms	Malware address	Ransoms
1KP72fBmh3XBRfuJDMn53APaqM6iMRspCh	26	18iEz617DoDp8CNQUyyrjCcC7XCGDf5SVb	23
132ckcKGxofNhZfG7bm9dv8kZGtVpNHwpr	15	12dGtJF5dM4QPZfYwZk7nsY8ABRZo4LBJh	4
1J6D7xqYBUj49atDRPVBCA3L22NL6pviZJ	4	16jMEqCjJn91ckzbw7MpKmc2ivG5PguYTB	3
1C4HXC1UK3b3ewKtySG2XaHz4wicE5iqwj	3	1Hx2Mz59JgTELeAgwpyMKpc35MqY5Rm5B4	3
1LMLFYedHSYBYDf6B1QEi1eM3gCHHB96gt	3	1LMRafbNmzG9H5g3GaAWjhX7wSnCXGNHMs	3
1Nz3P4pRebLsaJft83qYMTRvVsurN3BGKw	3	12mPaFLrUsqr18eh5LxJ2vjda4Tj79YNzB	2
13YooSYj267bUgRBny74esCGm4VKcMsJAE	2	13tvAGyKMGeUPck4yUFKffvWGTDESZE6xV	2
142JLE7k21dubJWKumgeFRA8cBga9BEWUV	2	14bh9uqKzgqV6yqTqHbU2bdbpY3a8pktzR	2
14zcNgMGuxWv5bb91jqibshwX8d5Q954Bq	2	15NaroT4CuEVswUcMvSHMVXeJynSd5LZVr	2
15qkjgvS11awYjyC7YoZ3KCgkXuNwQRaXv	2	17UYrYUqn4NDfgEq2EHr5qz62WRu4DGraT	2
17nKbdRW9pd7kco5fGcyHrbJxQepdWVWy1	2	18SP4vGAsTpd8yGjQ9GABQu7AKVaSHhAbs	2
19MNqUeDa78NfuSzJttaWbXVavc7mRZqj2	2	1AB1miU4GoRQzDggszBYd1gdMZqU265AUN	2
1At6BoXND9Bd2v58nG8ghXntNfvKA8YfXp	2	1BrQiciWXRtyoT3JDquQhKDKiWtYVLy2rv	2
1CRfmhUmjQRTuP4jnPPtUAffbkL6Dvfn73	2	1CTCrcgCzvVQn4KFm5AgoyeHH5fJW6ajdG	2
1Cvhg97rjbPLXKCBYVU6Qt5H9eb75neQnx	2	1DAQ2wmgcmFoDTFJ3jUmmzQczYRmeWkeLZ	2
1DZRHEosYzHfcARZSQJBMKWyLp9TuUhygy	2	1DsfbSx7miVyAztDacdDPmvMjPYKvQAHoW	2
1Dv38s2aEn3dU38DdL6MXvcUPo4ws2mnWN	2	1ECe37xTBbyXxZ5DhxYbw85tJttQ7zy8pd	2
1EeYeuEPWv3xWKGZ41QeLX515TkmQv5iA3	2	1FC8MEbXZsgmG8q86DQTwNy1hsTiwSCTmy	2
1FXUajdhKTjknLVq1VK38CcsdxLgytJM2E	2	1Fj8QatpHtV3FuqxhUEeYK8kmGcSN3EN3n	2
1GbvmuaQTbPiLYhFxjowiRPy4BR6UeEKhS	2	1HBwtE3ijt7ceMJJZ3ELfsUQ5scLTcguiC	2
1HFZF6PwHcuremLVrwAUznKRr1VpmTqBrz	2	1JWZ4gTAFcWYj7W4qq5XbpPKEvPSghnnda	2
1JpGVe6bGeeLZZN6kno7Mwt3KXesWdu3Tx	2	1JpzmkutgYkHZ8kAprdW3BVSVgLitE7J7G	2
1LaNYWVS5dq3FvXvA8PUK6hByQrGR7BTzQ	2	1LxiKcy5SH5RtPCqkEVeotw2JX5r5VLYYG	2
1M3oQzqk7TiPbxY8CqWNBd2czMa8NpgVxF	2	1MG5UCeoUbQyGteP4ze1CYCbSuR9t9y7oa	2
1MoWyT8Perux324bpWDhBdh1eTtvR8gw72	2	1PejiRQcz65paGK9vs5iu1iA4qoXQCscQy	2
1PuCT8f5Wk9MhBN3xPBE2r3BArkjyF24vP	2	1Q6T6hGvRUZBzWtMB38kE1qxw2N9RoRBnY	2
1147GrWQh7yW91kV65QP8BhFiGQQKuW3n5	1	125FCpENaKRu95psPP8gETM9u2EmvfAhF2	1
125vS1drhzddwkCAZfrMUjxMBU8xJ7Be9v	1	127tTWffUF93qp9jBjPiayBGpiKG9JgMfC	1
128ApausYAG5hCiRjzY5sKnaLAU7YSavC3	1	129U88iERn5dpjm2gdeaJBQWA87zeiULY2	1
12EEdNrva2FgqMHv33Maj8mNXXgWRy17Lq	1	12FnQgEWdqkKcRRM3TzW31SagyDBR776xf	1
12GtimBbqDvXrT5ExxoyWMKQHwwGSvg5iZ	1	12Lk5fHNJiR1YVR7aEHyF69fXqSnFcJUxb	1
12N1Z763yEMqKGGXTmWmMunhVim5CggcrX	1	12PXDTJFV84AWLoCiFNcd2Vd6wmukZWEDu	1
12QdTVrr9PodpSnP6uoFcJmudUVR6DNFBj	1	12Z71WcLDC6pLQiyWATU2xx91455PeE3ki	1
12auVFnXMMizCSAfFLXYpgtkAgvgpC34np	1	12bVTY5ZkmGXQoTYFgLrF7xdQeAKto6DT3	1
12cZEAGkTCnHmvgKP4GK1jXq36WrUkDiMh	1	12dvJyX6pzPxNvTi9u11RpDARHXsCjuy75	1
12i8GBDDLx93AnCdsGQTdN46EcDAz39vXa	1	12iz6uan4nWLU4xpuebxjdM8oJqcvKGM62	1
12k4mMYpaX6vgNFzL7qnFhsvy9BmHMUn1J	1	12vNSdg6TeS259ZJBuJu2xZrAqVkmNqSu9	1
12wQNVBEXuynSvXivTBstqwobjbRUJiHTg	1	12xkY3XezVZUsetYYxutJSUUSLQ2byyRBJ	1
12yzWvwsqkW6bAUY4pN2mrUmQo6hRjXujb	1	12zK22ydWiUWLVcG3DXZsX8crFMv82R9u6	1
12zWkbSTuKKe2mKgKJrrmZ5hm2xCDZJsQC	1	132R8sHpyvdchXJjPD8cpsA6Pw4gmWrJS1	1
137tDrDAoh24crTwNqTcAa3HanFVupE2dB	1	13B4z9sf3NcwopkcX2ZHz1eu67XgtHr3yR	1
13CHWK2e4gB7CS9KdR1cNy1J8mwj6E6AAL	1	13CtRbwChs6jrC3WDP7XxU5EzSmQ3eAhvP	1
13HPsTsERexydQNNkMPjjCPixFvxWpgPs1	1	13Hs65DrTNJfGC7Nvdvv1WL2kMf3vXpdCf	1
13LDpTY8c74hBvxiSEJKaMQg8zmzYJ74rS	1	13LihD38jd8Tsmr6YZLAqFMSDWHebXe88N	1
13MjaSU6vKCNgWLB9dVi2gZ4LNeAUSokMr	1	13N36rkC9Pbei3Z2fyNYvLd8wJwMxdXji8	1
13NKTcbc7Tjhs2cSywwDF4qTihqdW5FJzu	1	13PBA8pCYJTKWTVSzG1kcVfKhJ2cKQPKhu	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
13QfgCjpBd6vWwjf4NPa6B8DejnjozYhCQ	1	13RZBcRRFWoee6mMxBjB6NuB4PU6U9ZQay	1
13S4DGMrq2FNeMCR6w3TQwVGGDN641kMTK	1	13Tb67ue2X9zSb6ngopgp27DL6gV4YUPZ6	1
13ThBrrwbFLTyL2J7Vnh5QoM5jBJAB1quT	1	13Z9opdJAckCnLuTJ17wkWvUBbwbmeuPdG	1
13Zim8VJZTxYM8zrDH9DnjMt1hmkJR4ozn	1	13axjN3e7hRSKwP1SUcbvU2Mk6oJi1pdM5	1
13envUEGi1d2PdErDVu8Wym7jcDh1YwgqK	1	13o3SLKPZkpphJGa5jcLBe4WrpMPJbrG4s	1
13oEMXkrxQ7bzHhNNp9i2qW8ngEibt34w2	1	13oy8a8Ytfwue3oAR3nBG796nFNyKnmhMg	1
13oyQKhsb1PhEMfNyj2fEkaDMbhKCpWv3j	1	13wLbTLLkUgZCkeREDQEzmrhms4hQtqViY	1
