UNIVERSITY OF CALIFORNIA, SAN DIEGO

Using Crypto-currencies to Measure Financial Activities and Uncover Potential Identities of Actors Involved

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy

in

Computer Science

by

Yuxing Huang

Committee in charge:

Doctor Kirill Levchenko, Co-Chair
Professor Alex C. Snoeren, Co-Chair
Professor Terrence W. August
Professor Stefan Savage
Professor Geoffrey M. Voelker

2017

The Dissertation of Yuxing Huang is approved and is acceptable in quality and form for publication on microfilm and electronically:

Co-Chair

Co-Chair

University of California, San Diego

2017

DEDICATION

To my wife, Shuang, who is always there for me.

EPIGRAPH

Carpe diem. Seize the day.

Dead Poets Society

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am deeply grateful for my advisors, Prof. Alex C. Snoeren and Dr. Kirill Levchenko, for their understanding and guidance. They steered research in the right direction, especially when I was looking for the next steps after I published the paper on Bitcoin-mining botnets. They encouraged me to explore a diverse set of topics related to malware, Bitcoin, and economics, which led to a brief collaboration with Symantec and later a year-long research project with Google Maps. They taught me the importance of being rigorous in data analysis; in countless of meetings, they led by example, scrutinizing every data point to discover flaws that I had not even realized. Also, in countless practice talks, they showed me the importance of being crisp in describing research. I thank them for taking last-minute requests to edit manuscripts just before conference deadlines. I also thank them for being supportive when I went through a difficult period in my personal life.

I would also like to thank Prof. Terrence August, Stefan Savage, and Geoffrey M. Voelker for being my committee members and for their feedback on this dissertation. Prof. August, in particular, taught me how to think like an economist and showed me how an interdisciplinary thinking could discover more research problems.

In addition, I would like to thank Dr. Kenneth Yocum for his guidance during my first two years at graduate school and for sharing his expertise on data-center networking. Although our paper on data-center networking is not a part of this dissertation, it was my first project in graduate school, and he taught me important lessons on how to do research.

None of my research would have been possible without the support of system administrators, Cindy Moore and Brian Kantor. I especially thank Ms. Moore for setting up necessary systems for me on short notice. I also thank the both of them for helping me fix problems due to my experimental errors (some of which brought down our network

or raised security alarms), and for teaching me how to be a great system administrator myself.

Graduate school would have been less enjoyable had it not been for the support from my fellow graduate students. In particular, I thank Joe DeBlasio, who sat next to me for most of graduate school, for listening to my half-baked ideas and giving constructive feedback. I thank Louis Dekoven, who also sat next to me, for constantly sharing with me his industry insight on malware and botnets, which was helpful toward my ransomware project (not in this dissertation). I thank David Kohlbrenner, who shared the same office, for his deep insight on computer security whenever I needed the expertise. I thank Neha Chachra for being a supportive officemate whenever I had doubts about graduate school and research. I thank Vector Guo Li for helping me manage my computing devices, run experiments, and reverse-engineer malware samples even when I was away from San Diego. I thank Tianyin Xu for helping me with the academic job search process.

I also had the privilege of working with researchers outside of UCSD. Prof. Damon McCoy, in particular, contributed to many of my projects, and I feel deeply grateful for his guidance and support even though he was mostly away physically. I also thank Berkeley graduate students with whom I briefly shared an office — Paul Pearce and Frank Li — for providing helpful feedback to my work, for bouncing research ideas, and for helping me with academic job search. I thank Rebecca S. Portnoff, also a Berkeley graduate student, for giving me an invaluable opportunity to use Bitcoin to fight human trafficking.

Finally, this dissertation would not have been possible without my companion dog, Momo, and especially my wife, Shuang. No words are sufficient to describe their love and support, especially during the toughest moments of graduate school.

Chapter 2, in full, is a reprint of the material as it appears in the Proceedings of Network and Distributed System Security Symposium, 2014. Huang, Danny Yuxing;

Dharmdasani, Hitesh; Meiklejohn, Sarah; Dave, Vacha; Grier, Chris; McCoy, Damon; Savage, Stefan; Weaver, Nicholas; Snoeren, Alex C.; Levchenko, Kirill. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is currently being prepared for submission for publication of the material. Huang, Danny Yuxing; Levchenko, Kirill; Snoeren, Alex C. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, is a reprint of the material as it appears in the Proceedings of The Third Workshop on Bitcoin and Blockchain Research, 2016. Baqer, Khaled; Huang, Danny Yuxing; Weaver, Nicholas; McCoy, Damon. The dissertation author was an author of this paper.

Chapter 5, in part, is a reprint of the material as it appears in the Proceedings of the ACM SIGKDD Conference, 2017. Portnoff, Rebecca S.; Huang, Danny Yuxing; Doerfler, Periwinkle; Afroz, Sadia; McCoy, Damon. The dissertation author was an author of this paper.

VITA

2007–2011 Bachelor of Arts, Williams College, Massachusetts

2011–2017 Doctor of Philosophy, University of California, San Diego

PUBLICATIONS

Danny Yuxing Huang, Kenneth Yocum, Alex C. Snoeren. "High-Fidelity Switch Models for Software-Defined Network Emulation." *Proceedings of ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN)*. Hong Kong. August 2013.

Danny Yuxing Huang, Hitesh Dharmdasani, Sarah Meiklejohn, Vacha Dave, Chris Grier, Damon McCoy, Stefan Savage, Nicholas Weaver, Alex C. Snoeren, Kirill Levchenko. "Botcoin: Monetizing Stolen Cycles." *Proceedings of Network and Distributed System Security Symposium (NDSS)*. San Diego, CA. February 2014.

Kurt Thomas, Danny Yuxing Huang, David Wang, Elie Bursztein, Chris Grier, Thomas J. Holt, Christopher Kruegel, Damon McCoy, Stefan Savage, Giovanni Vigna. "Framing Dependencies Introduced by Underground Commoditization." *Proceedings of Workshop on the Economics of Information Security (WEIS)*. The Netherlands. June 2015.

Khaled Baqer, Danny Yuxing Huang, Nicholas Weaver, Damon McCoy. "Stressing Out: Bitcoin Stress Testing." *Proceedings of the Third Workshop on Bitcoin and Blockchain Research*. Barbados. February 2016.

Danny Yuxing Huang, Doug Grundman, Kurt Thomas, Abhishek Kumar, Elie Bursztein, Kirill Levchenko, Alex C. Snoeren. "Pinning Down Abuse on Google Maps." *Proceedings of International World Wide Web Conference (WWW)*. Perth, Australia. April, 2017.

Rebecca S. Portnoff, Danny Yuxing Huang, Periwinkle Doerfler, Sadia Afroz, Damon McCoy. "Backpage and Bitcoin: Uncovering Human Traffickers." *Proceedings of ACM SIGKDD Conference*. Halifax, Nova Scotia. August 2017.

ABSTRACT OF THE DISSERTATION

Using Crypto-currencies to Measure Financial Activities and Uncover Potential Identities of Actors Involved

by

Yuxing Huang

Doctor of Philosophy in Computer Science

University of California, San Diego, 2017

Doctor Kirill Levchenko, Co-Chair
Professor Alex C. Snoeren, Co-Chair

Bitcoin is a digital currency that has recently gathered significant interest. From e-commerce sites to darkweb marketplaces, merchants accept Bitcoin as a form of payment. Every day, millions of dollars are transacted across Bitcoin's payment network. The value of a single bitcoin has increased from $500 to $3,000 in a one-year period since July 2016.

A part of the interest may stem from the *decentralized* design of Bitcoin. A peer-to-peer network collectively generates new coins and maintains the distributed

transaction ledger, also known as the *blockchain*. The blockchain records transactions between public keys, rather than between real-world identities. This detachment from real-world identities makes it hard to measure financial activities and identify actors on the network, such as four cases that we study: (i) botnets stealing computational cycles, (ii) speculatively investing in digital currencies, (iii) delaying the processing of Bitcoin payments, and (iv) purchasing ads with illegal contents.

Despite this challenge, the decentralized design of Bitcoin and similar digital currencies offers public information on every transaction and the associated identities. This dissertation demonstrates that, using the four cases as examples, we can leverage this public information to analyze financial activities — e.g. measuring the cost and revenue — and to potentially uncover the identities of the actors involved.

In particular, we can measure the revenue and cost for Cases (i) through (iii). For (i), we show that botnets made a modest income of $118,000 between 2012 and 2013, but for some botnets we estimate the cost to victims to be more than twice the botnets' revenue. For (ii), we develop a new way to estimate the profitability of investing in digital currency markets. By simulating multiple investment strategies, we show the drastic variations in profitability and thus the extreme risks associated with digital currency investment. For (iii), we show that an adversary delayed Bitcoin transaction processing time from 0.33 to 2.67 hours, at a modest cost of $4,900 per day. Furthermore, we can uncover the potential identities of the actors involved. For (i), we identify 10 distinct botnet operations. For (iv), we identify ads paid for by potentially the same criminals.

Introduction

Satoshi Nakamoto developed Bitcoin, an "electronic cash system", in 2009 [49]. Since then, the popularity of Bitcoin has surged. Similar to Visa and PayPal, users can pay for goods and services online with Bitcoin. Many online merchants accept Bitcoin as one of the payment options alongside Visa and PayPal. Examples include Overstock [54], which is an online retailer like Amazon, and Namecheap [50], which is sells domain names online. Furthermore, Bitcoin is often accepted where mainstream payment options are unavailable — for instance, on the darkweb. In fact, many underground marketplaces, hosted on the Tor network, accept Bitcoin as the sole payment option. These marketplaces typically sell contraband goods such as drugs and weapons. Whereas transactions via wire transfers or credit cards are subject to government regulations and monitoring, Bitcoin transactions are partially anonymous and difficult to trace as a result of its cryptographic properties (more details in Chapter 1). Because of the cryptographic properties, Bitcoin is also known as a *crypto-currency*.

In addition to being a payment system, bitcoins themselves are tradable assets. One can buy or sell bitcoins at Bitcoin *exchanges*, just like one can buy and sell gold, US dollars, or oil.[1] The US Dollar to Bitcoin exchange rate soared above $1,000 for the first time in November 2013, and in June 2017, it almost reached the $3,000 mark [17]. Furthermore, there is a significant flow of cash in Bitcoin markets. According to estimates by blockchain.info, a website that tracks all trades against Bitcoin, at least $100 million

[1] By convention, we refer to "Bitcoin" as the digital currency, and "bitcoin" as unit of the currency.

US Dollars were traded against bitcoin every day in July 2017 [18]. In comparison, Alphabet Inc. Class C Capital Stock (GOOG) has a daily volume of less than $10 million[2] in the same period [53].

Bitcoin is created through an energy-intensive process. Whereas only the Federal Reserve can issue new US Dollar bills, anyone, known as a *miner*, can generate new bitcoins in a process known as *mining*, where computational devices, such as CPUs, GPUs, or even dedicated hardware, are used to solve special cryptographic puzzles (details in Chapter 1). Finding the solution requires significant electricity to power the devices, but as a reward the miners create new bitcoins for themselves, which they can use in commerce or sell at exchanges. Based on some estimates, Bitcoin mining is a billion dollar global industry [34]. This industry includes manufacturers that produce dedicated bitcoin-mining hardware (i.e ASICs). The industry also includes miners who set up mining farms that are filled with such hardware [63].

In short, there is significant interest in Bitcoin: using it for commerce, trading bitcoins, and mining bitcoins. Part of the interest may lie in Bitcoin's decentralized design. Unlike Visa or the US Dollar, Bitcoin was designed to not depend upon any centralized entities for payment processing or money supply. Instead, a peer-to-peer network of volunteer devices collectively maintains the transaction ledger, also known as the *blockchain*, and generates new units of the currency. In contrast to Visa/PayPal's ability to decline transactions, for instance, Bitcoin is designed to prevent transactions from being disrupted. In theory, no single entity can decline transactions. Also, whereas central banks can influence the monetary policies in a country, such as tightening/loosening the money supply to regulate inflation, new bitcoins are generated at a predetermined rate through mining. As such, the decentralized nature poses significant barriers for any single parties to influence Bitcoin.

[2]Unless otherwise stated, all dollar values in this dissertation refers to US Dollars.

In fact, Bitcoin is not the only decentralized crypto-currency. Thousands of similar crypto-currencies, developed after Bitcoin, share Bitcoin's open-source code base. Together, these crypto-currencies are involved in a wide range of use cases. Below are four examples that we study in this dissertation.

- Use cases that involve crypto-currencies' decentralized money supply (i.e. mining):
  - **Case (i):** During 2012–2013, botnets mined bitcoins and other crypto-currencies, converting electricity from compromised computers into money.
  - **Case (ii):** Speculative investors mine and trade crypto-currencies (typically non-Bitcoin).
- Use cases that involve crypto-currencies' decentralized payment processing:
  - **Case (iii):** Miscreants sent spam transactions in Bitcoin and slowed down payment processing, presumably to disrupt the Bitcoin payment network and highlight Bitcoin's fragility.
  - **Case (iv):** Human traffickers paid bitcoins for ads on a popular prostitution portal, where Bitcoin was the only form of payment accepted.

All four cases have significant implications in both legal and financial terms. In particular, Cases (i) and (iv) involve illegal activities, and Cases (i) through (iii) are associated with financial gains/losses. As such, one might be motivated to measure the activities behind the four cases and potentially identify the individuals involved. However, there are challenges. The identities involved in these transactions are not necessarily tied to anyone's personally identifiable information; rather, they are represented as public keys. Linking these public keys to real-world identities is thus not a trivial task. In

other words, the partial anonymity of crypto-currencies makes it difficult for us to track activities and identify the actors involved.

The decentralized design of Bitcoin and other crypto-currencies can help us tackle this challenge, as crypto-currencies provide a wealth of public information: transaction history, the identities (albeit partially anonymous) of actors in each transaction, and relationship among these actors (e.g. who sent bitcoins to whom). This dissertation demonstrates that, using the four cases above as examples, we can leverage this public information to analyze financial activities — e.g. measuring the revenue and cost — and to potentially uncover the identities of actors involved.

- **Measuring revenue and cost:** Using the public transaction history, we answer the following questions:
  - Case (i) / Chapter 2: How botnets mined bitcoins, how much money they made, and what was the cost to infected victims.
  - Case (ii) / Chapter 3: What are some of the investment strategies in crypto-currencies, how to estimate the revenue and cost, and what is the potential profitability behind these strategies across different crypto-currencies.
  - Case (iii) / Chapter 4: How to identify disruptive attempts to delay transaction processing across the Bitcoin network, what is the cost associated with launching these disruptive activities, and what is the time and financial loss related to such disruptions.
- **Potentially identifying actors:** The use of public keys in crypto-currency transactions can, to some extent, hide real-world identities, but we can potentially de-anonymize some of these identities by using the public transaction history and by inferring the relationships among the actors involved. In particular, our analysis answers the following questions:

- Case (i) / Chapter 2: How many botnet operations were involved, and what botnet/criminal organizations they were associated with.
- Case (iv) / Chapter 5: How to link ads back to the payers, and which ads were potentially paid for by the same actors. The answer may help law enforcement agencies identify human traffickers.

In summary, this dissertation introduces four use cases of crypto-currencies. For each case, the dissertation shows how we can leverage the public information of crypto-currencies, such as the transaction history and the identities of actors in transactions, to measure financial activities and potentially identify the actors involved.

Understanding such activities and the actors involved is important. Some of the actors are criminals (e.g. botnet and human trafficking), while some activities may cause disruption to typical commercial activities (e.g. Bitcoin spam transactions). Our analysis offers insight for future researchers to potentially stop these malicious behaviors. Furthermore, some activities are associated with potentially significant financial gains or losses (e.g. speculative investment in crypto-currencies). Our analysis can inform investors of the risks involved before they enter the market.

Chapter 1

Background

The goal of this dissertation is to show how we can leverage aspects of Bitcoin's structure (as well as that of similar crypto-currencies) to measure financial activities. To this end, we take advantage of various features of crypto-currencies. Primarily using Bitcoin as an example, we explain the decentralized features of crypto-currencies, and how these features help us achieve our goal. In particular, Section 1.1 provides an overview of the key concepts in Bitcoin. Then we focus on the technical details of how Bitcoin works as a decentralized payment system, from transaction processing (Section 1.2) to money supply (Section 1.3). Finally, we describe the conversion between Bitcoin and fiat currencies, such as US Dollar, in Section 1.4. For a complete background on Bitcoin, we refer to the original Bitcoin paper by Satoshi Nakamoto [49] and Princeton's Bitcoin textbook [52].

1.1 Overview of How Bitcoin Works

Bitcoin is a decentralized digital currency. Many merchants accept Bitcoin as a form of payment, much like Visa or PayPal. However, unlike Visa or PayPal, there is no central authority that manages Bitcoin, In this section, we offer an overview of the key concepts of Bitcoin — in particular, how Bitcoin processes transactions without a central ledger, and how Bitcoin generates new units of the currency without a central entity like

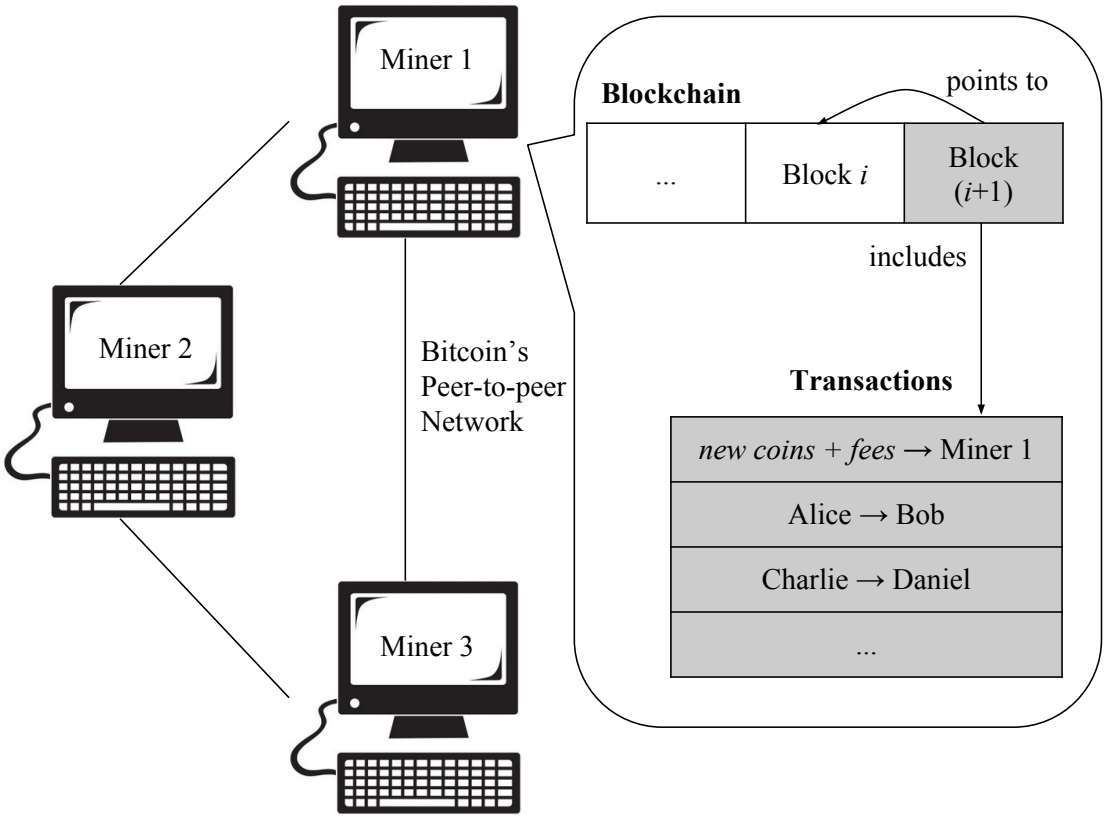


Figure 1.1. Overview of Bitcoin blockchain and transactions

the Federal Reserve.

Decentralized transaction processing: A peer-to-peer network collectively processes Bitcoin *transactions* — payments from one party to another. The network keeps track of these transactions across a distributed ledger, known as the *blockchain*. **Figure 1.1** shows an example of a Bitcoin network, which consists of three interconnected nodes. The human operator behind each of these nodes is known as *miners*. Each miner runs the Bitcoin *client software* on the device. A new miner can join the network as long as it runs the client software. All instances of this software have identical copies of the blockchain on disk. The figure shows the blockchain on one of the clients. The blockchain is a singly linked list of *blocks*, each of which consists of *transactions*. Blocks

are chronologically ordered. Block $(i+1)$ is created later than Block i. Whenever a new block is created, all miners in the network are notified. The block at the end (i.e. with the most recent timestamp) of the blockchain linked list is also known as the *latest* block.

Transactions are essentially records of payments. Suppose Alice is to send a unit of Bitcoin to Bob. First, Bob tells Alice his *wallet address* at which he expects to receive Alice's payment. Alice then creates a transaction which moves a bitcoin from Alice's wallet address into Bob's wallet address, along with a *transaction fee*. Note that wallet addresses are simply public keys that are not necessarily tied to any personally identifiable information of individuals involved in transactions. Furthermore, anyone can create an unlimited number of independent wallet addresses. In case one of the wallet addresses is tied back to identities in the real-world, the wallet address can always be discarded, and a fresh wallet address can be created.

After creating the transaction, Alice broadcasts it over the peer-to-peer network. At this point, the transaction is not in the blockchain yet. It merely exists in the temporary holding area in each client's memory, also known as the *mempool*.

Currently, suppose the last block in the blockchain is Block B_i (as illustrated in **Figure 1.1**). Per Bitcoin's protocol, a new block is expected to be created every ten minutes. Suppose the new block, B_{i+1}, includes Alice's transaction, along with other transactions (e.g. Charlie sending bitcoins to Daniel) that were previously in the mempool. The new block is then appended to the blockchain, pointing to the previous block B_i. At this point, Alice's transaction is in the blockchain, stored on the disks of all the clients in the peer-to-peer network. Alice's transaction is said to be *confirmed*. Due to cryptographic properties that we will explain in Section 1.3, the transaction can no longer be modified or rolled back. If Alice is paying Bob for some service, Bob, knowing that Alice's transaction is confirmed, is likely to perform the service.[1]

[1]Note that there are exceptions. As we will discuss in Chapter 5, Bob might ship the product before

Any new transactions, likewise, will be first broadcast over the network, stored in the mempool of individual miners, and added to block B_{i+2}, which will be appended to the blockchain after B_{i+1}. We will discuss how Bitcoin processes transactions in detail in Section 1.2.

Decentralized money supply: The peer-to-peer network is also responsible for collectively generating new units of Bitcoin. In a process known as *mining*, every miner attempts to find a *nonce* value, N, such that $H = \text{SHA-256}(\text{SHA-256}(N, (\text{other values}))) \leq T$, where T is known as the *target* value, which the Bitcoin protocol automatically sets.

Using the example in **Figure 1.1**, suppose that the currently latest block is Block i, and all three miners are mining. Suppose Miner 1 manages to find an N such that $H \leq T$. In other words, Miner 1 successfully mines a new block, Block $(i+1)$, which includes some of the transactions in Miner 1's mempool (including Alice's transaction to Bob and Charlie's transaction to Daniel). Miner 1 verifies these constituent transactions and broadcasts the new block. All other miners stop their attempts to find N. Every miner, including Miner 1, appends Block $(i+1)$ to its respective blockchain. In this way, every miner has exactly the same blocks and same transactions in the blockchain. The blockchain, as a distributed ledger, reaches a consistent state across the network.

Mining is a computationally intensive process, as SHA-256 essentially returns a random value, and every miner has to iterate through different N values with brute force before he finally finds the right N (or before another miner does). To incentivize this effort, every new block created also includes an extra transaction that pays whomever that has found the right N. Before they start mining, every miner has to set up his wallet address in advance. When Miner 1 finds the right N, Block $(i+1)$ will include a transaction that rewards Miner 1 (along with Alice's transaction and Charlie's transaction). In this

Alice's tranasaction is confirmed. In general, it is entirely up to Bob when to ship. This decision is independent of Bitcoin's protocol.

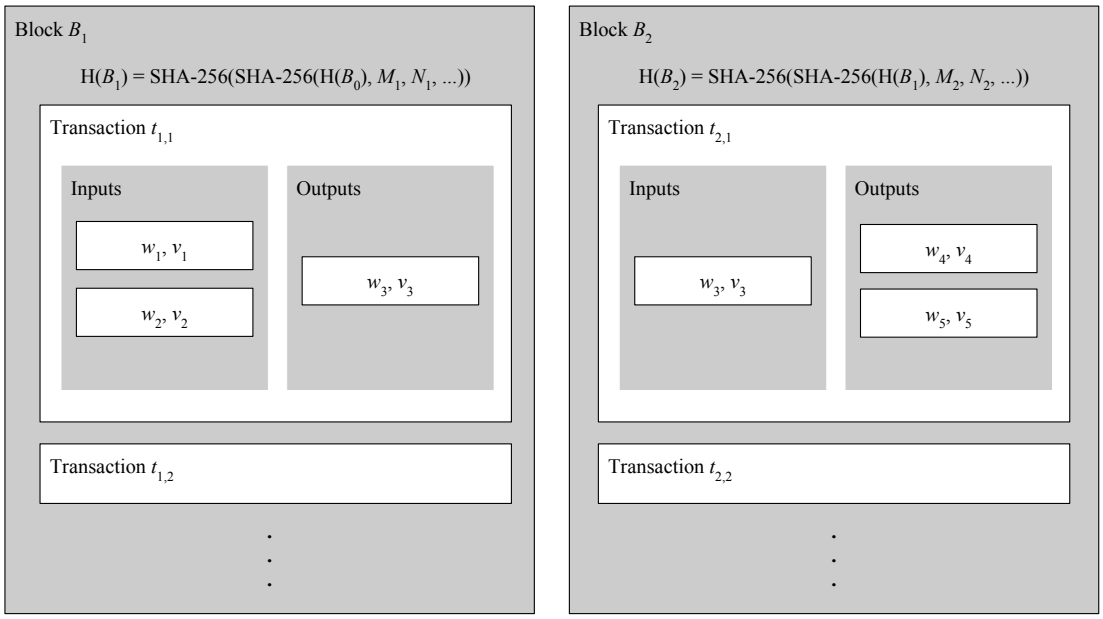


Figure 1.2. Details of data in blockchain and transactions

transaction, new coins are created and, along with the fees collected from Block $(i+1)$'s transactions, moved into Miner 1's wallet address, as illustrated in **Figure 1.1**. In this way, Miner 1 is paid for his energy input in the mining process. At the same time, because Miner 1 has already mined Block $(i+1)$, all other miners stop their current search for the right N, and their effort is basically wasted. We will discuss how to mine bitcoins in detail in Section 1.3.

1.2 Decentralized Transaction Processing

In contrast to PayPal or Visa, which has a logically centralized entity that processes all transactions, Bitcoin processes transactions in a peer-to-peer network of volunteer devices. In this section, we will describe how transactions are processed, from creating transactions, to distributing them across the network, to finally securing them in the global ledger. We will also explain how we can follow transactions and attempt to de-anonymize the identities involved in the transactions.

1.2.1 Transaction

Each transaction is essentially a transfer of bitcoin ownership from one wallet address to another. Take transaction $t_{2,1}$ (in Block B_2) in **Figure 1.2** as an example, where one wallet address is sending bitcoins to two wallet addresses (e.g. Alice paying Bob and Charlie). The transaction consists of the following components:

- The *output* wallet addresses, which are associated with the payment receivers. In this example, there are two output wallet addresses, one for each receiver: wallet address w_4, which is to receive an *output value* of v_4 bitcoins, and wallet address w_5, which is to receive v_5 bitcoins.

- The *input* wallet addresses, which are associated with the payment senders. In this example, there is only one input wallet address w_3, which is willing to pay an *input value* of v_3. Each input is actually the output of a previously confirmed transaction. Thus, the v_3 in transaction $t_{2,1}$ is in fact the output of $t_{1,1}$, where w_3 receives v_3 bitcoins.

- An optional transaction fee f. In this example, $f = v_3 - v_4 - v_5$. As we will discuss in Section 1.2.2, each block can include a limited number of transactions. Transactions that pay higher fees are typically confirmed in the next block with a higher probability.

Each transaction is signed with the private key of the input wallet address (which is the corresponding public key) if there is only one input wallet address. In the example, the sender's wallet address is w_3, which is equivalently the sender's public key. The sender uses the corresponding private key to sign $t_{2,1}$. If there are multiple input wallet addresses (i.e. multiple public keys), such as in $t_{1,1}$, then the private key of each of the input wallet addresses (i.e. w_1 and w_2) needs to sign the transaction.

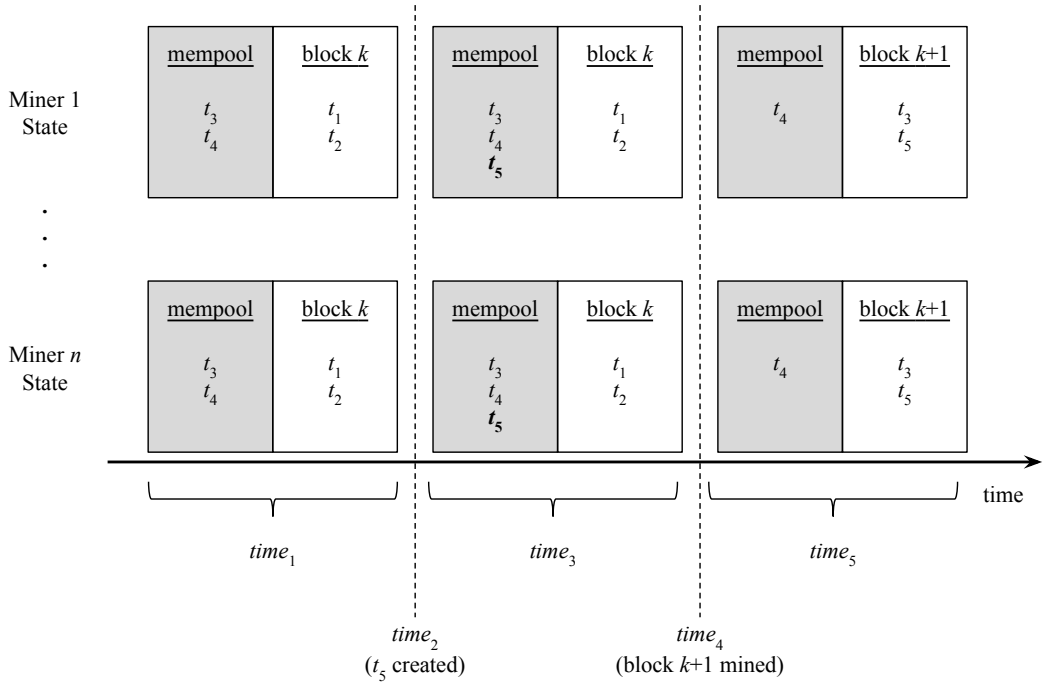


Figure 1.3. Bitcoin mempool

For $t_{2,1}$, the signature proves that v_3 bitcoins were under w_3's control, and that w_3 intends to hand over control to w_4 and w_5. In this way, w_4 and w_5 can send the bitcoins to any wallet addresses of their choice.