13wWnCAZUKM2eBb9fYKfkenK4wsQKiiUPx	1	13yhUDumXgcaoSe49GUjwCPYbrcibEhCx6	1
13zgH2V2LBdg4hC6nemhUc6VAj2HYa1FZb	1	143nAiwSq6GzwhadnwBfiWNLaEu5Lar7tr	1
1445gF1w48JC6Mbg5WjHphqvNPt7Axpj3M	1	144eFKYQJ2hUL9hJ9v9S25X453Cbtpvej1	1
149DHg7KuLX5132CELLcf1zJ1tAi68mFNo	1	14AX3nqeZFaet3MbKtFWcQVCT3yJAT6pzC	1
14ETSyJ5Mvi3nXvX8HUjhhr3LSzwqb38Qa	1	14FMhAYUJ5CSTTTLGNZpZUWfrA4dgkakeG	1
14FQo3JR4zFg3PxX36F74SY4Vm47JFuiAw	1	14FTQGLQEwndyctnHSxwDVMHmKnjER3DqR	1
14GkLnmsjpsDUY7UJvPFy5BG1cUPLpiTCR	1	14H4ZyQVM39jCJHyPaYtJ1uay45PuzxyHS	1
14HBfQk71U44ov8Tg7WgdEKutm3nP5oqrB	1	14KE194E9kfxxjYCZrSCXQFuC1qfHsu4zL	1
14RPC2uw5cWDpsLj8PXjCm4LkiQpNViDjS	1	14SyfoJmS2gQNvLcZw1NV4RbPWL5wRChDY	1
14UNup9ys1w5HCC3zEj7nynF7aSFCJiyNQ	1	14UPJZhzXV6CEty3c9RR9XkLGa4pLkEfWp	1
14UWPLULEXvtUz8Pc46AJTCLmFpg7FS9LN	1	14jDrNMDJi2MPnBJVWTZwB6aU4A9be3Thd	1
14oc1oA2uuybtFUnyxJVkwKtDyNk5gXgDv	1	14uW4eDNRPUZW5MzE3Cbbsn7DkzBSTs36x	1
14ujLQwtnQjnxyp7KtyHKhj8dsgoBkTwcb	1	14y9mdPgoHKW4jAGniWiKJa1dHsKoMHnr9	1
152k2fQnttYX9BLTnHge9mnJeZNeKkwX7t	1	153RFdk3UJ1pZqd4hBVeguiPxphEfoujyG	1
1544c47YAMaHtgqmNmy86hPjJrmEaXxvdC	1	1548XtfEVgbKjkzbB3KeZUcmf8vyWWZcae	1
154CfE38JJdkGK7jB7iDAXPFRGVmCSZLCb	1	1592QxvdaHbFuAoiXaR7BBnmKrGtwLNVBU	1
15A33N6NYFZXxB7xhhDZ83i2ATXoczBMKL	1	15GPKoAvNjsK6Fzb1VmF6bxL8MmYcwUJas	1
15GStXVRzaFEvJtA41CKfd5WmxPiEZ2fSe	1	15K49Q54yrHgJCKs33RsdaGhJh4Cge99MW	1
15Kgbsv1GpF2fHcV3t1xBKoYk9RywwDWe3	1	15LVzgJ27r8utkvVh3eu11YQ61s9TvWQib	1
15MExBr8yweakyY5jDJz3mZVuuDB5KrUUW	1	15SMH5YbWuixjghZ5AubhWDkRFNfLLkS9Z	1
15UuRJce34S4NqaEwkCrdw3BFSFdB4TyZt	1	15VjFAytgecHhjZXkNkbGdiC4sUEAKxuux	1
15XVKc4MPNxeHmdfWjX5nVFdT2X9hhuxfL	1	15aXu2EhAvpWDEcoWEcWeG73XCNe7vRjz7	1
15dv1zNWJEa6t7dcSXGa9JDCUPFYNeGo6G	1	15eQyTL43nxzPPbLMf28UsbbXPeKKkZ8gV	1
15kXk9a3LbtECeMHpWE2AqTvPcu9VTaTFp	1	15ouuBb9rwfevxzDWKYYc2yNNPnfjjGeN7	1
15rkjyqBoKWTjbQSb12jCxAyBEYHm515GK	1	15vKEVHThcmCCepzx91AQTFbD1bp6SNLi7	1
15ypJ3NZB38SWsvA2AKGd8vCTp5kjodR6a	1	165d1uBCceZvfG2RXkRHJY3px4GYNdTBtM	1
166yMjKQsEFngzBr4Ds459dmtWY4hGepQ5	1	168bWbZF6VcQnVm1wBoeKWKPyhy771W6dx	1
16AZ2L7HiKwTxG3chtzKs5qQQPeexkEELA	1	16LVKAnvfKgujUBU1PYSisoi33YDeA141d	1
16PDrs2uZncAcW95uo9bCiKqR6wc2kWpun	1	16Vn4aq2d7TXE4XqyQ5dbcZGTHAQhWZn8U	1
16XfNSRrUuLy198aFX3F13shEoRPzNcKdY	1	16XyoG3yB2SvWXFdQ2fy17sytCmf2RBk1b	1
16YH8udBjRPMLGiuWrAfGDAHMTQeVX6bm2	1	16Zaif2w3W7JHktZqvgrnNk7up2kFi914E	1
16aGi4L4navwgK3nyGzkFuXbSu3HYPtJHL	1	16c1UiYKg72aL5Y2wopkuWgMC8rSHhgWZr	1
16ffTM4mix7CWRjAF4WEgeRTWoMifyarkN	1	16gNsTYPz64LxXzx4hi3PEvbS1EBMa2zVr	1
16iLGNLtrjd9AvcwbFD71DcGGgFfiNsE52	1	16kPwnKPzVhZFynr7PVNooapx14o6btrsx	1
16mxyjFfFtQxUR7tnUCfdbqkHCkhv95AFc	1	16p1aBpxfAcW4R5MabhBjfGbEosdT3SHfq	1
16pWYY5LnZZnZQd6cy2bJkGzic2WudBKMc	1	16sDE6YYpCihXnAjRuSNCNNPiZo6wWEvAq	1
16sd6wwNke2AXMKgmjJ3yMXivmEog8xAfq	1	16uYjskZ5e85aZfrbYVM9fthDTrhTW9GbV	1
16vo4KisPbHmmNXq3ekAdtMDC1oKBxsxwT	1	16y6pGWVqbmJxN4ADT9PAqoUctWz2WbmuL	1
16z2bKh7tLyxKKcM6wutqTGB7MyHPPR3ym	1	16zenZ1MCjchhEN9v1cJ3wz7yMqvCZftbx	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
172fBq95Vw5qzR6TpNQDCjuj6Q3EiCPw7V	1	173eCvWDT3PBZM8a4Fiac7dHTjJTYfNJry	1
177HgELfgYkwNoXn3B4tMfA9X6bECwdpde	1	179dwoQeeDyiK8fdR14Efm7rNGm14bZcfW	1
17HsiiF9q5hzQLP31stAigDSrRBGqj7uXz	1	17Lx64PZ7uEV3owFgPTAUA8tcA8Quhi4zf	1
17MKykhda15eXy9arFr6o7cPjWm5kcbK4E	1	17QX3c8qh2zVCrRFidcLi6JEuaXx5J3EYC	1
17TgFQqurkG1ze4aHzjW9mXjekywvPoHk5	1	17TmrfBEWgXhW3DC9iWw7sobsD8rUb6TMS	1
17d3wUL8m2gmi3CAYB77n381vK6z213M5u	1	17fK93pKA6eAmqBB4vefWyi1AXyjTYr5ZY	1
17kEVhBMqJB67sYCJTJ4A3qfyB1MNL87MC	1	17mdKEAqspuuvtS6kcZZft7JBjscAQLjrL	1
17n1uri4Eh9uLQ1XWpjed1EjUrNkYTc7eB	1	17ntYg99aNqjVu4HEKxZHvp7doWXY5iHa9	1
17qvywrovR7K1WGszK4dQ1wmH4FpkiXukT	1	17ssaL5PDbsZswVaKNou4pGEkwP1ZeEjs7	1
17w9bMf2DEcDUbpfVFodb34LpJShHdCTRv	1	17xhPSYVFhRrsxWmX1pizKHmutCRHD33g6	1
17zKbutfsJrhxLn5RZiWGYu2DqYbpRt2AL	1	185QPodi3zaSVtBjEGAALKT2sdycWxrByv	1
187Fwi9MGJ4N31hcY6YNK8L152C755HmqU	1	188G3tCxaq3WuSXTww7w821iEgUpYhwsq8	1
18Aos3Rx72XdfZVRo6TmYGHYvkjGYRFy4H	1	18BvLh5Zjma7gTa9wWLSpLjY31j4SQR5KP	1
18CgGQm7gi8XNeJfcyWQnzXGyzB5HY5Sr	1	18EnkSexV3wjY6rmbBeziGunbP72Y9U1du	1
18GMEYE468HGTcEr5Nj2v8uC8Fiv5MNZGJ	1	18GZDbcEWxEVxcF3u8VjEsE3b3axoxJc5c	1