A transaction must fully transfer the ownership of a coin. Suppose Alice has 1 bitcoin and she is about to send 0.8 bitcoins to Bob. She would need to create a transaction that takes the 1 bitcoin from Alice as the input, and sends 0.8 bitcoin to Bob and 0.2 bitcoins back to Alice. An example of such a transaction has w_3 in the input, and both w_3 and w_4 in the output, where w_4 is the intended receiver.

1.2.2 Mempool

To send bitcoins, a sender must first create a transaction and broadcast it to Bitcoin's peer-to-peer network. At some point, a new block is mined that includes this transaction. The transaction is now a part of the blockchain and finally confirmed.

To illustrate this process, we refer to **Figure 1.3**. It depicts n Bitcoin clients, their mempool state (gray box), and the latest block in the blockchain (white box). Initially, at $time_1$, the latest block in the blockchain is Block k, which includes two confirmed transactions t_1 and t_2. In the mempool of each miner are two transactions t_3 and t_4, waiting to be confirmed at some point.

At $time_2$, a new transaction t_5 is created and immediately broadcast to the network. Within a few seconds, at $time_3$, t_5 appears in the mempool of every miner. There are no changes in the blockchain.

Suppose a new Block $k+1$ is mined at $time_4$. The miner which mined this block has decided to include t_3 and t_5 in the new block. Here, t_4 is not included. One possible reason is that t_4 might be paying a smaller transaction fee, which we will discuss later in the section. As a result, at $time_5$, only t_4 remains in the mempool, while both t_3 and t_5 have been removed from the mempool and included into the latest Block $k+1$. At this point, neither t_3 nor t_5, being in the blockchain, can be modified or reversed; Section 1.3 will discuss the cryptographic principles behind mining that leads to the integrity of the transactions.

Each miner makes independent decisions on what mempool transactions to be included in the next block. Typically, if a miner runs the default Bitcoin client, it makes decisions based on a transaction's fees or its *priority*, P, computed as:

$$P = \frac{1}{S}\sum_{i=0}^{n}(v_i \times \Delta t_i) \tag{1.1}$$

where S is the transaction's size in KB, n is the number of inputs to the transaction, v_i is the value of bitcoins in input i, and Δt_i is the *age* of input i. The age of an input is the time difference between the previous output and input i itself. Using **Figure 1.2** as an example, suppose $t_{2,1}$ is created now; it is still in the mempool and not confirmed yet.

Then Δt for the input of $t_{2,1}$ is the time since $t_{1,1}$ was confirmed into block B_1, since the input of $t_{2,1}$ is the output of $t_{1,1}$.

Each Bitcoin miner needs to make a decision on which transaction to include, because each new block can hold at most 1 MB of transactions. According to one study, the average transaction size is around 600 bytes in October 2015.[2] Thus, a block typically can hold thousands of transactions. Even so, at a high transaction volume, not every transaction can be included into the next block. Some would have to wait for multiple blocks to pass before being confirmed. A typical Bitcoin miner, using the default Bitcoin client, prioritizes transactions in the following ways (although the Bitcoin protocol requires none of the following):

- **Case (i):** Each block has a 50 KB section for transactions that do not need to pay any fees. Such transactions must satisfy the following conditions:
  - The transaction size, S, is less than 1 KB.
  - Each of the outputs is at least 0.01 bitcoins.
  - P is higher than other Case (i) transactions. In other words, a typical miner sorts all Case (i) transactions in descending order of P. The miner picks the top transactions until the 50 KB section of the next block is full. The other transactions remain in the mempool.
- **Case (ii):** Otherwise, a transaction has to pay fees. Priority is based on the per-KB fee (i.e. f/S). In other words, a typical miner sorts all Case (ii) transactions in descending order of f/S. The miner picks the top transactions until the next block is full. The other transactions remain in the mempool.

[2] https://tradeblock.com/blog/analysis-of-bitcoin-transaction-size-trends

Prioritizing transactions based on P or the per-KB fee, as illustrated above, is a commonly accepted practice that the default Bitcoin protocol follows. There is no guarantee, however, that a miner will adhere to this practice. We have seen evidence of miners who included no-fee transactions into their blocks, even though the transaction size was more than 1 KB. We have also seen miners who simply removed transactions from the mempool possibly due to low fees.[3]

Until a transaction is included in a block, it remains unconfirmed and is subject to change — for instance, through double-spending attacks. Suppose Alice is to send a bitcoin to Bob. She creates Transaction 1, where the output is Bob's wallet address, and the input wallet address has a bitcoin that Adam had previously sent to Alice (i.e. previous output). While this transaction is in the mempool, Alice creates Transaction 2, which has exactly the same input as Transaction 1. The only difference is that Transaction 2's output is Alice's wallet address. If Transaction 2 is confirmed into the blockchain before Transaction 1 (e.g. by paying a higher fee in Transaction 1), then no miner would include Transaction 1 into the blockchain, because its input, i.e. Adam's coin to Alice, is already *spent* in Transaction 2. Transaction 1 is effectively nullified. Because of the risk of double-spending, if Alice is paying Bob for a service, Bob is likely to wait until the transaction is confirmed before performing the service. There are exceptions, such as what we will discuss in Chapter 5.

The wait time for confirmation to occur can vary from several minutes to hours. The exact timing depends on two factors:

- **Transaction backlog.** Per Bitcoin's protocol, a block can hold at most 1 MB of transactions. At a high transaction volume, miners will be unable to include all transactions in the mempool into the next block. Thus, transactions are prioritized

[3]This observation is based on anecdotal evidence collected by the author, rather than scientific publications.

based on P (Case (i)) or the per-KB fee (Case (ii)).

- **New blocks.** On average, a new block is mined every ten minutes. If a transaction enters the mempool shortly after a new block is mined, then the expected wait time for confirmation will be about ten minutes if the new block includes the transaction. Conversely, if a transaction enters the mempool and a newly mined block includes it shortly afterwards, then the wait time will be significantly shorter. The expected wait time for a transaction is therefore five minutes in the absence of transaction backlog.

Chapters 4 and 5 extensively use the mempool data in the analysis. While transactions in the mempool are unconfirmed, they provide useful insight on when transactions are created and confirmed. Other work in this space includes using the mempool data to measure confirmation delays and transaction fees in the Bitcoin network [48]. The research community recognizes that confirmation delay is a significant problem with regard to Bitcoin's stability and scalability, as outlined in the a recent study [19]. In one specific example, Becker et al. [12] discuss a hypothetical scenario where opponents of Bitcoin collectively sent spam transactions to the network, using up valuable space in the blockchain and driving normal Bitcoin users away.

1.2.3 Clustering Bitcoin Transactions

Even though identities in Bitcoin are represented as wallet addresses, which can be easily created or changed, Bitcoin is not fully anonymous. We can cluster wallet addresses for which the same entity has control over the private keys. If we can link one of the wallet addresses to a real-world identity, then all the wallet addresses in the cluster are likely to belong to the identity [47].

Specifically, here is how the technique works. First, we examine the inputs of

a given transaction. Recall that a transaction with multiple inputs needs to be signed with every private key that is associated with the input wallet addresses. In the absence of *CoinJoin* — a technique that allows multiple input private keys to sign the same transaction without revealing the private keys — we assume that the entity which creates the transaction must have access to all the private keys. If having the private key implies ownership, then it follows that all the input wallet addresses belong to the same entity. These input wallet addresses are known as *co-spent* addresses. In **Figure 1.2**, w_1 and w_2 are co-spent addresses in $t_{1,1}$. By iterating through all Bitcoin transactions and discovering co-spent addresses, we construct clusters of wallet addresses that belong to the same entities.[4]

At this point, however, we still do not know the real-world identities of these clusters. To this end, we discover at least one wallet address that we can link to real-world identities. For instance, let us suppose we want to find all addresses that belong to Eve. First, we ask Eve to provide us with a wallet address of hers to which we send our payments. Eve can create a new wallet address, w_1, just for us, but it is possible that this address may be co-spent with other addresses of hers, $\{w_2, w_3, \ldots, w_n\}$ — for example, to aggregate bitcoins that she has collected from us as well as other sources. In this way, we can infer that all the w_i's for $i = 1, \ldots, n$ belong to Eve. By repeatedly asking Eve to generate a new wallet address to which we send bitcoins, we can discover a subset of Eve's wallet addresses to which we did not send bitcoins. This technique also applies if we ask Eve to send us bitcoins. Each new wallet address from which she sends us bitcoins can potentially be co-spent with other addresses of hers. In this way, we can

[4]CoinJoin breaks this technique. Basically, CoinJoin obfuscates the bitcoin movement trail. Suppose there are n wallet addresses with a total of v bitcoins in balance. Together, the owners of these n wallet addresses can construct a CoinJoin transaction that takes these n wallet addresses as the input and sends the v bitcoins to n different wallets addresses. The owner of each of the original n addresses does not need to have the private keys of the other addresses to sign the transaction. This type of transaction invalidates our assumption that owners of co-spent wallet addresses have access to all the private keys. As a result, our clustering technique will no longer valid [19].

identify wallet addresses that belong to organizations, such as mining pools (Section 1.3) or exchanges (Section 1.4).

Chapters 2 and 5 use the clustering technique above to de-anonymize organizations (such as mining pools and exchanges). As an aside, a recent study examined the graph structure of Bitcoin's transaction graph for de-anonymization [61]. We do not use this technique in the dissertation.

1.3 Decentralized Money Supply

For a fiat currency such as the US Dollar, a logically centralized entity such as the US Federal Reserve is responsible for issuing the currency. This central entity has the power to loosen or tighten the money supply, thereby adjusting inflation and regulating the country's economy. In contrast, no such central body exists for Bitcoin. A peer-to-peer network of volunteer devices collectively supplies new units of Bitcoin at a pre-determined rate according to Bitcoin's protocol, in a process known as *mining*. In this section, we explain how mining generates new bitcoins and secures the blockchain.

1.3.1 How Mining Works

Anyone can participate in the creation of new bitcoins through the mining process. Using computational devices such as CPUs, GPUs, or dedicated hardware, miners attempt to find a special type of hash collision with brute force. The miner who first finds the hash collision is rewarded with new bitcoins, along with the fees from confirmed transactions. This process is computationally intensive and it consumes a significant amount of electricity. Effectively, mining converts energy into bitcoins.

To illustrate the mining process, we use **Figure 1.2** as an example. Here, we assume that Block B_1 is the latest block, and B_2 has not been mined yet. In other words, transactions $t_{2,1}, t_{2,2}, \ldots$ exist only in the mempool at this stage.

1. A miner downloads and executes the Bitcoin *client*, a piece of software which connects the miner to Bitcoin's peer-to-peer network.

2. The miner specifies a wallet address, W, at the client software. W is used to receive mining reward later.

3. The client initializes an empty set M_2.

4. For transactions in the mempool that pay fees, the client adds them to M_2, prioritized by their per-KB fees (i.e. Case (i) in Section 1.2.2). For transactions that do not need to pay fees, the client adds those to M based on the P values (i.e. Case (ii) in Section 1.2.2). By default, the Bitcoin client follows the practice as outlined in Section 1.2.2, but a miner has the liberty to modify the client and add any transactions to M.

5. Using the CPU, graphics card, or even dedicated mining hardware (e.g. ASICs), the client computes the SHA-256 function twice on a concatenated string, which includes (but not limited to):

   - N_2, an 32-bit nonce.
   - The hash of a subset of transactions in M_2.
   - The hash of the latest block, B_1.

6. Let $H(B_2)$ be the result of the double SHA-256 hash function. The client repeatedly computes H, and one of the following might happen:

   - $H > T$, where T is the global target value. In this case, the client repeats Steps 3, 4, 5 and 6 using a different N_2 value. M_2 may be different at every iteration, as new transactions are constantly added to the mempool.

- $H \leq T$. In this case, the client is considered to have found the special hash collision. A new block, B_2 is created, with H being the block hash. All transactions in M_2 are removed from the mempool and included in Block B_2. This new block also includes a special *coinbase* transaction that pays bitcoins to W. The coinbase transaction does not have any inputs, and W is the sole output. Effectively, bitcoins are created out of thin air to reward the miner's effort. The client announces Block B_2 to the entire network, and B_2 becomes the latest block.
- Another client announces that it has mined B_{i+1}. This client stops computing H, removes from the mempool all transactions in B_{i+1}, and goes back to Step 3. Effectively, all computations so far have gone to waste.

The algorithm outlined above is also known as the *proof-of-work* algorithm (PoW). Finding the right N so that $H \leq T$ requires brute force, but it is trivial to verify N results in $H \leq T$. The implication of PoW is that it is extremely difficult for transactions, once confirmed into blocks, to be changed or reversed. Recall that computing H depends on all the transactions in M. Any changes in the transactions will result in a different H value for which $H \leq T$ may no longer be true.

The random nature of finding hash collisions makes mining competitive. Every miner's chance of discovering a valid block is proportional to both the number of SHA-256 calculations (the *hash rate*) it can perform per second (usually measured in millions of hashes per second (MH/s), billions (GH/s), or trillions (TH/s)) and the hash rate of the Bitcoin network as a whole. An average desktop PC can perform anywhere from 2 to 10 MH/s, while a dedicated ASIC mining system can reach 13 TH/s or more [14]. On November 30, 2013, the Bitcoin network's hash rate was approximately 6,000 TH/s, which implies that a single 10-MH/s PC would have expected to receive less than

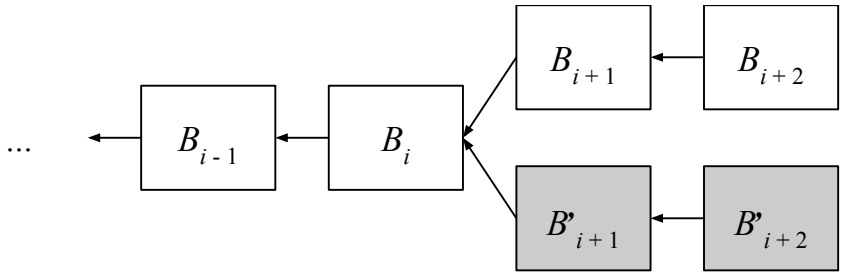


Figure 1.4. Forking of the blockchain.

0.0000002% of all Bitcoins produced globally during the period it mined [16].

As the speed of hardware increases, finding the right N such that $H \leq T$ takes a shorter time. To counter this effect, for every 1,204 mined blocks, the Bitcoin network automatically adjusts the *difficulty* D, defined as $D = 2^{32}T^{-1}$, such that on average a block is mined every ten minutes. In other words, as the network mines blocks faster, D is increased, thus lowering T and subsequently the probability of mining a block. This feedback mechanism ensures that the rate of block generation and the rate of bitcoin rewards remain relatively constant, regardless of the total computational power in the network.

In general, there are two benefits associated with mining:

- **Miners are rewarded.** A miner is rewarded for mining a block. In the example of **Figure 1.1**, Miner 1, which has successfully mined block B_{i+1} receives a reward. In the reward transaction, new coins are created and transferred to Miner 1's wallet address, along with fees collected from transactions confirmed in B_{i+1}. As of August 2017, a miner receives 12.5 bitcoins per block, which can be sold at $50,000.

- **Difficult to reverse transactions.** Once confirmed in the blockchain, it is extremely difficult to reverse a transaction. For example, suppose the latest block in the blockchain is B_{i+1}, which points to the previous block B_i, as shown in

Figure 1.4. Suppose B_{i+1} contains a transaction t where some wallets w_1 sends v bitcoins to w_2. Suppose the owner of w_1 decides to roll back t, such that the v bitcoins will remain in w_1. The owner would need to do the following:

1. Mine a new block B'_{i+1}, which points to B_i.
2. At this moment, two blocks, B_{i+1} and B'_{i+1}, point to B_i. The blockchain is now *forked* (i.e. bifurcates). Subsequent new blocks will point to B_{i+1} and B'_{i+1} with 50% probability respectively.
3. To make sure that his fork of the chain will be recognized by the Bitcoin network, the owner of w_1 (i.e. the miner of B_{i+1}), must ensure that his chain is at least a block longer, as the Bitcoin network, by default, accepts the longest chain. So he mines block B'_{i+2}, which points to B'_{i+1} — before the rest of the network mines B_{i+2}, which points to B_{i+1}. Basically, he is competing against the computational power of the entire network, which is typically much higher than his own computational power. As such, it is unlikely that his fork will be longer, and that he can reverse t.

Note that forking this blockchain in this manner is just one way that an adversarial miner can potentially tamper with confirmed transactions. There are other techniques, as illustrated in four recent studies [28, 46, 42, 29], such that an appropriately powerful adversary can game the mining process.

1.3.2 Pooled Mining

At the difficulty level in August 2017, a desktop PC mining at 10 MH/s can expect to mine more than 100,000 centuries before finding a winning block. Even with top-of-the-line dedicated mining hardware capable of 13 TH/s, or one block every 9 years, at this difficulty level, mining becomes a lottery. To overcome the uncertainty of

such *solo mining*, a miner can join a *mining pool*, which combines the mining power of a large number of individual miners and pays a small amount for each unit of work performed toward mining a block. Essentially, by parallelizing the search for a winning block, a pool can be thought of as buying multiple lottery tickets for any given drawing. With a typical mining pool, each miner is paid in proportion to his hashing power, but the income is significantly steadier due to the decrease in variance in the expected time required (for some member of the pool) to successfully mine a block.

In pooled mining, the pool server runs the Bitcoin client and manages the mempool. A miner joins the pool by first creating an account on the pool's website. Instead of directly connecting to the Bitcoin client, the miner, known as the *worker*, communicates with the pool server, using a simple HTTP-based RPC protocol, known as *getwork*. Similar to solo mining, the miner needs to repeatedly compute SHA-256 twice on some concatenated string until finding the resultant hash is no larger than the target value. The main difference is the following.

- The pool provides the miner with the hash of transactions in M. Effectively, the miner has no knowledge of individual transactions in M.
- The pool also provides the miner with a new target T', which is typically much larger than the actual global target T. A larger target value increases the probability of finding N such that $HleqT$.
- The miner computes H and wins a *share* when $H \leq T'$. The pool subsequently gives the miner another hash of transactions in M, and the miner repeats Steps 1 and 2 to gain more shares. As $T' < T$, a miner typically wins a share every few minutes.[5]

[5] In the case a miner computes $H \leq T$, he cannot assemble the full block himself, as he has no knowledge of the full set of transactions in M.

A share is a record of the miner's effort. When the mining pool later mines an actual block, the pool will divide the block reward across the contributing miners in a way that is commensurate with individual shares. Most pools require miners to register a user name, password, and associate a *payout wallet address* where the pool server sends the user's share of bitcoins, as the pool will periodically pay the miners based on the miner's contribution (often using a pool-specific payment formula). Some pools also support *pseudonymous mining*. In this case, the worker provides a wallet address rather than a user name to the pool server, and all earnings are directly sent to the specified address.

1.4 Trading

A mined bitcoin has value because miners can sell it at public marketplaces known as *exchanges*. There are at least a hundred known exchanges that anyone can join to convert bitcoins into other assets, such as US Dollar or Euros [25]. Notable examples include Coinbase (based in the US) and BTC-e (based in Europe) [24, 21].

For a miner to sell this bitcoins, he does the following.

1. The miner sets up an account at an exchange, which is typically free.

2. The exchange provides the miner with a *deposit wallet address*.

3. The miner sends his bitcoins to the deposit wallet address. This transfer is recorded in the blockchain.

4. Now the bitcoins are under the control of the exchange. The miner simply has to trust that the exchange will take good care of his bitcoins, although there is evidence that some exchanges lost bitcoins due to attacks [10].

5. The exchange lists all current trades, where the miner can see how many coins others are buying/selling and at what price.

6. The miner sets a price for his bitcoins and lists them on the exchange. The miner can set any price.

7. If a buyer agrees to the price, the buyer purchases the bitcoins, and the corresponding fiat currencies (e.g. dollars) will be credited to the miner's account on the exchange. This exchange of assets happens entirely within the exchange, since the exchange is in control of both the bitcoins and the fiat currencies that the buyer had deposited earlier. The event is not recorded in the blockchain.

8. To withdraw these credits (i.e. *cash-out*), the miner provides the exchange with his banking information.

9. The exchange sends the corresponding fiat currencies to the miner's bank, minus some processing fees.

In this way, a miner can convert his bitcoins, obtained computationally by expending energy, into fiat currencies.

In addition, exchanges allow anyone, not just miners, to buy and sell bitcoins. An investor can make money through speculation — purchasing bitcoins (or similar crypto-currencies) at one price and selling them later when the price rises. In fact, speculation is not new. There is existing literature on penny stocks trading [31, 20]. In many ways, crypto-currency markets are similar to penny stock markets in terms of volatility and size. Many of the speculative behaviors in the penny stocks literature can also be observed in the world of crypto-currency. Chapter 3 will discuss in detail.

1.5 Summary

In this chapter, we show how Bitcoin processes transactions over a peer-to-peer network, how mining is a decentralized process, and how one can sell mined bitcoins at exchanges. In the next chapter, we focus on Bitcoin mining — a computationally intensive activity that converts electricity into bitcoins. We show how botnets took advantage of CPU cycles on compromised computers, mined bitcoins without paying for the energy bill, and made money from this operation. More importantly, using features discussed in this chapter, we track bitcoins into and out of these botnets to better understand their business model.

Chapter 2

Tracking botnets that monetized stolen computation

As described in Chapter 1, mining essentially converts electricity into money. While typically miners pay for electricity costs during the process, in this chapter we discuss a new type of miner that does not pay for electricity — botnets that possess a large number of compromised computers. In particular, we focus on botnets that mined bitcoins between 2012 and 2013. We use Bitcoin's blockchain, along with malware-related data, to examine the business model and profitability of mining botnets. In particular, we discover 10 such botnet operations with a total mining revenue of at least $118,000. This revenue to the botnets was at a cost of hundreds of thousands of compromised hosts; for some botnets, we estimate that the energy cost to the compromised hosts was at least twice the botnets' revenue. Still, the revenue was small, compared with the multi-million dollar spam industry in which botnets typically participated. We show that as dedicated Bitcoin-mining hardware gained popularity, botnets suffered from an exponentially decreasing revenue per unit computation, to a point where bitcoin mining on botnets was unlikely to be profitable. Nevertheless, this phenomenon is the start of a new trend in which miscreants can convert computational power into money, using whatever crypto-currencies that are profitable for mining.

2.1 Introduction

Mining crypto-currencies is tantamount to converting electricity into money. As shown in Chapter 1, a miner looks for hash collisions via brute force. Typically, this process fully utilizes computational resources such as CPUs or graphics cards, consuming dozens to even hundreds of watts per device, potentially over a prolonged period of time, thus incurring significant energy cost. Successful miners are rewarded with newly minted crypto-currencies. Miners can keep the reward or sell them later in exchange for fiat currencies such as US dollars.

One special type of miner does not need to pay for the energy cost: botnets [37]. They possess a large number of compromised computers, whose computational capabilities can be used for mining while infected victims themselves pay for the energy cost. In this chapter, we study a particular instance of this phenomenon: botnets that mined bitcoins between 2012 and 2013. The first Bitcoin mining malware was observed in the wild in June 2011 [51]; since then, numerous families of malware have taken up Bitcoin mining. The first family we identified with mining capability was NGRBot, a malware kit that has been available for several years. NGRBot is a generic malware platform with many different capabilities, such as stealing personal information, automatic spreading on USB and network disks, and launching distributed denial-of-service (DDoS) attacks. Instances of NGRBot have continued to mine, and recently an NGRBot variant spread through Skype messages was seen mining [13]. In mid-2012 several news stories documented ZeroAccess performing both bitcoin mining and click fraud at large scale [65]. Shortly after ZeroAccess and NGRBot, many other families of malware began to appear that installed (or dropped) bitcoin mining functionality.

To botnets, bitcoin mining offers an additional revenue stream with little extra infrastructure. When Bitcoin gained popularity, some botnets started tapping into the pre-

viously under-utilized computational capabilities and began mining bitcoins, in parallel with existing monetization schemes, such as stealing personal information and launching DDoS attacks. Effectively, botnets gains a new revenue stream by using existing infrastructure.

While bitcoin mining increases botnets' revenue, infected hosts experience higher energy costs as a result of the computationally intensive nature of mining. In one instance, as we will show later in the chapter, a botnet that made $8,000 of revenue from mining incurred an estimated energy cost of $20,000. Furthermore, users of infected hosts may notice degraded performance, as presumably some of the computational cycles are devoted to mining, rather than for users' productivity purposes.

Our goal is to show that, by tracking the movement of bitcoins on the blockchain and correlating with malware-related data, we can identify bitcoin-mining botnet operations and measure their economic activities: how much revenue and cost, and using what business model.

To this end, we first collect bitcoin-mining malware samples from multiple sources, such as security industry malware databases ThreatExpert and Emerging Threats. For each executable, we identify how it mines bitcoins — using both sandboxed execution and binary analysis — and extract the wallet addresses associated with botnet operators.

In order to estimate botnets' mining revenue, we track bitcoins moving into these addresses, only focusing on bitcoins that have previously come from mining, so that we exclude bitcoins that were associated with non-mining activities. Furthermore, using the blockchain data, we track the difficulty of mining over time, in an attempt to estimate the energy cost to the infected hosts and the per-CPU revenue to the botnets.

In addition, we combine our blockchain-based analysis with other data sources, such as communication with mining pool operators, passive DNS, and malware databases. We identify, where possible, the infrastructure that each botnet operation used and when

each operation was active, thus providing a comprehensive view of existing botnet bitcoin-mining activity over botnets.

In brief, our main contributions are:

- We identify a new monetization scheme for botnets to tap into compromised computational resources through mining.
- We analyze the blockchain to identify clusters of wallet addresses that belong to individual botnet operations.
- By examining these wallet addresses against the blockchain, we compute the total revenue of Bitcoin-mining botnets to be around 4,5000 bitcoins or $118,000.
- Combining the blockchain analysis with external data from mining pools, we show that, for some botnets, the estimated energy cost to the victims is twice the botnets' revenue.
- By analyzing the difficulty of mining on the blockchain, we show that botnets are likely to suffer from an exponential decreasing profitability from mining.

The rest of this chapter is organized as follows. We first describe how botnets mine bitcoins in Section 2.2. In Sections 2.3 and 2.4 we describe our measurement methodology before presenting our results in Section 2.5. We discuss the economics of malware mining in Section 2.6 before concluding in Section 2.7.

2.2 How Botnets Mine Bitcoins

Botnets typically mine bitcoins using mining pools. Recall from Section 1.3.2 that mining pools allow miners to receive incremental payouts with significantly lower variance than solo mining. We observe three distinct botnet mining pool structures in the wild, summarized in Figure 2.1.

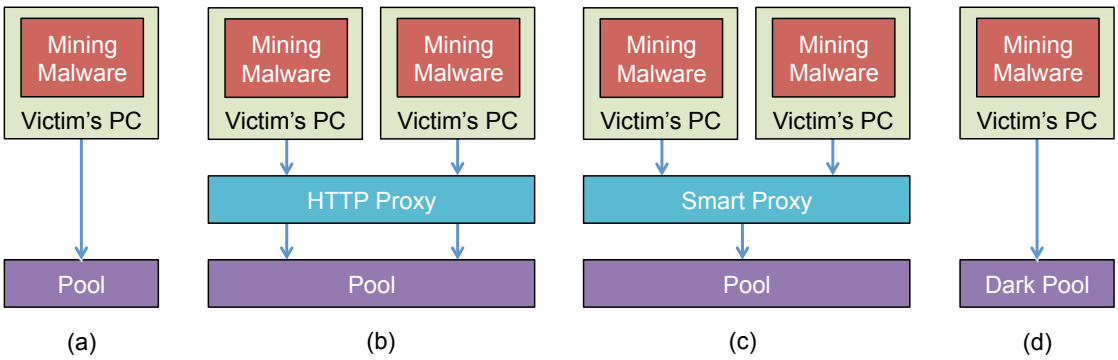


Figure 2.1. Different ways in which mining malware connects to mining pools: (a) directly to a pool, (b) via an HTTP proxy, (c) via a smart proxy, and (d) directly to a dark pool.

Direct pool mining: At its simplest, mining with a botnet is no different from mining using one's own hardware. One of the more popular techniques for botnet-based Bitcoin mining is to simply distribute a mining executable (such as cgminer.exe or bfgminer.exe) inside a wrapper script that specifies all the parameters required to mine. This removes any cost associated with developing or modifying botnet software and is popular with Trojans distributed as pirated software. An example is the FeodalCash family of botnets (Section 2.5.4) that mine directly at Eligius, a public pool.

A botmaster simply needs to specifying a mining pool and provide his own credentials. Each compromised PC will connect directly to the mining pool as a worker and start doing work on behalf of the pool; the pool will direct payments to the botmaster's account. We call this approach *direct pool mining* or simply *direct mining*.

Mining pool operators can easily detect direct mining, as it involves a large number of hosts, all using the same account, with each host providing very little CPU power for the mining task. Once detected, most pool operators will ban such users.[1] Once banned, the botnet becomes useless if there is no way to change the mining pool or credentials used by the bot.

[1] One pool operator reported having to relent after his pool servers came under DDoS attack from the botnet.

Proxied pool mining: To overcome some of the drawbacks of mining directly, a botmaster can proxy connections to the pool through a server he controls, a mode we call *proxied pool mining* or simply *proxied mining*. Since the getwork protocol (Section 1.3.2) uses HTTP as a transport, a botmaster can simply employ an HTTP proxy (e.g., nginx). Using a proxy has two advantages. First, it hides the IP addresses of the bots: all connections appear to come from the proxy itself, making the botnet seem more like a single, powerful miner[2]. Using a proxy also provides a level of indirection, allowing the botmaster to switch to new credentials or mining pools if banned.

Alternatively, the botmaster can design a more sophisticated *smart proxy* that does more than blindly pass through getwork requests. In this configuration, a proxy operated by the botmaster requests work from a pool as a normal worker, but then splits the work into smaller units provided to the bots. This architecture requires modifying existing mining pool software to support such operation, an additional investment. The Fareit botnet, however, uses a form of smart proxying. The server to which the bots connect operates as a mining pool server for the bots, but appears as a single worker to a P2Pool mining pool; more detail is provided in Section 2.5.5.