18H4tdfhEMgPug98pY6UDFTFMxTzqnUhhQ	1	18KoxE5QjASmtQoanJa2AFphtjqWvYJabx	1
18Mfsc5RVbzij1kew5HFg5nxdkDLd6knNp	1	18P5UpeD5Dn9F9r2G79KXzwWvdE9Jmxvx	1
18SghQmtGjDaUre3yyhsfBajeMrEDAkEtJ	1	18TzEwoQy8JhU3m8fufXwThEqNet9j8Yc4	1
18Ug2QUiEKqZTtFjpR6iui5oRP8Q4SkG4q	1	18VQRLUWGt8dE4B5Gd6yUqzBH1vozERMUb	1
18WWGX3vsAr4rHNJbwkvYDadswp1LSeVww	1	18Xi4fRqeCE9DcBx7h8Xcs3EpWZH9aZW5R	1
18ZRqw6vKj5pPff8iDaFDsfXgfkpjqiuqk	1	18cB3t3ob1HhoFALSqqQPpnyVs7A8pvPsC	1
18eFXVT8goCQPAvvTEzSJab5QB5pDdDLNx	1	18ePWwRrJvnH2YAS8bN9fPF1NT1Y91KLqG	1
18epejFt7wA1qUrQPsFNt9KSwVo57mYd83	1	18ixwb5si3AQBpjkUA91vGE3ESoynXd95g	1
18mWpTuUPiZXu8MxRzwNfKv57DQDfJ9azK	1	18rqSx5LMf7PYvxw8QTKgJWUV4LLsRxQxV	1
18upRKhpRr2GMtUW8cR6dWMNJEvzQXbQKc	1	18w41e49sSx3eSzm26L6iZ3rL9BAvzsF2u	1
18wxVCTAmN5FHUwy43fb2UnMGgASVd49nS	1	18zzMtaprqzSBuaFFxod73LZMz3Xspm2NX	1
193N9bw37LRjs26NmCSN8vERN76MnbNU2Z	1	193tqzCaEGbUDw114QwZYoLJvEoB3Ao5K8	1
195phaUWwSq6CEiL8avFs89FjktWbnAUq4	1	199K1rKLpCZXwLae2C6QvFa2jYMN5LFcPg	1
199mTAvoE5URLSovHDUwcJsTcviTrVEA2y	1	19Cx9zXJhfDoFsLpL5iyCMaXfCD8FAVcGQ	1
19GTqXwKmMW5FKDuLh7fjLyahq1Y2YPNmc	1	19GiP1bC5qYmyoxqBmeHcFoAdRgyk1C37K	1
19LHsVsHVbgtevDenVb2CNTDPM5hjxsBqU	1	19LwEWuBDqaHHG3sCmmoXqfUi7E4v8S8Kj	1
19QiXsxYQN4QwkdDq2ee72QKVvjbCU65ye	1	19Wem2PzT7BVyLYPxzcwULigX7ZotRoZa1	1
19X6YnufnD7zsb9GPPmseeEHmGY6ti5qzt	1	19XH5t4odCnzGKU8f2LWiA3uwHVJ7PM9L1	1
19Z2S2rGAuU93ShKvRguQdP1tZ2wh4emfp	1	19bJ2pXT8cEPDS1FCYatYX5QohnV7Rnvyv	1
19fYob7vM6ZZb8e3cCnZXAjpPiWbPHjkDR	1	19h9GRcx1ABHPEzVpmWqqA9TTgug6JNcBc	1
19ik4sK6mGnTgqKe9EcHxgFmTyQFJQHBsi	1	19jZ4bUspCa6zEKHyuxvnm3yWdYLk5na7k	1
19joSY2NHVpY6r5dEsjN7XJHMQV6tmT13s	1	19k3qpwke9NkxKoW2VExEa1ZvfqYrqGgL1	1
19kSg47CibsBZPwABbTRQN6B8porvaaWze	1	19rW64ioQs43Xt1DiQXqqc5cwEpmWENMr7	1
19sUq7btf49S4gMBKES4NUa5ez8dZqX84N	1	19taqqVnko8Ujg549mS5QztYuczaAj4TVq	1
19uRAxstbHYb7t7vAJssBBGUErXanBWXnt	1	19uVk8uzQwTnLJYXTyWH5pihmpYPWdfm2Q	1
1A2VrTUtBHg4ZA7mEv3YEEBNhfVgZmsdsN	1	1ADEh9XUaMegTkZnU1Y1WLqcGHg6Ng6CTQ	1
1AEhRR23Pg2tzWFPSjxNW52T2DDnbRpMGT	1	1AFiCbx7HVarBgFBP5xH8BdwNJKTRk3mvr	1
1AHKr7uEemLay1AfSMaMtjMiPBUvUL9EJU	1	1AK3bMYn9pzndzKLeREE7bKvAuRyVd5f61	1
1ALzUKGxniXVkcdoBERpNre19257QH3R8R	1	1AQGdK9vXfvP8jN1k1bG9hGk9Dfy9gfHxP	1
1AQHyJMVoro6REmNdb6q2LLmj8Ci5EQaSE	1	1ARTideV24Y8yHKfB5APPoabRffeuqdKhh	1
1AU4Bdf13bCkc2fTkyAQc4zA25kuJQbu8H	1	1AUvX11NzrtrZUho6yTdp7KBooyAMBcGY7	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
1AXEDzWtrYHBypc7yjByF4457svsuaM9Xm	1	1AXNQvRxpLXLF2z3WrrREq3LHgdbpsAm38	1
1AZ7eyuw5YucEaqvdwWCjioMRq4jdTtHEn	1	1AdeHsocAtHwk47rc4xP4L7bGGXtpTMEY9	1
1AfnpkhYunFmZWb7tw2vwgc3DB99CU9HzM	1	1AhNqm4nPTLCPLHmYuPaoNiVbxbMvnYqxu	1
1Ai8sfxU2E99WiCdwHodRXBjA8rvtiBfLF	1	1AoNBYpVUy5Me6kN7ufwaLBaFBgpGW9EgL	1
1AoU2rZrCfADR3jyKKX4roksTBnXWoGe3X	1	1ApY6uap9fCTHZZTnJLoFg5E6a1BRokfda	1
1AvxVKVYdi4tCuh7e6Xz3yxdGcmhURZkEF	1	1AyBaDLN3rgHKTcXh5jceqUNNSCwrjWJ7	1
1Azj2voL5AnDnGeND6DxjUVtv5MUNLkcok	1	1B4SjCEppw4jgJLLH6WHf7zzdp5gMLreqB	1
1B58CmceR7A9zSCtzgGp5EHy1cVT8Kg3TM	1	1B5MG4Hnxo5cnWKb3C8Zxt82yxdRb8AMmr	1
1B7Tjwf2VKPSUx6Cf7woKdgJ5auvzXgSew	1	1B7nB4Uesa7Td5xHEko7GA7WyVdNhUCHHC	1
1BAhkiEGZuToy5HdyFSTUMAwFwfLkX2M1w	1	1BBAoHPsTwHRTDWWyZGKoshw6FfSpEiyr2	1
1BF8JATdy7sB1v8XEyWH3paRnfEQKgUvZd	1	1BFguNQ3CWURg8TTqkiowF9dWbh4dgcqo4	1
1BHkG1zMXr2bj6HMNj5kv67pD6VSE6trDP	1	1BPJSfJxCKYv7UN4n8UgoDYVi95don4TnD	1
1BPYyfxXXfNiA9AWQZPQd6LGhs9cUUEkv5	1	1BQqNrGCeMK6uFfQ4PrQ3bXZqzaYbpDy76	1
1BcxSeU1ghEa2S5MCLRSbzUTEbiFkP7Tfg	1	1BgBX4AHDDyCNaHFUFdgn9HpUUbC4uNGuf	1
1BhyNjRFU1Ywa6LFREPNetexsJyArcoA2p	1	1Bi2JWqZgZuVGgWT998v5WA9hX8zR3RKwi	1
1Bkq2Fu1VZCG9c9hgTahV6yAkC4Rmm8ypG	1	1Bp8Fya4qwBqLpzRCKaFQEdv4uui9raFTh	1
1BtpgJP35v9QwgEtPcKvVLs6Sc2bMD5UpA	1	1BwknQgBUPGXbc7GnPrscdyoYqHieZvEAk	1
1Byox5Rh3nU4kfhQNEh7812QQDsGdBqd86	1	1C1PtDix6Qpr6SjQ6r9KvekWzSKbaVw4UF	1
1C1TCi71raLkCSq6JP5DJh2Z1cPgnuxVuC	1	1C4DD4Uzast5jLALkc22vy8TVccjbeRMkt	1
1C4dn8d9cRkbFKWHQGwXWG9kzGGViMZ2yw	1	1CB5MFBw8hNLq7fnBAnYMXmCYwv8AYCpnv	1
1CBhbofpCiHeYekqqvtsD4gN7VRXyy1kV5	1	1CBtKExFSKoePhBHuuEkSfr7f6R8qYJ4ik	1
1CCetcghJh8GcDbE2BjBaJArygYCoruCHG	1	1CCuc7hReEDzPM8hbVRnKMtMncTD117yWt	1
1CFR9kkrnXEmJnXyzwkQRAryCHmKZ9XaDU	1	1CKbDAqne5Y5NZkYG9uXoEoFxxwfL5f192	1