The downside of either form of proxy mining is that it requires additional infrastructure — the proxy. Several mining operations we observe use this mechanism, such as DLoad.asia and ZeroAccess (Sections 2.5.1 and 2.5.2).

Dark pool mining: The final option is for the botmaster to maintain his own pool server. In this mode, which we call *dark pool mining* or simply *dark mining*, bots connect to a mining pool controlled by the botmaster. In this configuration, the botmaster's dark pool must participate in the Bitcoin peer-to-peer network. In addition to

[2]It is still possibly detectable, as the time between sending a getwork request and providing a corresponding proof of work could be longer than then the gap produced by a single miner. The number of getwork calls is also unchanged.

the infrastructure investment in the pool server, the botmaster loses the consistency of payouts provided by a (larger) pool. The botnet now only generates revenue if it finds a block itself. A botnet of 10,000 compromised desktop PCs each capable of 10 MH/s running continuously mines one block every 16 days on average, as of August 2013.

2.3 Using Blockchain to Estimate Botnet Revenue

In this section, we show how to use the blockchain to estimate botnets' revenue from mining bitcoins. In particular, we first identify wallet addresses that botnets used (Section 2.3.1). Then we identify income to these addresses from mining pools, so that we consider revenue exclusively from mining, rather than other botnet activities (Section 2.3.2). Finally, we analyze the mining revenue as received by botnets' other wallet addresses (Section 2.3.3).

2.3.1 Identifying Botnets' Wallet Addresses

Recall, from Chapter 1, that mining pools generally require registration to mine with the pool. When mining, the worker supplies the user name created at registration; all earnings are credited to that user and periodically transferred to a wallet address specified at registration. (The exception are pools that support so-called pseudonymous mining, in which the worker specifies the payout wallet address — rather than a user name — when connecting; in this case, no mapping is necessary.)

Pools do not normally list miner wallet addresses publicly, making it difficult to connect mining activity to payouts. To obtain this information, we resort to non-technical means, contacting the pool operators directly to ask for information about specific accounts. Some operators kindly provided us with this information, either sharing with us the payout address or the total amount paid out to it. Operators are sensitive about privacy and only provided information about users they themselves had identified as

botnet miners.

When a botnet does not use mining pools, we extract the botnet's wallet addresses using steps described in Section 2.4.2.

2.3.2 Estimating Revenue From Mining Only

Because all transactions are visible, knowing the addresses to which mining payments are sent allows us to estimate the earnings of a specific miner via examination of the blockchain. We use the payout addresses provided to us by various pool operators. Given these addresses, we first need to isolate mining payouts from other types of transactions. To identify mining pool payouts, we use the technique of Meiklejohn *et al.* [47] to identify the payout transactions of five major mining pools: 50 BTC, BTC Guild, Deepbit, Eligius, and P2Pool. Briefly, this technique relies upon knowledge of patterns or addresses specific to each pool. For instance, a Deepbit payout transaction always uses the same address as the sender, and BTC Guild always sends its initial mining reward to the same address (at which point it pays each miner in an identifiable chain of transactions).

Once we have a collection of transactions representing the mining payouts to the address, we then consider all mining revenue to be derived from botnet mining. This number forms a lower bound on the actual mining revenue, as the techniques of Meiklejohn *et al.* [47] may fail to identify certain payout transactions in order to avoid false positives. While one might argue that it is possible for a botmaster to re-use this same address for legitimate mining operations, we view this possibility as unlikely. First, re-using the same Bitcoin address for multiple purposes has the potentially negative effects that it confuses bookkeeping for the owner and serves to de-anonymize her (as two users now know her by the same pseudonym). Second, re-using the same address has essentially no positive effect, as generating a new Bitcoin address requires generating only

a signing keypair, and thus has virtually no computational cost. Finally, and perhaps most importantly, re-using the same payout address jeopardizes legitimate mining revenue, as botnet miners are routinely banned by pool operators.

In addition to using the blockchain to estimate revenue, some mining pools offer public statistics on the earnings of individual miners. One pool, Bitclockers, provides a leader board, showing total user earnings, and work contribution for each solved block. We use this information to determine the earnings of 38 users (Section 2.5.10). In addition, the Eligius and 50 BTC pools provide public statistics about users mining pseudonymously. For malware mining operations using these pools, we obtain earnings and other information directly from these public statistics. Finally, our source of information about the Fareit botnet is the botnet itself (Section 2.5.5). This botnet operates its own mining pool servers, operating as a dark pool (Section 2.2). The mining server software is a fork of the P2Pool mining server code base.[3] This particular mining server provides miner statistics, which we are able to obtain directly from the dark pool servers.

2.3.3 Clustering Wallet Addresses

While re-using a single Bitcoin address might be unattractive to a botmaster, there are a number of reasons why one might use multiple addresses. For example, using different pool credentials for each malware distribution campaign would allow her to track earnings for each campaign separately. Using separate addresses also offers some protection against detection by a pool operator, as it spreads the activity across several accounts; even if one address were blocked by a pool operator, only those bots mining to that banned address would be affected.

To identify addresses belonging to the same botmaster, we rely on the observation — due to Satoshi Nakamoto himself [49] — that addresses used as inputs to the same

[3] http://github.com/forrestv/p2pool

transaction are controlled by the same user, as described in Section 1.2.3. This technique is employed frequently in studies of anonymity within the Bitcoin network [47, 60, 61], and we use it to cluster otherwise distinct malware. This clustering is especially useful for smaller mining operations: e.g., in the case of the BMControl malware, we first identified the family using this technique, and later confirmed the clustering by identifying and decoding its Pastebin-based command-and-control channel.

Clustering also allows us to identify other wallet addresses used by the botmaster. We refer to wallet addresses directly associated with malware mining as *primary* wallet addresses. We refer to wallet addresses in the same cluster as a primary wallet address, but which are not themselves primary addresses, as *secondary* wallet addresses. The income received by secondary wallet addresses may include mining income from other malware mining operations of the same botmaster that are unknown to us. It may also include, however, other sources of income, including some that may be legitimate. For this reason, we report the included income of secondary wallet addresses separately.

2.4 Discovering Botnet Operations

In this section, we describe our methodology to understand the modus operandi of botnets, from identifying mining malware, extracting mining credentials, identifying pool proxies, to estimating the infection population.

2.4.1 Identifying Mining Malware

To our knowledge, all malware engaged in Bitcoin mining uses the HTTP-based getwork protocol supported by existing mining pools (as of 2013). We therefore rely on this signal as our primary means of identifying mining malware: we use mining protocol traffic in network traces of a malware binary's execution as evidence that it is engaged in Bitcoin mining.

To obtain network traffic of various malware, we execute the binaries in our own malware execution environment or rely upon data from public and private sandboxes, including ThreatExpert[4] and Emerging Threats[5]. Some environments also provide OS-level monitoring such as logs of registry keys changed and files modified. We manually assess if a sample is performing Bitcoin mining by inspecting the traffic and looking for evidence that a particular sample is requesting work from a Bitcoin pool server. Then, using traffic and OS-level logs we construct queries to identify additional samples with similar characteristics. In total we identify over 2,000 executables that connect to pools and mine bitcoins.

2.4.2 Extracting Mining Credentials

Most mining malware relies on generic, off-the-shelf mining clients to do the actual mining. The malware executes the client and provides the pool name and worker user name — mining credentials passed as parameters to the miner — on the command line.

Command-line arguments: In many cases, we can extract these command-line arguments directly from the packaged binary statically. In other cases, we extract the mining credentials from the process execution environment; an example is the BMControl malware (Section 2.5.3), from which we extract the usernames from the memory dump.

HTTP basic authentication: We can also extract the pool name and miner user name from the network trace of the malware. The getwork protocol relies on HTTP basic access authentication to provide the miner user name to the pool.[6] With HTTP basic authentication, a Base64-encoded user name and password are submitted in an

[4] http://www.threatexpert.com

[5] http://www.emergingthreats.net

[6] The password is ignored by all pools of which we are aware, since there is no benefit to doing work in another miner's name, nor is there any obvious harm to the miner in whose name the work is submitted.

HTTP header, making them easy to extract from a network trace. We use this method of extracting miner identifiers for binaries executed in third-party sandboxes.

Command-and-control channel: Some malware does not embed the pool or worker name into the binary. Instead, the mining credentials are obtained through a custom command-and-control channel. The Fareit botnet (Section 2.5.5) uses the Dropbox and Pastebin Web services to disseminate mining credentials to bots. The contents of the Dropbox or Pastebin document are usually obfuscated using algorithms ranging from simple Base64 encoding to custom encoding schemes.

We manually reverse-engineer the malware to determine the technique used to obfuscate the data received through the command-and-control channel. For simple obfuscations, we can recreate the de-obfuscation algorithm and use it to continually retrieve the pool information and worker credentials. One example of this is the first version of the BMControl botnet that uses Pastebin to host Base64-encoded configuration information. The configuration includes the command-line parameters for the mining executable (in this case bfgminer[7]) as well as a list of the pools and worker credentials to use.

More complex obfuscation can be difficult to reverse-engineer; in this case we run the malware and take a memory snapshot after the malware has de-obfuscated the payload. An update to the BMControl botnet included a change in the obfuscation technique, so we use memory snapshots to capture the decoded payloads. The Fareit malware family also uses more substantial obfuscation, making memory snapshots a prudent technique for automatically decoding the configuration.

These techniques allow us to identify the mining credentials for all the samples of malware we find mining bitcoins. Based on the pools we observe the malware accessing,

[7] http://bfgminer.org/

we find that 74% of the samples connect to well-known public pools (light pools), while the remaining 26% connect to unknown private pools (dark pools).

Pool operators: Finally, some user names and wallet addresses were provided to us by public pool operators as miners they believed to be using a botnet. We confirm their claims by locating the corresponding malware MD5 hashes (e.g. the gamer-targeting botnets in Section 2.5.10).

2.4.3 Identifying Pool Proxies

The techniques described above work for malware engaged in direct public pool mining; that is, where the malware connects directly to a public pool, or for malware where the dark pool provides information, as in the case of the Fareit botnet. In some cases, however, the pool server to which the malware connects is not a known public pool nor does it report any useful information via a statistics Web page. These types of malware mining operations are the hardest to measure. If the server is no longer in operation, our options are limited still further. Here we describe the techniques we use to glean what information we could about such mining operations.

Cross-login test: Since the getwork protocol uses HTTP as a transport, it can be proxied by an HTTP proxy such as nginx without modification. In the simplest case, incoming connections are transparently proxied to a public mining pool. Such a proxy passes through all HTTP headers unchanged, including the Authorization header used by HTTP basic access authentication. In this configuration, bots must use credentials that are valid for the destination pool. To detect this form of proxying, we create miner accounts at several major mining pools and attempt to connect via the suspected proxy using the registered user names, as well as one randomly-generated name we confirmed did not correspond to an exist user name at any of the major pools. If the suspected

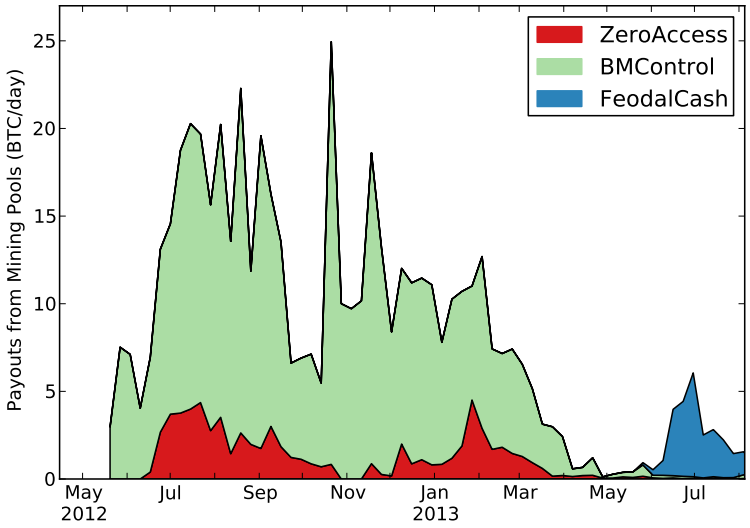
proxy proxies to one of the major pools, then exactly one user name should succeed in authenticating — the user registered with the public pool to which to proxy is pointed. We identify one transparent proxy: `domain-crawlers.com` transparently proxies all connections to the 50 BTC public pool.

We also test whether pool credentials found in malware could successfully authenticate to a public mining pool. We find this to be the case for a number of worker user names; however this test is not conclusive so we draw no conclusions from this test alone. Rather, we use this information to engage with pool operators; in cases where the pool operator independently confirms that the miner was suspected of mining using a botnet, we include the miner in the analysis.

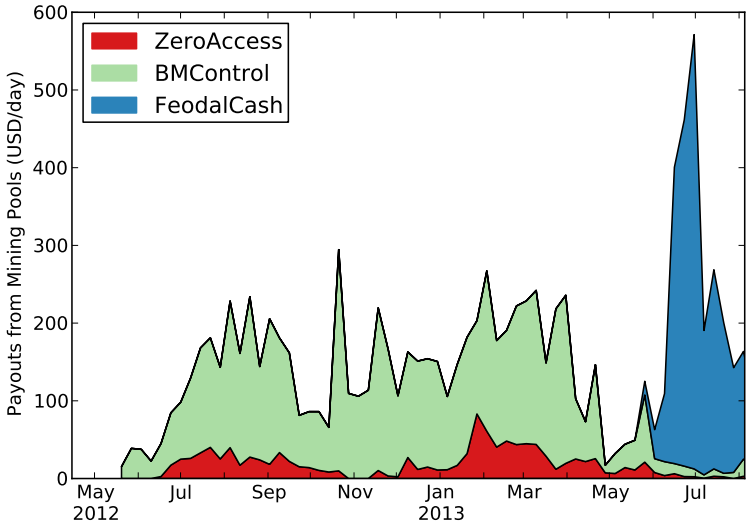
Passive DNS: The lifetime of a dark mining pool is usually indicative of the lifetime of the corresponding botnet. To determine when such pools were first and last seen, we use the passive DNS data from the ISC Security Information Exchange.[8] The passive DNS dataset contains DNS lookups issued by recursive resolvers at several vantage points, including a major US consumer ISP. We use this dataset for the purpose of discovering the DNS A-records historically returned for domain names of interest between October 2011 and April 2013. In addition to showing the first- and last-seen dates of dark pools, this data set also illustrates the overlap of A-records across different domains. For instance, two dark pools, `dload.asia` and `aquarium-stanakny.org`, pointed to the same IP addresses in the past. This coincidence suggests that the same botnet operation may be behind both domains.

Block reversal: In some cases, we can attribute a successfully mined block to a particular mining pool. The getwork call needs to provide different work for each worker, but if there are no new transactions added between getwork requests, subsequent

[8] https://sie.isc.org/



(a) Total daily bitcoin payouts for each botnet.



(b) The same payouts expressed in USD at the day's exchange rate.

Figure 2.2. Two stacked line graphs showing the amount of mining payouts that botnets received over time, in BTC and in USD. The aggregate mining of all the operations never exceeds 0.4% of the bitcoins generated each day.

calls would produce the same value. Pool servers counter this problem by changing the coinbase value for each call to getwork, using it as an additional nonce. Here, we use it as a signature of a particular pool: while the pool clearly will not provide the same coinbase to two different workers, many pools provide similar coinbases across workers.