1CLMS4zKMkY7NymUKZYC3LcjMw8DPqXhBU	1	1CLjRj6Ff6WKf8MJYc1kpbNXF2Lgr3eAer	1
1CNEZs5akS2K9WhtzrBf5vKrzh8BjwNQ8	1	1CRT39Np5hGd3mBLSyewjJ2yB9WHmNpR3P	1
1CTM19Ri6f47VqqD5JA8L2v6tytttHejqq	1	1CcYTGyWVmoL9XX7UB4BxCFjkMucWLxjDR	1
1CgRV3xjTSnWKj37w7zs8Tc4yc22xFygb3	1	1CkwD26HPNaDBwUjQK6ZPmfpmWH1aiDHUa	1
1CmANXHMweSdJ7z6QGQvoskTMTByNQZXaU	1	1CmkmKftbxc27YcySVw25QzdhLG5P5y82Q	1
1CnLpveGjnL3T1a9h2N7ekZLQM9odjuxDr	1	1ComcQszPh95J8ppMzcR2cNik9CzHnxvfu	1
1CpnX11iY6vxveFJFKVUN23dBvZsHvqFwQ	1	1CrfUuTxjtjRBo9BBPbozqsUU74RHTDEua	1
1CsJoXHnkKrBhVhkf5W4jsGWrQF8Kt8hR7	1	1CtFWbp44gvmejR93rbjjBNGogsb1vdN8R	1
1CuMuLRpEAddm5mhpz4BfGsqp1a6Tr5KYt	1	1CuYHGqGAaKUr71NtEC3FS7fXqWePpHbKZ	1
1Cw2ZF6UjjSfa7ZVtRtRTb9riptMH9VY8e	1	1CxHPQfAbzMbXLvrewzyJS1d2j1qQ3Man3	1
1CxRjCpiXe1wW62GrKMQvE7mQ47eqYJ2UR	1	1D5NxztvrVSsBA5Yc9RCZJWQYJ5MgC7N8z	1
1D8M8Y99QNdeoDJPEXZRbAHrm2BCvkYXB8	1	1DA1aQfTd1eLuTWUGnNNjjC93MERUwR2Px	1
1DAeUa9K2unQzQP1zaGeuQbewV8spScf3r	1	1DBSRad2zK8nCRB6u2ThKHL1oa9JadinWi	1
1DCyZLjUbSCdWpqJqhx1pQhcPpz8CSA41T	1	1DDtqaaQMNf1mCztprvwSMXMg21t8GiwHg	1
1DEiAbvToEZDSvsvwzTdGz6BJ6gc9Wv2PB	1	1DQi15kCBtSYEdGqCH5F1DTZDvpj6x1W9y	1
1DS9WQvcY6suYHpsNejLqEuwkdanXSBeiz	1	1DW18Pt3qUXmojN3BjcnXkvb3TZAHKxUvZ	1
1DX4wWS6TsTS7QEBxhRDBigwUtbUUAip4i	1	1DXFJ9L8i8YTQomQnF9s7mmFwsBQ2UtDwy	1
1Dbo8TdBobq5m7GTHNNBDFZLA8Xz6isu81	1	1De2hmHPVfGnCTUKcmTeghnR1PX9KvxkbD	1
1DjLXiqJr3JSJDuELpa2GS1qtwYjfca9TD	1	1Dnyz1TqkEWGZdowH1KjdaRGQXrUWeG9Xy	1
1Dp9xaJ7vgpma9iyp9Xg4XDHZpX1qnW8RP	1	1Dsbehicbb8WmQiwMF6RMZqSh2LhrwRq82	1
1DsjoUo82Sr169usUW1yqUjnReffjJzLfF	1	1DtpRU49A2obVy3EHn5fAAkJrMhaLxgres	1
1DukvSVnaeGxYArhJcCVNbXZgxFjdJzNUx	1	1DvLkA7iv5BVbiSXLkT141UjYZew8xP1J1	1
1DwwPSfWQcDcnYbJi28xSPDuEbGdKP8xVT	1	1Dzms4coJ6BnJocxeicLykS7MnEtt6RQTV	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
1E4SCH63xACNgZT6uAgAzymxx4EUepDAns	1	1E4ZaMnkE2XAcyuFMiMKZ8xpRCUTd55SDe	1
1E4wTaMR2Tgyf9NVhQDvk5nWHYfE5Lrpgy	1	1E5XCdK29D1ZJ1PCwb4QMQK5ykwTyb3xex	1
1EBevRx4iG9DeLiCEQJd3xiVLvnFfEYLvx	1	1ECKTr4hU5Cr5ix7JCGKNHRgeBhP3nSSui	1
1ECtyMAbqHhsmyWqvPVE6cbPjNDUgwh7f2	1	1EDhbqWc1u8HEQn9SRzx4dK2CC4RTubcEW	1
1EEVmrKkLP7LmBVrrX9YAP3JwKVh31Covx	1	1EHjcjsP17zPNHMWT9tAZ5X39AZUTVGwEQ	1
1EKdXWq2RZqPKyfgY2FhzVsNGhkJB3aScJ	1	1ETnvsNw3P3RA1NK9x6KCAF74c8PFmd3EB	1
1Eb71AB53EUWyq6LYWaFqcmEnXgj7wiYur	1	1EbSeUyF3DTYMgApySovqCi2JvPRqy5S42	1
1EddPDLLdVn28FquRCaiV1Gk9DKPxyTRnU	1	1Ee6kAYWizdFG7bxiHY8W3NsgbCceU91U7	1
1EeDUzBfberCNb5AoSn1FKmZAiqegB882H	1	1EjoQZ5Rzoc3ZiJcP2doHxDh3NyZpGdhx5	1
1EmNKwkfxfVBiEJo8qRa7UfZTLXERYJm7E	1	1EoQAfpDVHAgbkNqAh4odyuaiyAo3csU5X	1
1EyNoagZyGi9G4ntcgyT3rZa2vy3qyTpVn	1	1EzoppkGf1MTB6U3NsaypS2Dq916EuTzn6	1
1F17BAk9BAT4GFzc1L1pc8R1dyDGepqb78	1	1F1maThkZaAhLjftMEhQjWKHXmTVN2q9Vb	1
1F4VDvqfJuBZqdmzyagMhhZSoqYLJSfWmb	1	1F5USwafW7bbgzPU7xcnaV7FJztBgGLzt7	1
1F9W8EmdJsQ67FygDAFxcKtefQqYCbyivN	1	1FCALPEBJjwxUJmPQHohabMCEgGuXZ6B6j	1
1FE32xcN5mPetyPqAcJSzSjgVeL84jEkWP	1	1FJ5ka9LhzYXtnWxR2Eg5Vz3J68dqRVfnf	1
1FKyUwJicY5GQWFzFBsoq6ZbPUA1iH8dbY	1	1FL7yXdVaAFPew6yFHoCBUXwmf1prHUNRo	1
1FLUcg2mLX5JEe5uePUJfrUPwzTrBceUae	1	1FPbuf6qGM4UvmyVLwZ42eXmXqTZTChWje	1
1FWMMeLcSEowNZQqihvtNUA3QnMXTWdjkA	1	1FXAevqm1uTVjZYsMP1H4qsUsM6jkXmr6S	1
1FXjExYfwrevdn2ebud7gqzGoAnsER2j1W	1	1FXuyMgajuYWJHYNgY43bhMFjw8vprBUTD	1
1FYuKG14vqGKpyQJDEw8id3n8JysYJuETq	1	1FcomMQ1nKoVM5RfSzrB4uUSxMKWKhNB13	1
1FdWKDQg6Zqh4s6Zc89hyAbRBAeNRUwiAz	1	1Fe5bXZdDyfQwJsfqGNTZn5b9SLd7etatT	1
1FeXvY7QRKXJxCgTRgtvnDUWy7bpQCpgP1	1	1FfpF7bcUNghqvWJfV8D7yeSwYMxKAgLSi	1
1FiJZbEPxEhyejt9AYUyZqhVrAz7zmD8S7	1	1FiqZ8kvve4H6kNM377wZoVM4vZv7rUrVf	1
1Fo5MU4CLnXnEhSSPgbP4J7B9ZDsa7NDuU	1	1FomF7gPpjvmhySGWNuG33zAPq2sBCymLu	1
1Fpg89LmrdWvCtkTZF4eTccCdoCrSqniGh	1	1FpnnbNyRYULiCqUWTBnpb2r9ke834EjLZ	1
1FpwqB5eR3v1rNvBHhG25uneZVe4CBNqp6	1	1Frx1yLmQDfwW7qwbjzJcmhSLV5UXbDqHw	1
1Fu4ZtjsiwoWWkrzFKeLcLzMCoWb7h8iXf	1	1FuSaQCKCP9FzvTToUXzJVfkBHvksTtmP5	1
1Fz1vD25Lfz2f8etMAEf3Lfo61YjBFKGxN	1	1G78aJr3UJSJCa9Kvo6UgR256mMZ2BC2xe	1
1G7QEjGLsZhJmwCoWjGc2pYXUNMtAquSes	1	1G83cC32bsj7EoqTcW7w5gyZGVdztqAD75	1
1G9s21qsoiMMp8xYXBRiDdiXH654pSgzsP	1	1GABxutuqRpC2vLwTaQ8s7TNPWSBVVLsP5	1
1GARaX1nng9n8SboqvAgABj7UG4tGoCRD8	1	1GDhTJf2NYNvs7Cf3zFhGtvp8PFArC4UnV	1
1GET3HUUucNJbynop4V3YXDsXNw1gW2jKB	1	1GGJk8DjqMSQSiGbtfJiYv8d63pq1m63Mb	1
1GGm1pDwN4qp3AyWfP5hsswFWiXm3TcHcw	1	1GKKgv1ti6KyCwTR7qeUAzGJMgpM5CQG25	1
1GLWrDUYcLxFpxsbRXVQgkP2ooXUt1oNvs	1	1GQweUgXZAtzbgrTYCBveUmtmL9d8VTiRn	1
1GRwhW2XH2bv9bGFCGDcaL5jkZgxp3Zdwk	1	1GTTj2YiHZc1JF6wdY956QHTAKQDYCdwNA	1
1GTxj5VvnMDqHvGTdzyxSP2n2cGQBELZ6c	1	1GY2zSQN33EUdSx5GYosFFNoz5XmiaKun4	1
1GZQuQnPzBr5thipDapTBGqLcQW1Ruz2nr	1	1Ga2126kt8LjDydfnkiHsVeraoza2bFKou	1
1GecZMfEtpWgEZaWXyGyehmuzXKCW9DhGL	1	1GfYHVDVvb2g6a3mK1xDTPB3CLiZ8Gs6Jz	1
1Gh2SQa7v2mMH749TNSmzi1MreC2aRQdjo	1	1GkYLJ2Vsq5qXpZYGuqjDpknNiin3tiukT	1
1GmwPizjg7a6LTdpr9mAYqDxxGEZ6fjau	1	1GnBAQXrjsJPtLD41p4nMKWGiJvhtGDwST	1
1GovQ81qq8JyBAzcqK8T6CSUxpHf7BuFF8	1	1Gry7e5ACm6iXr3Gn8idgZ1x5Q8L43kkA7	1
1GwRMGzPdKmaLmwB2Gy8mm6YFAMvjrDnBs	1	1GxSaCJxDoSA2rS1fUp9ozQgsV9U64UGSb	1
1H2rvLTArfMTNsWK7U4tNqvN9V5K5cDZaF	1	1HASDyP1vzmh5Va7vdE9bJtKAs7RLnPhq1	1
1HB6hmcTeCLjeLBcafLx5ZpMTbXKWMn9m4	1	1HEJo6qYfSxnFmCtGE7AzCE4hgBYPrdbt9	1
1HEYE618LxL37EFSW1WKUoWkNMhPo2Hd3V	1	1HGhCRoGbsvXm2dWrvN9PHnHcuLtKhvKyL	1
1HJefw65SQi2HeWys3B4ZsinbpLifmNSbs	1	1HL71LjsvspbhhKhWUyAV9XHt37s3zoy33	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
1HNVA4DQj3e31SxQwcYqUQJdKEaU7h3MYM	1	1HNiqcy59y3toikWoECd73XP8DJxGJ2sxU	1
1HRFyimD3gw1cZeujv16qbDdkVZzvWUHJ7	1	1HXEb9sDcGmLR29bMbTeFnvHPdeSrKZBsu	1
1HYHeBkQ8fgPFEtKVkCjCouZjwEbTm2HMW	1	1HYsM31uGzShVJzkAbVbCTwyCRNzk7xAcG	1
1HYtcZdXwi2YRBr7JcqB5BCej7Zuts9Lph	1	1HZFJBtpPXmpxi8v4iSrWRydkPfP77QvBT	1
1HZHtdkkrCgTpXzNah1s825SedX1zFj7C8	1	1Haq892MR5i9tHymDBAa13yFaGqsorMjYK	1
1Hd5zaKwM7PMTTb3kPeDdyeJK4jnnhSHqH	1	1Hd6C1xyNmtJMQ89BpNGFEZwkiPH54nbCN	1
1HmtPJehWEHoVm5im1sRrEh4ivfFtSLwKC	1	1HodueGYwUi71qfij7Hyf5kcq27CG7LqRW	1
1Hojtyssp7qz7RwT4eKRYtP2nrZ96H3qZg	1	1Hpb5KmRWwseh33e5Jd3Pn7ULSsDix19N2	1
1HrYXbD156rr4NXNXUQXcQRUrpsZsdK5Jt	1	1HsuVCVJaxqiv7UsPoG4HvHCmDAhtHtE9R	1
1HuXuo6KaUZUwRS5apbcDZz94MMgaoEBfG	1	1HuY8s3zB6msu3Kv7R9TtVctReyVbyuicQ	1
1HwF7DXbxxzvRP7U5uRqVh1jw2wzjt9mrP	1	1HwbWAqoqL5cytfZRCoaD3P4WwtdeVjjs5	1
1Hy3yJDW23FBXdco1edFv8xJ6Q7w9po4bR	1	1HyqQaTQmUT3FpnNMCBcpjBFv9BS7dnbZL	1
1J2vjU43q7Ga9moLvMz6vHEZtFGx5cTien	1	1J7GbwwtEtoXhrmCwBtSuuT4yZTxwdn8Rw	1
1J8kNsXiDkoaHH54UidfhQya3ztH4CVWhF	1	1JAdHKLyhUuE1W1Fwkqeseq5ybnzwFvM7w	1
1JDaYbuQD7PWvKhM3y9UbpemkHc4kybFxP	1	1JF5wnR7mPLR2Cga3LUQ5XDwv6XuBC6vDr	1
1JKJByjqRibD5oYbLujEc3hwSgQmWb1FNU	1	1JLHc1VFhGo1KtKYnpBJmRwCimuSj1baHj	1
1JM1NifNYWFyJBbw6DRAmXgzfWCvihvoHp	1	1JMhN3hHoYotkKNGEFRytc9JgiGCaHqYUf	1
1JNrW6qR2KC9WQT5KHd7aaWKbmUCWp16dq	1	1JRZwX3cmQxtDvCvmoova2j2ZXaX4V8Ekq	1
1JYJTrqDfZK9jXUwrQ9zb5EndTWRDAHKis	1	1Ja8vLupptShzK58xaVp8yq3zJzzCEEAjj	1
1Jbe1c8Z3k6FeF4q3FhMzjgsqxh91SxRXo	1	1Jbrbj66UWqNLxoMcmxxtnDPA9oDTRzKej	1
1JdJ3qFFBnXbMSyuxFosniB63RxxBGY6aj	1	1JgAxMvC4RjEtRLADCEH6YCFiBEyZgg77H	1
1JowuBwNg3GnhNTDny4zcE2rZtm2khL2oJ	1	1JqrSXoWKcjpSr1bfzu7Gbyiyho3CYVKGu	1
1Js77if7NxxyE2w4UahAZv1fsa3emq2Pb7	1	1JsMYv9rEe1FCDfejvJyviRqnGf9pStZ6e	1
1JsXconekGH1YyQLBT4GeCEv6KLWBUHBsE	1	1JukPadus2HCjmXWpMyac9Coobzd3AMSnt	1
1JwoR6p8sBmh62GKm16ZWChP63CVAESUSo	1	1JyxrkELCNMECqLkE57jEFMX5otmGgcBSZ	1
1JzbvZ8zvWecnf6k7rp5isW9QQfpHpXZrT	1	1Jzy4sUHKjCDxykXDuFCztPH27MAf59Tng	1
1K2SbvQSWtRaiUdvvPKNnuXhwm9wuHnKFx	1	1K4qn9vJFVMeiZ72cVbnn1QzzAJEPtrre2	1
1K5jNeFdq5KWdyPuPBpodmgzGJaGmVExhm	1	1K7SGqhu5qzWQsSxP8Z8K2VCUTR4AphaU8	1
1K8Mz2g1FzNEUrZPDe4ffXJ5kqKnUYqG4L	1	1KAeAQ3tWeQ3B7sQak9dWed4nj76HVaari	1
1KFqey6J9NP2vvUCpgEuvk22iFysVqCSPt	1	1KHB1T89jBPGseWDb7DFug5JLktspRkasc	1
1KHDAbeA4YWdMHCj3he4sAzJXarWX7HFht	1	1KMSt9JPspuszaNEaGA85afwhviG3hPsCE	1
1KMYBUHzLwgJCXU6NotVUFDkaCQHnMD7Ek	1	1KVy4ADmBt52JgQnrKEui7D9Ka2J6XsXir	1
1KaL5hmQ8apJWP67WhsJ1685cdtQjerc5w	1	1KarHM9eLhGjNAxN28dP6EHPG6CoVHPG4C	1
1KcCWoyq7YzBXsgoiEenXS7EtksmYFrEqb	1	1KfPJQ9HSod2WUwUzr4MK4ySpNnnWK9oAy	1
1KfY1pzLRHDPHzGjzqHyfrKmy3Z2Z4ifeW	1	1KiJiDFptxgogXb6iMfvsQ3bdfaZrUXuFs	1
1Kmku9uQ2ckxwFUnhqPTHdKqiJN57qBRTq	1	1KvqosZ3B996QwPg3w51eCZvorj5gVUFet	1
1KzHfDDpPs9f11bXTU9yxu67E3Yrh5THZE	1	1L4ypAukN5qmh4B8f6jVE5xu7yiUBveGrf	1
1L7WevwfDj7AzEXfi9j1BbAYyCZhjKLvBv	1	1LB6eL1iGZsJVWfhDLASmU2L9o8PhEzcc7	1