We repeatedly poll all known pool servers several times a second for a period of three weeks. Then, for every block published during our monitoring, we perform a brute-force search modifying the coinbase of the published block (based on changing only the bits which change when examining the most-similar coinbase in the blockchain at the time) and checking whether the modified coinbase corresponds to one of our recorded getwork requests. A match indicates that the monitored pool is likely to have mined that particular block.

Although this approach is only effective against pools with low coinbase entropy, we are able to attribute blocks to both the Deepbit and 50 BTC pools. It also confirms that `domain-crawlers.com` was proxying to 50 BTC, as we discover multiple blocks where getwork calls to both a 50 BTC pool server and the `domain-crawlers.com` server correspond to the blocks published in the blockchain, suggesting that `domain-crawlers.com` simply forwards the getwork request on to 50 BTC.

Leaked data: In one case, we could glean information about a Bitcoin botnet mining operation from leaked data. Specifically, information about FeodalCash, an affiliate-based program that pays botnet operators to install their Bitcoin mining malware, was publicly posted on the Internet. This data enables us to identify earnings from the entire operation as well as earnings from individual affiliates of the program.

2.4.4 Estimating Infected Population

We contacted a top anti-virus software vendor (with an install base of millions across the world) with the MD5 hashes of the mining malware, and obtained from

them, for each of our 976 samples, an aggregate list of countries from which the mining malware was seen to be operating along with the count of unique machine infections detected, over a period of about one and a half months around July 2013. The vendor also provided us with the count of their install base per country. Based on this information and the distribution of computers across the world, we can extrapolate the total population of malware infections per malware family as follows.

Let I_i be the number of infections observed by the vendor in country i, and M_i be the number of machines in that country that subscribe to the vendor's monitoring service. Using an approximate number of computers T_i for country i, the estimated bot population E_i can be computed as

$$E_i = (I_i/M_i) \times T_i.$$

The CIA factbook [23] reports the number of Internet users for 2009; we assume one computer per Internet user for T_i. In practice, this assumption holds for known data points (e.g. Worldwide PCs deployment is projected[9] to be 1.9 billion in 2013, while the CIA factbook estimates the total Internet population as 1.8 billion).

We expect the estimates here to be lower bounds for the following reasons: first, computers that do not have anti-virus protection from the vendor are not counted, including computers with no anti-virus protection at all — such computers are likely to be infected with malware over the long term, contributing to more mining. Second, the estimates are only for the specific binaries we collect. Many of the malware families involved in mining are polymorphic, so we expect many more samples that are not considered here.

[9] According to the Computer Industry Almanac, Worldwide PC use executive summary

Table 2.1. Bitcoin mining operations covered by our study.

Family	*Sec.*	*EXEs*	*Wlts*	*Active period*	*BTC*	*USD*
DLoad.asia	2.5.1	322	—	Dec '11 – Jun '13	10,000?	10,000?
ZeroAccess	2.5.2	976	3	Dec '11 – Nov '13	486	8,291
BMControl	2.5.3	54	47	May '12 – May '13	3,097	46,301
FeodalCash	2.5.4	—	238	May '13 – Nov '13	168	15,941
Fareit	2.5.5	5	1	Apr '13 – Nov '13	265	30,448
Zenica	2.5.6	67	—	—	170?	—
HitmanUK	2.5.7	5	1	Mar '13 – Nov '13	4	362
Xfhp.ru	2.5.8	42	—	—	—	—
Skype Miner	2.5.9	17	—	—	250?	—
Misc.	2.5.10	—	—	Dec '11 – Nov '13	539	17,166

2.5 Analysis

Recall that our goal is to identify major Bitcoin mining operations, their scope, and revenue. In this section we describe nine major mining operations, including a Bitcoin mining affiliate program (Section 2.5.4), as well as 80 smaller mining operations, most represented by a single executable found in the wild.

Table 2.1 summarizes our findings. The *EXEs* column shows the number of executables we observe engaged in mining. Several families of malware — ZeroAccess especially — are very aggressive about repacking binaries; it is likely that our sample does not represent the entire set of binaries in the wild. (Recall that our main means of identifying mining malware are reports by ThreatExpert, VirusTotal, and Emerging Threats.)

The *Wlts* column gives the number of wallet addresses known to us that receive payouts. FeodalCash, an affiliate program, has the largest number of wallet addresses (238) because each affiliate mines to a unique wallet, allowing earnings to be credited properly. BMControl also has a large number of wallets (47); we suspect it is also an affiliate program.

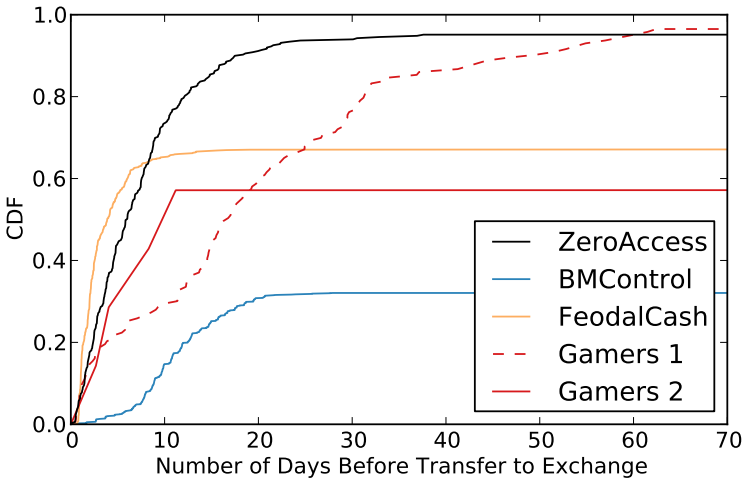


Figure 2.3. Delay in transfer to exchanges for different botnets.

The *Active period* column shows the period when the mining operation was active, based on mining data and malware distribution activity.

The *BTC* column shows our estimate of each operation's total earnings. In most cases, earnings are measured directly, either from mining pool statistics (ZeroAccess, Fareit, BMControl, and FeodalCash), or from the blockchain based on payout address (DarkSons and HitmanUK), based on earnings reported by the pool (Skype Miner and Zenica), and finally based on order-of-magnitude estimates by the pool operator (Redem). Small mining operations covered in the miscellaneous section (Section 2.5.10) use all four of the above types of estimates.

The *USD* column provides an estimate of the earnings in US dollars, using the exchange rate at the time of payout. Thus, although earlier mining operations (e.g., DLoad.asia) earned over 10 million dollars' worth of bitcoins at the exchange rates in effect on November 30, 2013, at the time of mining a bitcoin was worth considerably less. In two cases — Skype Miner and Zenica — we do not have accurate information about when the bitcoins were earned, so cannot estimate the equivalent US dollar value accurately. We plot our estimate of the daily earnings of the five largest operations

(in terms of revenue) in Figure 2.2, and their cumulative earnings in Figure 2.4. The latter further breaks down the earnings into just those transferred from the primary wallet addresses (i.e., the address(es) associated with the mining credentials used by the malware) as well as transfers from associated wallet addresses (see Section 2.3.3).

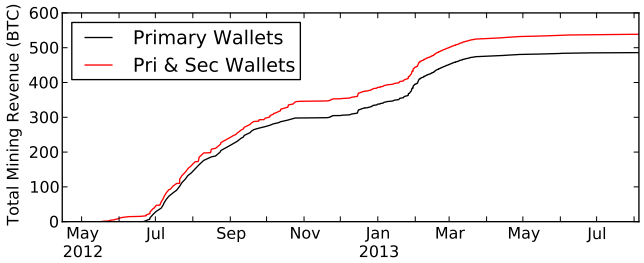
Our estimates notwithstanding, the true "takehome" earnings in terms of USD (or any other fiat currency) depend entirely on how — and when — the bitcoins are "cashed out", typically by transferring them to an exchange. Hence, transfers to exchanges are of particular interest, as they serve — with very few exceptions — as a necessary precursor to cashing out of the Bitcoin economy. Unfortunately, because most exchanges double as online banks we cannot claim definitively when — or if — all these earnings were converted to fiat currency.

Moreover, in some cases, the mining profits might travel through several intermediate addresses before arriving at an exchange. For simplicity, we consider only cases with no intermediate addresses; i.e., cases where the bitcoins earned from mining are spent immediately at an exchange. We define the transfer time as the interval between the mining payout and the actual transfer. We use the techniques of Meiklejohn *et al.* [47] to identify wallet addresses associated with exchanges. Figure 2.3 shows the delay between when a botnet receives payment for mining and when it transfers its earnings to an exchange. In most cases, botmasters liquidated their bitcoins shortly after mining.

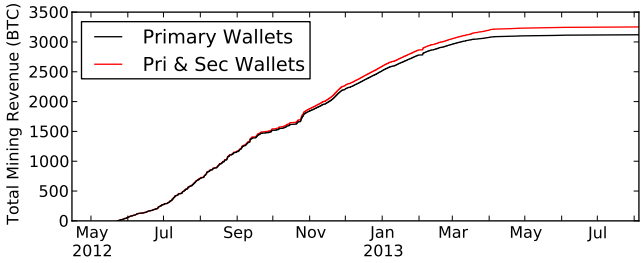
In the remainder of this section, we describe each of the mining operations listed in Table 2.1 in greater detail.

2.5.1 DLoad.asia (Redem and DarkSons)

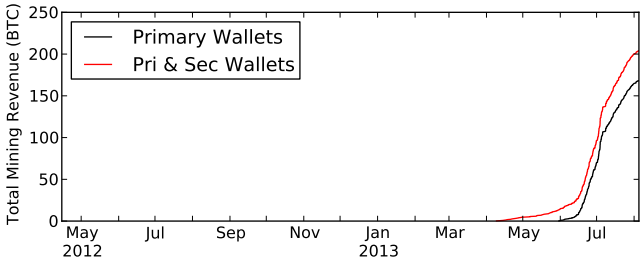
The DLoad.asia operation is one of the earliest major mining operations we encountered. More properly, the DLoad.asia operation consists of several mining operations using shared infrastructure. At least two individuals are behind the operation, known by



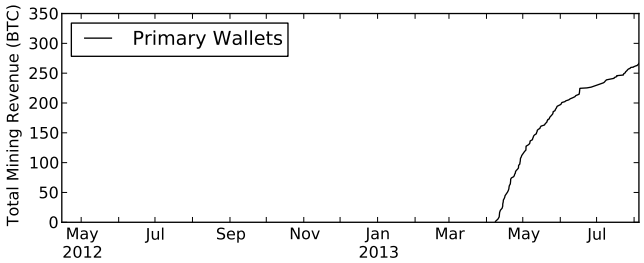
(a) ZeroAccess



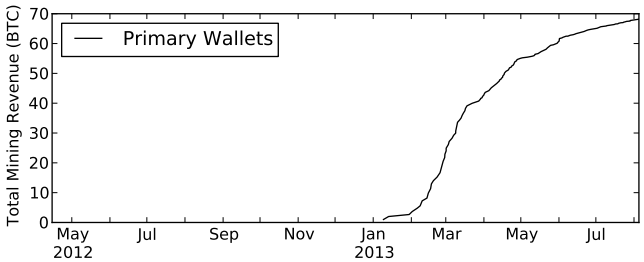
(b) BMControl



(c) FeodalCash



(d) Fareit



(e) Gamers

Figure 2.4. Mining revenue by different botnet operations.

the handles Redem (a.k.a. Mpower) and DarkSons (a.k.a. MrDD).

Operation: Based on information provided by one public mining pool operator, these individuals began mining in 2011, initially connecting to the pool directly and later via a proxy. These botnets continued mining using the same pool user names (variations of "Redem" and "DarkSons") even when connecting through a proxy. Later generations of malware used different miner user names and proxy domain names. Despite this, the server IP addresses and domain names were not changed in unison, making it possible to track the infrastructure as it evolved. Most recently, the DLoad.asia infrastructure was used as a mining proxy and NGRBot command-and-control channel. As documented by the "Inside Your Botnet" blog, the Redem and DarkSons names continued to appear in IRC channel and user names [35, 36, 37, 38, 39, 40], as well as in domain registration records.

Earnings: The pool operator shared with us the wallet address that DarkSons used. The wallet address was last active in November 2012, at which point it had amassed 2,403 BTC. The techniques of Meiklejohn *et al.* [47] are unable however, to identify any direct payments from mining pools. The blockchain does reveal a number of transactions in which the DarkSons wallet received block rewards through an intermediate wallet. The botnet received a total of 1,681 BTC through these transactions.

We are unable to locate a payout address for Redem. However, the pool operator recalls that the botnet connected to the pool using over 100,000 unique IP addresses and had a peak mining rate of over 100 GH/s. The operator estimates that the bot earned at least 10,000 BTC. During that time period, however, a bitcoin was worth only about $1. Our estimate of Redem's portion of DLoad.asia earnings in US dollars shown in Table 2.1 is therefore based on a 1:1 exchange rate.

Table 2.2. Distribution of DLoad.asia infections by country.

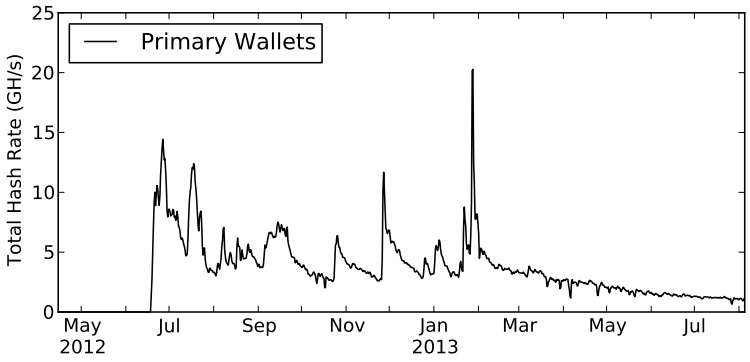
Country	*Share*	*Est.*
Brazil	16.0%	10,600
Malaysia	9.5%	19,300
Indonesia	8.7%	11,700
Russia	5.9%	4,200
South Korea	5.8%	7,100
Others	54.1%	71,800

Population: Although the DLoad.asia infrastructure is no longer active, infected hosts can still be found in the wild. Table 2.2 presents our estimation of the geographic distribution of infections based upon the data provided to us by a major anti-virus software vendor. The *Share* column shows the ratio of the number of infected hosts to the total number of hosts with the vendor's product in a given country. Based on this percentage and the number of computers in the country, we estimate the infection population. Brazil accounts for the largest share of infections; the vendor's coverage is smaller in Brazil than in Malaysia, however, so the estimated number of infected hosts is larger in Malaysia.

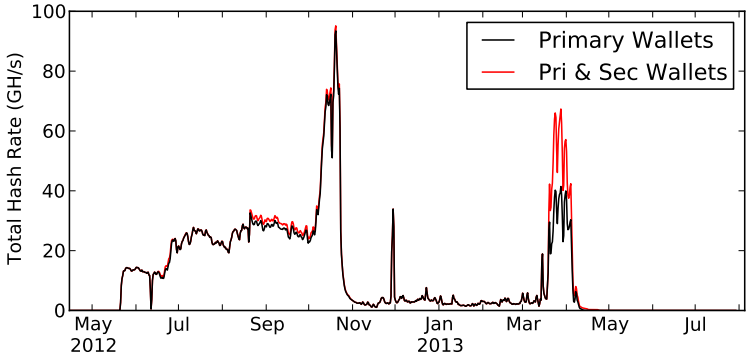
2.5.2 ZeroAccess

As of November 2013, the ZeroAccess botnet was one of the largest botnets, with estimated 9 million infected PCs of which one million are online at a given time [65]. ZeroAccess uses drive-by downloads and other methods to infect victims [33]. The core of the botnet is a rootkit and peer-to-peer command-and-control (C&C) protocol. Using the C&C protocol, bots can fetch modules that enable the bot to carry out tasks such as mining bitcoins or committing click fraud. Bots can be configured to perform only one task. It is possible to update the modules as necessary through the same C&C protocol.