1LDKfUJGBEbso4gdbnJQ3HBm78H7hd1j6g	1	1LGfYq32BKZosBjaoK5isnXSj93VY3kBXJ	1
1LJusykjpqSbBU4LJwaRak41SHdPqyjf1a	1	1LK4EgUYGm5uE3xUWNr16nmYoYChn7TwQf	1
1LYVUUwkikgc6mApuqaeVfHvE2y687J43Z	1	1LYWiERCo6m7fEykJ4uuHuCtmeokV4haPy	1
1LZRPrhDh3b9EAs26jKhG2tmYNrJ9TC2cw	1	1LbJLb2kgqA2aZo5radKYsQP27wi7oNpis	1
1LbsNHA5yTtUPmDozodEpvRQpBRNtuGuhu	1	1LbwzyzGVGDzZ12r3fenoZwKfaKKR5SoUq	1
1Lm23UB6oaiAazN5toNp18N2EuVSvFwm78	1	1LmoZhPL7DXoNW82LmmzKTw1vEsoKUjK3F	1
1Lr5vY8SD8wC5UoGmFW1MiWcXSzFj7sKcj	1	1Lr9HrVPMqE6hBUor4F9bQn4oevazwGGLT	1
1LysaXN5BbfL1FQcQRNLVs2KFPoNhgeNZ9	1	1M2JeWxEsgPK643KP3Y23RAHVKzxMoXtwY	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
1M4A5HRARXyfMQgDWjSzQCpKqkMuCAHQdU	1	1M5GHvYVAtwJ9aZi2GWiNpNQwcimYReXqs	1
1M83NXYuPpjEjYt8baXYxriQNCDyfWU8i3	1	1M8D6Z3sCPCBmSbNDqcu4DHtTsSQw4B6QF	1
1MBhjH2kLBFoYLLucjKeuVhRhfuWZRGX2G	1	1MCJg35DzCuxd6JTQiDEhK2m5rVRcgFwxj	1
1MCPK5yQZnPBCYFDLsDtuHxxqDyAqoYtMG	1	1MDf3iwRESRcNAvcJqbMf7VXVoejUUzn6T	1
1MEVHp2divX2HUKzYY3bFRcM7fWtn3XREp	1	1MFFrqqbHSva1VTXoLN9dVVgqS2kNVmjUY	1
1MHmXHCQiJ578GXCuUosvTbwa3CH8wjqgj	1	1MPXvqtB923bhA22ru11TacfCagY3Chwhi	1
1MRss2agjSaScZg2wHEvURncwxKvQE44BC	1	1MS4zNjpMKqNmPC3Mq1gAp2tYNQB3bbrmP	1
1MSFtiA8DfnpxGJsUg7adc4q5T9rSFDnvC	1	1MUi4DPgYpknfPnYy6TtLPA4R3H4dYJEau	1
1MdtKc62oYkhjLvLufGcXvMz1KaYrp6p7D	1	1MwCwmKezKDkVeJo6XB3wgZAoxebWpRXCb	1
1My2nTaiG7bbtBdjUUw8huReufaoVRwLeT	1	1Mz8NVbYkWY1c1U8CsFKSZHa8VAzU6Fjd2	1
1N2bGYAqcZJ1nBDpFo2rhGq4kSCWDYgKsG	1	1N4hjEZd2KEz6d2Cd8JVCfJFvNJAQDtX8P	1
1N53V8f8NQPyHUHqj6e4mHCqrJsgsYMdq6	1	1N9DCCacu84xok24gJcYoj2hWmfEa9gHxD	1
1NBGp3YUVwMFc3vKvHE2XLD8S3uzPN9Ujm	1	1NE4Do4DZg9GuzRq7sruvvnTgpG27xGDWx	1
1NEc9b7C1cYXXHJgpVNfzJFPrYGhDWzFn7	1	1NMVXwfpZpvG8DECiv1KjFKvuUzVLS8FQ2	1
1NMiiPNGb93VwTXGrJY8FPvnLnq2ZSdct1	1	1NNn5Ybnk9DCiJR5ypVdrvf7WVxcqFWCc2	1
1NPZWdqghuS1SufK25wJWKpb8Ub7BbC95	1	1NPqptRx484Ntyt8SuZRPKVsekJHeYjTme	1
1NTqaT3skX1ButcNrDX9qxo9V8Z3CzdPXh	1	1NXVijH9dvaLD93UkJ2xxyd76831mXX4zg	1
1NZe4jSrKsjLHbYUpxSvXqVtKvesbHVu7F	1	1NZnh3sWpHWrP5TjaSJFHYLBN9jNL61RRa	1
1Nb7ULYgbkLH8Agx8AopVMVAEjnXHzTBSm	1	1NcSH3gPBpbLeUE4WZGHu5JVWnh6JPwaJv	1
1NeiNq9hZxAS75HMfzYFzxnYDpSGVEPDuY	1	1Ni2d6ntLrza9646HmdSgymseAibe8oyKG	1
1NkXbJhkb6any5KSN9jhaEw75xpxo4FM5C	1	1NmDnGWpGysUaKRYC7QWqwYPjuNsGNa6wx	1
1NmnkzfqEtaLCSjVAH3sc2cXgAREz21dk1	1	1Nq6H2kb95nAfZ7oH8NRm9nen11VG59tpa	1
1NqoLabo31gu3fopiTrWA3UVXkCVEmb9E7	1	1Nsrh7pZpz1ueSQWdRm7JHnw3Trkv79eZa	1
1Ntp9y37tWGpibxUs4jSzLkMnT4GRn9Txv	1	1NzQGLLzxWumfnP2he2bQhdmxu4ka5i9Uj	1
1P1yJN9BkUEPhqQ4s6umLJiSnKJfrff7XH	1	1P4wU3HaZT2ANajv4cnC2VD8t1WVUd4eo9	1
1P52Hgptcnre8ejGz6GtHsdwLQP9fPP3M9	1	1P6GDJ7NwsDygz6tyKPDGEuh4GRA6aXrgQ	1
1PARbtVpdQ2DpkMHRvYkkkuSGgSDVpFrSF	1	1PBZGajyhi8M9436Amt4SYLf9r2BMHCEPA	1
1PDEFwCuYBt1t2hny85Bvac5bJTZsJaftr	1	1PFdNRASdBV3sLEmZRgxR8ZC1U1s36Zjkc	1
1PFr6CqUrkjfdmeMm5HpUdwDYAAT6NJpi2	1	1PHq99kxUC9PNd8jYfvZ6K5ov3oGw3LfpV	1
1PLqvqitTbafqdmfMPPT8nKaJqJTVPxZms	1	1PMeia8xa4c3CnoAw3C2EPsXbeoTSpPm89	1
1PUHb4rzzVw82vGWrTx2esTqWV34qsiLoM	1	1PVTFqz6ZeuwqCKnEULF2tYUBoNocNFczH	1
1PVa2LdducsHXWgtAUMCrHRUhNyEgQYWNU	1	1PbnSDMsTJCvtWkz5vxyC618sjA7rQzpp3	1
1Pc4r1wBWRXmJiEBQWPrw9pTdF5srESZLP	1	1PcM4ytXhzSubRCLvVm21BG3os7gCjDAEy	1
1PcMY64nst8n8vLjkBkPGPdp9ADvKhYuMk	1	1Pf7ZDS6hmHNZHHkL9WRmRWoLpqJ1TeVTe	1
1PgE9vt9sfCyFj4rmL8uL9WqM2BqMXeE25	1	1PgttDwaoBn9fT1yCSS9SqW2y7XvutQhoD	1
1Ps7VKoKheZggkT49aREmxmaDzCDSS2jJJ	1	1Psk3JsaE97rubL9mu9GqM4S9yj1hmhtQe	1
1Pv5FYte855Ew8RDTeaqhKXFipJ28DrPte	1	1PwVZy5UBwvJ8KmmS9pe2AuQTv4nyjHywe	1
1PwsadYujh5RQMpp7QCedWshvbhkEMbrA3	1	1PxJj8iogRk8ELea4t8mobNt2LinZGZonw	1
1Q2A1uvYqNNWgVSKiZSdMYb9MrVeTSdiXK	1	1Q3BJuu8t99i23wNF4pk8rLsKwLWWT9xvn	1
1Q4LkTm1MRR88jRUNSnegGyYi8MpqKMGMC	1	1Q62f7bkkd3FUg7RrrEPwPmEZZGQhspNGF	1
1QA3Ho9g5b2U31gBCQzo4b5tnJVmM8R5Kv	1	1QCttxVQLvJMGGR1LFsRNxHfQeNWvH6zW	1
1QDSsu2oNhj8BcVXrgTXCXgYkHZ7kH9qx9	1	1QH52jLJDaD8zYeaZDM4M687z4FpdX7CZZ	1
1QJCsWZ8CCrwECfyxxfcegrvNAbJbg9Zks	1	1UCJ8wot2HqfEEBebANG36FTVMpwAE44r	1
1WWPwBRyUxs9qCKmf1QWQU1wsZbk8bGJg	1	1WbtRF7ZsqQjQPSXY2usAKJCBjDikDbCg	1
1WcrmM6iAHyoZWriGoMW2AZtFDSL8gYAU	1	1ZeekMuuqzDD3KNXe1Tg5pRugKEHsydWh	1
1aX2zTKBnZrtxGpPA8j77V2FfCn1EGSib	1	1dPwUZjv2fuWw8MCFm6UJYncQZimbHwx6	1

Continued on next page

Table continued from previous page

Malware address	Ransoms	Malware address	Ransoms
1fkey8rWjxhxgSaKscEfSHrENqz595nGq	1	1fpYNCzkX1jnmkDXVidvLH3iEsJqS1w7B	1
1iPwrjCp5761ofo5xbAf89xaaSWZZAP7K	1	1tP8PaTzUCyCfVahS71jV58ubtNn7zfhH	1
1v1hLeFZczCBQeuDRyKwjXrs5pKWqp6kB	1	1vtbjSigZsfg9APnoANQUpG3nsYczRPy1	1
1wK4k9TMx41KUrW8D51QCNvQGrEdVkEwe	1		

List of labels for addresses

Label	Type	Meaning
first_seen	NUMBER	Timestamp of the first appearance of the address in a transaction
last_seen	NUMBER	Timestamp of the last appearance of the address in a transaction
recv	NUMBER	Total amount received by the address
sent	NUMBER	Total amount sent from the address
balance	NUMBER	Balance of the address
n_tx	NUMBER	Number of transactions in which the address appears
cluster_id	NUMBER	The ID of the cluster the address belongs to
mining	RATIO	Ratio of transactions coming from direct or pooled mining
gambling	RATIO	Ratio of transactions to/from gambling sites
exchanges	RATIO	Ratio of transactions to/from exchanges
wallets	RATIO	Ratio of transactions to/from web wallets
bitcointalk	RATIO	Ratio of transactions to/from known BitcoinTalk users
bitcoinotc	RATIO	Ratio of transactions to/from known Bitcoin-OTC users
freebies	RATIO	Ratio of transactions to/from faucets or other freebies
donations	RATIO	Ratio of transactions to/from known donation addresses
OTA	BOOLEAN	One-Time-Address: appears in just one transaction
OLD	BOOLEAN	Not seen for a long time (*tunable*)
NEW	BOOLEAN	First appearance is recent (*tunable*)
EMPTY	BOOLEAN	Balance is close to zero
EXHAUSTED	BOOLEAN	EMPTY and has received more than current balance
RECENTLY_ACTIVE	BOOLEAN	Last activity is recent (*tunable*)
ZOMBIE	BOOLEAN	Was empty and dormant for a long time, then got used again
SCAMMER	BOOLEAN	The owner is marked as a scammer
DISPOSABLE	BOOLEAN	**OLD**, a few transactions in a short period of time (*tunable*)
MINER	BOOLEAN	Related to mining activities
SHAREHOLDER	BOOLEAN	Related to shares on Bitcoin stock exchanges
CASASCIUS	BOOLEAN	Related to physical Casascius coins
FBI	BOOLEAN	Related to FBI seizures
SILKROAD	BOOLEAN	Related to the Silk Road black market
KILLER	BOOLEAN	Related to contract killing
MALWARE	BOOLEAN	Related to malware activities
BITCOINTALK_USER	STRING	The BitcoinTalk (forum) username of the owner
BITCOINOTC_USER	STRING	The Bitcoin-OTC (exchange) username of the owner