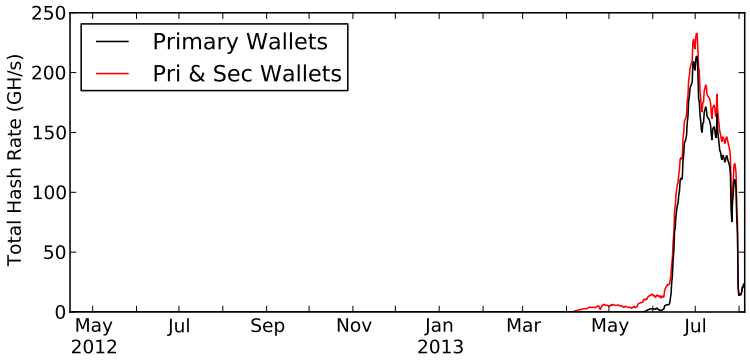
Operation: ZeroAccess began mining via a proxy server, `tang0-hote1.com`, and changed proxy servers several times to domains such as `google-updaete.com` and



(a) ZeroAccess



(b) BMControl



(c) FeodalCash

Figure 2.5. Mining rates of three botnet operations at Eligius, a mining pool that publishes each user's hash rates over time.

great-0portunity.com. According to our passive DNS data, these domains were first active in December 2011. As of November 2013, they were not active; as of mid-June 2013 ZeroAccess was mining directly through Eligius, a public mining pool that offers detailed hash-rate graphs for every user. Using credentials embedded in the ZeroAccess malware, we find one wallet address, 1ASNjJ, that it uses to mine at Eligius. Figure 2.5 presents the daily mining rates for ZeroAccess and two other operations that also use Eligius (discussed in subsequent sections).

Earnings: Our analysis of the earnings of ZeroAccess is limited to the most recent version that mines directly through Eligius. So far, the botnet has received more than 400 BTC from mining payouts (Figure 2.4a). As of November 2013, the botnet was mining at less than 1 GH/s, although the peak in February 2013 was close to 20 GH/s (Figure 2.5a).

Population: Based on information provided by the security vendor, most of the Bitcoin-mining bots were located in Europe, with over 25 countries in Europe accounting for about 50% of all observed infections. On the other hand, the malware itself is widespread, with infections detected in more than 60 countries. Table 2.3 shows distribution of the observed bot population for 976 binaries for the top five infected countries, as a percentage of total infections observed in the *Share* column, while the *Est.* column gives the number of infections extrapolated as described in Section 2.4.4.

The population estimate is much lower than previously published estimates [62, 65], which suggest the ZeroAccess bot population is between 1.2 to 9 million. However, our estimates are only for the 976 binaries we obtained and know to engage in Bitcoin mining for specific wallets. ZeroAccess is known to be polymorphic, so a large number of binaries are expected. Anti-virus vendors we checked with had well over a hundred thousand binaries labeled as ZeroAccess. Hence, we do not claim that our estimate

Table 2.3. Distribution of ZeroAccess infections by country.

Country	*Share*	*Est.*
United States	14.9 %	2,600
France	12.2 %	1,800
Russia	8.2 %	800
Czech Republic	5.1 %	900
Canada	4.8 %	817
Others	54.8 %	10,600

represents the overall ZeroAccess bot population or its potential mining profits, only the subset we observe in action.

Another way to localize the botnet is to use the diurnal pattern of its operation [26]. To analyze the periodicity, we find the hours of each day (in UTC) at which the botnet's hashing rate reaches a local minimum. Then we compute the probability distribution of these relatively dormant hours, a histogram of which is shown in the leftmost portion of Figure 2.6. As shown in the graph for ZeroAccess, the botnet is the slowest around midnight UTC, suggesting that the majority of infected hosts that mine with the `1ASNjJ` wallet address are located in Asia [26]. However, Table 2.3 suggests that the US has the largest bot population. It is likely that the botnet uses multiple wallet addresses in its binaries, of which we are able to find only one.

Energy cost to infected hosts: Figure 2.5, if we assume the mean hash rate is about 3 GH/s per day for a year (during which the botnet was active), we can estimate the total energy cost per victim. We further assume that a standard 30-Watt CPU can mine at 4 MH/s, and that electricity costs $0.10/KWh. Then the total energy cost is: 3 GH/s / 4 MH/s $\times$ 30 Watt $\times$ 365 days / 1000 KWh = $19,710. Compared with the $8,000 of total revenue, the energy cost to the victim is more than twice of mining revenue for the botnet.

Transfers to exchanges: The ZeroAccess line in Figure 2.3 shows the distribu-

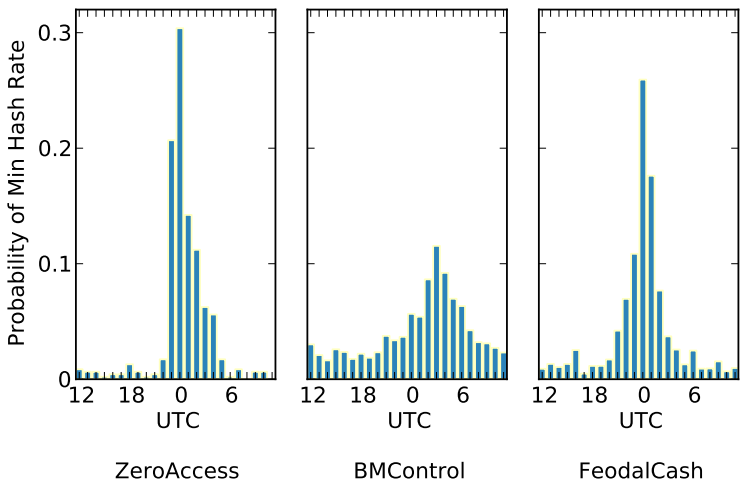


Figure 2.6. The distribution of times (in UTC) at which Eligius-based botnets achieve minimal mining rate.

tion of mining revenue that was transferred (within a single hop) to an exchange. The botnet transferred to an exchange more than 90% of the mining revenue that its primary wallet addresses received, using BTC-e as the primary exchange. The median time to do so is about a week. The botnet moved the remainder of its revenue to wallet addresses that we cannot identify as exchanges. These earnings might have been reinvested within the Bitcoin economy, or they might have been transferred to an exchange through intermediate wallet addresses.

2.5.3 BMControl

Another botnet that mines at Eligius is one we call BMControl, which can be identified by its command-and-control channel that uses specific PasteBin URLs to distribute configuration data to bots. We name this family of malware based upon the PasteBin user that uploaded the configuration data, BMControl. Upon startup, the malware retrieves and decodes the data contained in the PasteBin URL, and executes the mining binary. The configuration is a Base64-encoded string that includes the parameters

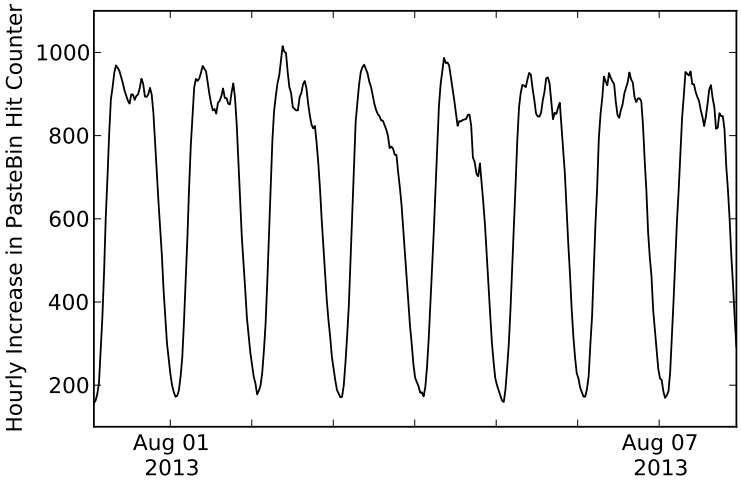


Figure 2.7. PasteBin counters for BMControl configuration URLs.

to run the mining executable and credentials for logging into the pool servers. The BMControl botnet was documented online in September 2012 [66].

Operation: BMControl has mined through proxies as well as directly through several pools. The configuration file at PasteBin contains a list of worker credentials for pools. When we first began monitoring this botnet, the configuration included only Bitcoin mining pools and used wallet addresses as worker names (a common feature of several Bitcoin pools). The primary pools that were used for Bitcoin mining were Eligius, 50 BTC and EclipseMC. Subsequent versions of the configuration file do not use Bitcoin wallet addresses as worker names, instead preferring to list usernames and passwords for the pools. The most recent configuration files for BMControl have included Litecoin pools and new worker credentials.

Earnings: For each distinct PasteBin URL, there is a counter of unique visitors that attempts to identify new visits based on cookies and IP addresses. For the two primary BMControl PasteBin URLs, there are over 8 million unique visits. This number increases between 200 and 1,000 every hour. The rate that the counter increases for one

week at the beginning of August, 2013 is shown in Figure 2.7. Since this PasteBin post is only useful if one can decode the contents and it requires knowledge of the URL to find it, we can reasonably estimate that increases in the counter are due to new infections and daily check-ins by the malware. Using this, we estimate that there are around 16,000 bots online each day.

As seen in Figure 2.4b and Figure 2.5b we no longer see the BMControl botnet mining bitcoins. Instead, the botnet has changed to mining litecoins through `litecoinpool.org`. We have confirmed this by decoding the configuration file as well as through contacting the `litecoinpool.org` administrators who have acknowledged that the workers used by BMControl are earning litecoins. We discuss Litecoin mining further in the Epilogue.

Using the last known mining rates (Figure 2.5b) and the estimate of 16,000 bots active per day from the PasteBin counter increases, we estimate that the average mining rate per bot is 3.75MH/sec.

Population: Eastern European countries account for more than 80% of the BMControl infections, with Bulgaria dominating the list shown in Table 2.4. This matches well with the diurnal cycle of the mining rate shown in Figure 2.6. The minimum mining rate happens around 3:00 UTC and Bulgaria is on Eastern European Time (UTC +2 or +3).

Energy cost to infected hosts: Using the same technique to estimate the energy cost as Section 2.5.2, if we assume the mean hash rate is about 10 GH/s per day for a year (during which the botnet was active), we can estimate the total energy cost per victim as $65,043. Compared with the $46,000 of total revenue, the energy cost to the victim is roughly 1.5 times of mining revenue for the botnet.

Table 2.4. Distribution of BMControl infections by country.

Country	*Share*	*Est.*
Bulgaria	50.6%	99,900
Turkey	28.8%	40,000
Macedonia	2.7%	7,000
Brazil	1.4%	2,200
Slovenia	1.4%	3,300
Others	15.0%	52,000

Transfers to exchanges: Using the same methodology as for ZeroAccess, we examine how BMControl transferred its mining revenue — which it received in its primary wallet addresses — to exchanges. According to Figure 2.3, the botnet transferred around 30% of the revenue, with a median transfer time of around two weeks. Most of the transfers took place at the Bitcoin-24 exchange.

2.5.4 FeodalCash

Details about FeodalCash, the last of the major botnet operations we see mining at Eligius, were leaked and publicly posted onto the Internet [44]. From this leaked data, we can see that FeodalCash is an affiliate program that provides (GPU-capable) Bitcoin mining malware that affiliates install on their bots. In turn, FeodalCash then pays affiliates a fraction of the revenue earned by their bots. This type of labor division enables the affiliates to focus on gaining more bots while the affiliate program can focus on maintaining the malware and infrastructure.

Operation: According to the leaked data, the botnet started operating in May 2013 and there were 238 active affiliates at the time of the data leak. The Bitcoin mining malware was configured to directly mine with the Eligius Bitcoin mining pool and each affiliate was assigned an individual wallet.

Earnings: Since this botnet used Eligius, we can gather a complete profile of

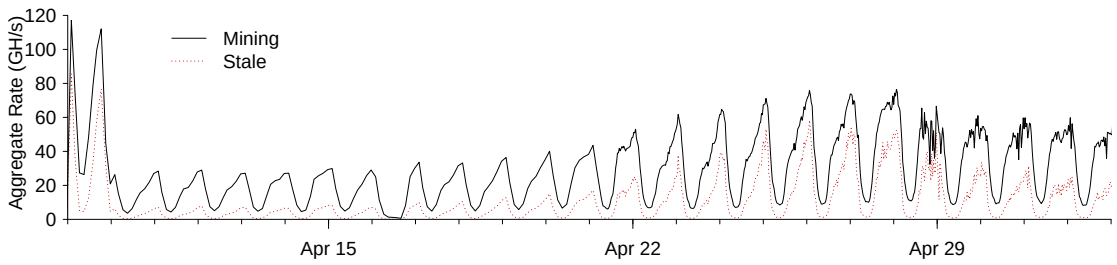


Figure 2.8. Fareit hash rate and stale share rate as reported by the proxy pool server `coonefix.ru`.

their earnings and hash rate over time, as shown in Figures 2.4c and 2.5c. The botnet did not start earning much until more affiliates joined the program around the end of June 2013. At this point the botnet reached a peak of almost 250 GH/s and has since experienced a steady decline in earnings as the difficulty level has increased while its hashing rate has fallen. As of November 2013, the botnet has earned 168 BTC, which translates to approximately 15,941 USD.

Population: If an average PC can mine at about 4 MH/s, we estimate that the bot consisted of 62,500 hosts at its peak hashing rate. In addition, we analyze the diurnal patterns in the hash rate graph. We focus on the hours at which the hashing rate is the lowest every day. As shown in the rightmost portion of Figure 2.6, the botnet reaches minimal activity around midnight UTC. This suggests that the majority of the infected hosts are in Asia.

Transfers to exchanges: As shown in Figure 2.3, the botnet transferred more than 60% of the mining revenue to exchanges. The botnet almost exclusively used WebMoney as the exchange service. The median transfer time is less than five days.

2.5.5 Fareit Bots

The Fareit botnet originally focused on stealing passwords and DDoS attacks. However, on April 9th, 2013 it began distributing Bitcoin mining malware [64].

Distribution: This botnet uses the popular Black Hole exploit kit to install a small executable that contacts `kgtxdu.info` to download an open source Bitcoin mining client called CGMiner[10] onto the victim's system. CGMiner is disguised as a Flash.exe and once downloaded, a Visual Basic script is used to invoke the miner program with a predetermined command line string. The Visual Basic script is then copied onto the Startup directory of a windows system so that the miner will be persistent even when the victim reboots their computer.

Operation: The Bitcoin mining malware contacts a proxy server, `coonefix.ru`, which proxies connections to the public pool `p2pool.org`.[11] It is an example of a smart proxy, as shown in Figure 2.1. The proxy server reports fine-grained data, such as mean payout values, current hashing and stale share rates, which we plot in Figure 2.8. All of this information provides us with deeper insights into the inner workings of their botnet mining operation.