Figure 1: List of labels for addresses

List of labels for clusters

Label	Type	Meaning
cluster_id	NUMBER	The ID of the cluster
first_seen	NUMBER	Timestamp of the first appearance of addresses in the cluster
last_seen	NUMBER	Timestamp of the last appearance of addresses in the cluster
recv	NUMBER	Total amount received by addresses in the cluster
sent	NUMBER	Total amount sent from addresses in the cluster
min_balance	NUMBER	Minimum balance of addresses in the cluster
max_balance	NUMBER	Maximum balance of addresses in the cluster
avg_balance	NUMBER	Average balance of addresses in the cluster
n_tx	NUMBER	Number of transactions in which the addresses in the cluster appear
BITCOINTALK_USER	STRING	BitcoinTalk usernames of owners of addresses in the cluster
BITCOINOTC_USER	STRING	Bitcoin-OTC usernames of owners of addresses in the cluster
mining	RATIO	Ratio of transactions coming from direct or pooled mining
gambling	RATIO	Ratio of transactions to/from gambling sites
exchanges	RATIO	Ratio of transactions to/from exchanges
wallets	RATIO	Ratio of transactions to/from web wallets
bitcointalk	RATIO	Ratio of transactions to/from known BitcoinTalk users
bitcoinotc	RATIO	Ratio of transactions to/from known Bitcoin-OTC users
freebies	RATIO	Ratio of transactions to/from faucets or other freebies
donations	RATIO	Ratio of transactions to/from known donation addresses
OTA	RATIO	Ratio of One-Time-Addresses in the cluster
OLD	RATIO	Ratio of OLD addresses in the cluster
NEW	RATIO	Ratio of NEW addresses in the cluster
EMPTY	RATIO	Ratio of EMPTY addresses in the cluster
EXHAUSTED	RATIO	Ratio of EXHAUSTED addresses in the cluster
RECENTLY_ACTIVE	RATIO	Ratio of RECENTLY_ACTIVE addresses in the cluster
ZOMBIE	RATIO	Ratio of ZOMBIE addresses in the cluster
SCAMMER	RATIO	Ratio of SCAMMER addresses in the cluster
DISPOSABLE	RATIO	Ratio of DISPOSABLE addresses in the cluster
MINER	RATIO	Ratio of MINER addresses in the cluster
SHAREHOLDER	RATIO	Ratio of SHAREHOLDER addresses in the cluster
CASASCIUS	RATIO	Ratio of CASASCIUS addresses in the cluster
FBI	BOOLEAN	The cluster is controlled by the FBI
SILKROAD	BOOLEAN	The cluster is controlled by the Silk Road
KILLER	BOOLEAN	The cluster is controlled by a contract killer
MALWARE	BOOLEAN	The cluster is controlled by malware authors
SCAMMER	BOOLEAN	The cluster is controlled by a known scammer

Figure 2: List of labels for clusters

SQL schemas

Blockchain database

```
CREATE TABLE blocks(
    block_id BIGINT NOT NULL PRIMARY KEY,
    block_hash TEXT NOT NULL,
    time BIGINT NOT NULL
);

CREATE TABLE tx(
    tx_id BIGINT NOT NULL PRIMARY KEY,
    tx_hash TEXT NOT NULL,
    block_id BIGINT NOT NULL,
    FOREIGN KEY (block_id) REFERENCES blocks (block_id)
);

CREATE TABLE txout(
    txout_id BIGINT NOT NULL PRIMARY KEY,
    address CHAR(40),
    txout_value BIGINT NOT NULL,
    tx_id BIGINT NOT NULL,
    txout_pos INT NOT NULL,
    FOREIGN KEY (tx_id) REFERENCES tx (tx_id)
);

CREATE TABLE txin(
    txin_id BIGINT NOT NULL PRIMARY KEY,
    txout_id BIGINT NOT NULL,
    tx_id BIGINT NOT NULL,
    txin_pos INT NOT NULL,
    FOREIGN KEY (tx_id) REFERENCES tx (tx_id)
);

```

```
CREATE INDEX x_txin_txout ON txin (txout_id);
CREATE INDEX x_txout_address ON txout (address);
CREATE INDEX x_txin_txid ON txin (tx_id);
CREATE INDEX x_txout_txid ON txout (tx_id);
CREATE INDEX x_tx_txid ON tx (tx_id);
CREATE INDEX x_txout_value ON txout(txout_value);
CREATE INDEX x_blocks_id ON blocks(block_id);

CREATE VIEW tx_full AS SELECT blocks.time, tx.tx_hash, tx.tx_id, txout.address,
    txout.txout_value FROM txout LEFT JOIN tx ON (tx.tx_id = txout.tx_id) LEFT JOIN
     blocks ON (tx.block_id = blocks.block_id);
```

Features database

```
CREATE TABLE addresses
 (
    address         TEXT NOT NULL PRIMARY KEY,
    first_seen      INT,
    last_seen       INT,
    recv            INT,
    sent            INT,
    balance         INT,
    n_tx            INT,
    mining          REAL,
    gambling        REAL,
    exchanges       REAL,
    wallets         REAL,
    bitcointalk     REAL,
    bitcoinotc      REAL,
    freebies        REAL,
    donations       REAL,
    ota             BOOLEAN,
    old             BOOLEAN,
    new             BOOLEAN,
    empty           BOOLEAN,
    exhausted       BOOLEAN,
    recently_active BOOLEAN,
    zombie          BOOLEAN,
    scammer         BOOLEAN,
    disposable      BOOLEAN,
    miner           BOOLEAN,
    shareholder     BOOLEAN,
    casascius       BOOLEAN,
    fbi             BOOLEAN,
    silkroad        BOOLEAN,
```

```
    killer          BOOLEAN,
    malware         BOOLEAN,
    bitcointalk_user TEXT,
    bitcoinotc_user  TEXT,
    cluster_id      INT
  );

CREATE TABLE clusters
  (
    cluster_id      INT NOT NULL PRIMARY KEY,
    first_seen      INT,
    last_seen       INT,
    recv            INT,
    sent            INT,
    n_tx            INT,
    mining          REAL,
    gambling        REAL,
    exchanges       REAL,
    wallets         REAL,
    bitcointalk     REAL,
    bitcoinotc      REAL,
    freebies        REAL,
    donations       REAL,
    ota             REAL,
    old             REAL,
    new             REAL,
    empty           REAL,
    exhausted       REAL,
    recently_active REAL,
    zombie          REAL,
    scammer         REAL,
    disposable      REAL,
    miner           REAL,
```

```
    shareholder      REAL,
    casascius        REAL,
    fbi              REAL,
    silkroad         REAL,
    killer           REAL,
    malware          REAL,
    bitcointalk_user TEXT,
    bitcoinotc_user  TEXT,
    min_balance      INT,
    max_balance      INT,
    avg_balance      INT
 );

CREATE INDEX x_cluster_id
  ON addresses (cluster_id);

CREATE INDEX x_last_seen
  ON addresses (last_seen);
```

Trades database

```
-- Tables
CREATE TABLE trades(tid integer, currency text, amount real, price real,
date integer);

-- Indexes
CREATE UNIQUE INDEX trades_currency_tid_index on trades(currency,tid);
```

List of Acronyms

ASIC Application-specific integrated circuit – an integrated circuit customized for a particular use, rather than intended for general-purpose use

BTC Bitcoin

DB Database

ECDSA Elliptic Curve Digital Signature Algorithm (DSA) – a variation of DSA based on calculations of elliptical curves over finite space

FPGA Field-programmable gate array – an integrated circuit designed to be configured by a customer or a designer after manufacturing

GPU Graphics processing unit – a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the building of images in a frame buffer intended for output to a display

List of Figures