Earnings: To identify the botmaster's wallet address, we look for a wallet that receives payouts from P2Pool, such that the payout rate is consistent with the pool's hash rate, and that the first payout occurred on the same day (April 9th 2013) when the malware started mining. As of November 7th 2013, the Fareit botnet's wallet has received at least 265 BTC of mining revenue. As the global Bitcoin difficulty increases, Fareit has been receiving mining payouts at a slower rate (Figure 2.4d).

Population: We leverage the *stale share* rate to estimate the botnet's population. A share is the proof-of-work that miners submit to the mining pool. The share becomes stale when the another mining pool has mined the block. Whatever work the mining pool, along with its miners, has put in so far is essentially wasted. If a total hash rate of a pool

[10] https://github.com/ckolivas/cgminer

[11] The pool server code on `coonefix.ru` is a fork of the original P2Pool open source software available at https://github.com/forrestv/p2pool

is high, it is less likely that another pool will have mined the block first. The stale share rate will thus be lower.

To estimate the number of infected hosts based on the stale share rate, we perform the following experiment. We mine for P2Pool with a standard desktop computer, which was capable of mining at 4.6 MH/s. We observe that our stale share rate is 24%. Meanwhile, the Fareit proxy reports a stale share rate of 34%. Since the stale share rate goes up as hashing rate goes down[12] , at this point in time the average hashing rate of a bot is less than 4.6 MH/s. If we assume a compromised host mines at 4 MH/s, a low standard deviation in the hashing rate of bot and a total hashing rate of 50 GH/s (a long-term average of the mining rate shown in Figure 2.8), we can estimate there are about 12,500 bots mining in this botnet.

2.5.6 Zenica

Zenica is a botnet that mines at a major public pool. It appears to be operated by one person. Unlike the other major botnets, there are few activity reports of this botnet on anti-virus websites, security blogs or online forums. We are not sure how the malware is distributed or how the botnet operates. However, its sheer size and large earnings merit close scrutiny.

Earnings: We find 67 malware binaries that connected to the mining pool via the username `zenica@gmail.com`. We contacted the pool operator about this user. The operator claimed that the account "had 312,000+ active IPs" and was "paid out about 170 BTC in 3 months."

Population: Zenica bots are most prevalent in Southeast Asia (Table 2.5), with Vietnam and Thailand accounting for over 70% of the sampled infections.

[12]We confirm that a higher hashing rate results in lower stale-share rates by mining with a CPU capable of 18 MH/s and observing a stale-share rate of 14%.

Table 2.5. Distribution of Zenica infections by country.

Country	*Share*	*Est.*
Vietnam	61.0%	119,800
Thailand	9.2%	16,000
Romania	4.3%	5,200
Taiwan	3.2%	5,100
United states	2.3%	3,300
Others	20.0%	32,200

2.5.7 HitmanUK

HitmanUK is a botnet that mines at a major public pool. It has a relatively small mining income: 4 BTC to date. Even so, it makes an interesting case study, because the botmaster launched a DDoS attack on the pool when the pool first blacklisted the botnet.

Operation: We find five malware binaries with the username "hitmanuk." According to the pool operator, the account is associated with the wallet address `1ARHrS`. The binaries and the wallet address were first seen in February 2013. It appears that the botnet has remained active since; the wallet is still receiving mining payouts.

At some point, the pool operator blacklisted the botnet's account, possibly due to reports of malware. The botnet immediately retaliated by launching a DDoS attack on the pool's mining server, paralyzing the entire pool and preventing other users from mining for a few hours. In the end, the pool operator gave in and unfroze HitmanUK's account. This incident suggests that the botnet was — at least at the time — of considerable size.

Earnings: HitmanUK's wallet is active to this day. At the time of this writing, it has received 4 BTC, worth $362 at the time of payout.

Table 2.6. Distribution of infections by country for Xfhp.ru.

Country	*Share*	*Est.*
Indonesia	10.9%	3,200
Mexico	7.3%	1,200
Peru	6.1%	1,900
Thailand	5.5%	1,800
Brazil	4.8%	700
Others	65.5%	28,000

2.5.8 Xfhp.ru Miner

This botnet uses ZBot, also known as Zeus, which connects to `xfhp.ru`. At the time of writing the domain is still active and runs a stratum proxy pool server. ZBot then downloads a plugin that does Bitcoin mining.

Population: Most of the infections for this malware come from Southeast Asia and South American countries, perhaps indicating that the botmaster chose to buy cheaper hosts. Table 2.6 shows the distribution by country and extrapolated population.

Although the estimated infected population of the instance is rather modest, this is another example of a major malware family incorporating Bitcoin mining in addition to other activities.

2.5.9 Skype Miner

We name this botnet "Skype Miner" because at one point it used a combination of Skype and social engineering to distribute the malware. To carry out the attack, the bot sent a Skype Instant message from a compromised Skype account by the name of "Carolina Chapparo" [13]. If the victim clicked on the link in the message, she would be taken to a webpage that contained a drive-by-download exploit pack. The executable would attempt to install the Bitcoin mining malware.

Operation: The initial samples of this malware that was distributed beginning in July 2012 used the same credentials as the version that was distributed via Skype during April 2013. The original malware sample uses `keep.husting4life.biz` as its pool domain and the newer version uses `suppp.cantvenlinea.biz`. Information included in the Stratum headers indicates that both of these domains are proxying connections to the same public pool. In private conversations the pool operators confirmed that this botnet was proxying to their pool.

Earnings: According to the mining pool operators, the user received about 250 BTC. However, they did not provide a wallet for us to confirm these earnings.

2.5.10 Miscellaneous

In addition to the mining operations above, we also find numerous smaller mining operations, many of which mine directly using a fixed set of credentials embedded into the malware binary.

Mining at registration-based public pools: Bitclockers is the only registration-based mining pool that publishes each user's earnings. From malware reports, we extract all usernames that were associated with Bitclockers. We look up all 38 of them in Bitclocker's public records and examine their earnings. After summing up the individual payouts, we find that they have earned close to 30 BTC in total. The biggest earner accumulated 9.6 BTC between November 2012 to January 2013.

In contrast, most major registration-based mining pools do not publish user statistics. We have to manually contact the pool operators, via email or IRC, for user information. One pool operator reports to us a botnet that specifically targeted gamers (we therefore refer to it as Gamers). He provided us with four usernames and their wallet addresses. According to a forum post — purportedly written by an infected user — the

Table 2.7. Miscellanous mining operations.

Worker	*BTC*	*USD*
`ophelion` (Gamers 1)	67.45	4,552.64
`1HUVG8`	65.03	532.35
`1ES11K`	45.59	600.93
`13CnZa`	37.99	494.35
`19zKyp`	37.80	629.74
`18G7T7`	35.23	4,016.90
`1H1xa5`	29.14	357.54
`1PbPiV`	24.74	208.31
`1AfBS5`	24.37	323.08
`1FiPR4`	23.96	163.29
`17F8N9`	19.92	468.02
`1ByFLx`	17.70	208.87
`1AFVcM`	14.54	135.53
`12W29H`	11.51	839.97
`sarajevo`	9.56	119.02
`15p86j`	7.80	923.78
`boywonder`	7.67	103.71
`1a3dpd`	7.03	79.85
`1PwfoA`	6.82	828.91
`process1`	5.39	72.86
`17pdMw`	5.37	326.07
`15LuUP`	4.85	58.09
`archy10`	4.10	48.57
`1PyoNm`	2.08	250.61
`1Kjvxd`	2.03	25.17
`ridetohell` (Gamers 2)	0.55	50.57
Others	21.58	798.14
Total	539.24	17,166.30

malware disguised itself as a game executable, which connected to the mining pool via one of the four wallet addresses.[13]

We analyze the mining payouts for the primary and secondary wallets for the Gamers botnet and present their earnings in Figure 2.4e. It shows that the botnet first became active in January 2013. Mining activities have waned since mid-June, possibly after a crackdown by the pool's operator or anti-virus companies.

We are able to trace how two of the botnet's wallets transferred the mining revenue to exchanges, as shown in Figure 2.3. The first wallet (Gamers 1) took a median of three weeks before transferring more than 90% of the mining revenue. The second wallet (Gamers 2), by contrast, transferred a little more than 55%. Both of the transfers happened at the Bitstamp exchange.

The first wallet was also associated with Eligius, another public mining pool. Its hash rate graph displays a typical diurnal pattern that is strongly suggestive of botnet activity. Moreover, the average hash rate is around 4 GH/s in the last two months, with a total of 70 BTC paid out by Eligius. Assuming that an infected host can range from an average CPU-only computer (4 MH/s) to a typical gamer's PC (50 MH/s), we can estimate the size of the botnet as somewhere between 80 to 1,000 infected computers.

In addition to the Gamers botnet, the pool operator also gave us four more malware wallet addresses. We do not know their mode of operation. The four wallet addresses alone have only earned 7.7 BTC from mining since December 2011. However, they are associated with more than 40,000 secondary wallet addresses. We believe that not all of them are involved in receiving mining payouts. One common practice is to have a small number of wallets for mining, while the rest are used for "mixers" — services that attempt to obfuscate the trail of transactions before cashing out, making it difficult, but not impossible, to trace the transactions. Using the techniques in [47], we identify

[13] https://bitcointalk.org/index.php?topic=159307.0

the transactions for mining payouts. We find that both the primary and secondary wallets have received 886 BTC of mining revenue.

Bots for no-registration public pools: We find four additional wallet addresses that mining malware uses to connect to 50 BTC, a public pool that supports both conventional registration-based and no-registration mining. Since receiving their first mining payouts in December 2012, these four addresses have only received 2.6 BTC from mining. If we are to examine all the secondary wallets — all 24 of them — the total revenue from mining amounts to 242 BTC.

At Eligius, we find 29 wallets that do not belong to any of the major botnet operations we study. These wallets alone have yielded an income of 332 BTC from mining since their initial mining payouts in March 2012. They are associated with more than 600,000 secondary wallet addresses. Again, we believe only a small fraction is directly involved in mining. Even so, the total mining revenue for these secondary addresses amounts to more than 30,000 BTC. Some major botnet operations may be behind this, but we leave it to other researchers to analyze.

Bots for proxies to light pools: Recall that we identify `domain-crawlers.com`, a dark pool, as a proxy to 50 BTC, as described in Section 2.4. We find a total of three usernames associated with mining malware at `domain-crawlers.com`. The operators of 50 BTC confirm them as pool users, but tell us only that the accounts have a total balance of 0.1 BTC. This small amount suggests that the botnet may have already cashed out their mining earnings, but the exact revenue remains a mystery.

2.6 Discussion

Bitcoin mining, as evidenced by the operations we examine, can generate non-trivial revenue for a botnet operator (see Tables 2.1 and 2.7). Still, these numbers are

nothing like the spectacular earnings — millions of US dollars — estimated for spamming and click fraud [41]. Bitcoin mining as a botnet monetization activity is ultimately judged by its profitability, that is, the expected revenue from Bitcoin mining minus costs.

Mining revenue: Mining revenue — whether from a botnet or a legitimate mining operation — depends on two factors: hashing power and network difficulty. For revenue measured in US dollars, the BTC-to-USD exchange rate is a factor as well. Daily revenue is thus given by:

$$\frac{\text{USD}}{\text{day}} = \frac{\text{sec}}{\text{day}} \cdot \frac{\text{MH}}{\text{sec}} \cdot \frac{\text{BTC}}{\text{MH}} \cdot \frac{\text{USD}}{\text{BTC}}.$$

Here BTC/MH is the expected revenue, in bitcoins, per million SHA-256 computations. At the difficulty level as of November 30, 2013, this is 8.22×10^{-12}MH/sec.

Denote $D = \text{BTC}/\text{MH}$ for short. Denote the exchange rate $U = \text{USD}/\text{BTC}$, which was slightly over $1,100 per bitcoin on November 30, 2013. Let R be aggregate hash rate in million hashes per second; $R = \text{MH}/\text{s}$. A low-end PC without a GPU is capable of about 4 MH/s, a newer PC without a GPU of about 20 MH/s, and a top of the line AMD Radeon 7970 GPU is capable of about 500 MH/s.

A botmaster's revenue per bot per day is thus given by

$$\text{USD Daily Revenue} = 86400 \cdot R \cdot D \cdot U. \tag{2.1}$$

At the exchange rate and difficulty as of November 2013, this comes to $\$0.00078 \cdot R$. With $D = 8.22 \times 10^{-12}$ and $U = \$1,100$ as above, a low-end PC generates about 0.3 cents per day; a PC with a discrete GPU capable of 100 MH/s can generate about 7.8 cents per day. A network of 10,000 low-end PCs would generate about $31 per day; if at least one in ten have a discrete GPU capable of 100 MH/s, it would generate another $75

Figure 2.9. Daily miner revenue per MH/s of mining capability. Daily revenue per MH/sec of hashing power is given by $86400 \cdot D \cdot U$, where D is the expected revenue in BTC per million hashes computed, and U is the USD:BTC exchange rate.

per day.

Figure 2.9 shows the daily revenue in USD per MH/s of hashing power as a function of time. That is, the graph plots Eq. 2.1 with parameter $R = 1$ and parameters D and U varying with time. Revenue per unit of hashing power is at an all-time low — nearly an order of magnitude lower than the previous lows in October 2011 and December 2012 (when the block mining reward halved to 25 BTC).

Botnet costs: We can divide the costs into the cost of acquiring the bots and the cost associated with the monetization scheme itself. Compromised PCs in Asia cost $5 to $10 per thousand, as reported by Caballero *et al.* [22], which agrees with our own informal survey of such services. (We note that the wholesale price may be much lower.) The cost of a bot is amortized over its lifetime. Unfortunately, we are not aware of reliable estimates of how long a bot remains infected. (The spikes of activity in Figure 2.2a suggest the median lifetime is on the order of a week.)

Much less still is known about the non-acquisition costs. These include the cost

of infrastructure, development, and day-to-day operations. Neglecting non-acquisition costs, if we estimate that a low-end bot costs 0.2–1 cent to acquire and and can generate 0.2–1 cent per day from mining, then the time to break even ranges anywhere between 1 day and 25 days (operating continuously).

Profitability: The volatility of the BTC-to-USD exchange rate, increasing Bitcoin network difficulty, variance in PC hashing power, and unknown botnet acquisition and operating costs make it difficult to accurately estimate the profitability of Bitcoin mining. Setting the question of determining profitability aside, let us examine the possible outcomes qualitatively. At any given time, the profitability Bitcoin mining, or more generally, any botnet monetization activity, falls into one of three classes:

Absolutely profitable. The revenue from Bitcoin mining exceeds the costs of operating a botnet solely for mining.

Marginally profitable. For an existing botnet, the additional mining revenue exceeds the additional costs associated with Bitcoin mining.

Unprofitable. Mining revenue does not cover the additional costs of Bitcoin mining.

Throughout most of 2012 and the first quarter of 2013, Bitcoin mining could generate over 1 cent per day from a low-end PC, so that even at retail pricing for bots, Bitcoin mining was an absolutely profitable botnet monetization activity. At least some of the operations dating back to that period, such as the DLoad.asia family, appear to be known only for their mining (of course, we cannot exclude other activities with absolute certainty).

Where we are today is less clear. We observe, however, that the marginal cost of Bitcoin mining, that is, the cost of Bitcoin mining on a botnet already engaged in another activity, is very low. Bitcoin mining does not interfere with other activities such as spamming or click fraud, since it exploits a previously untapped resource: computation.

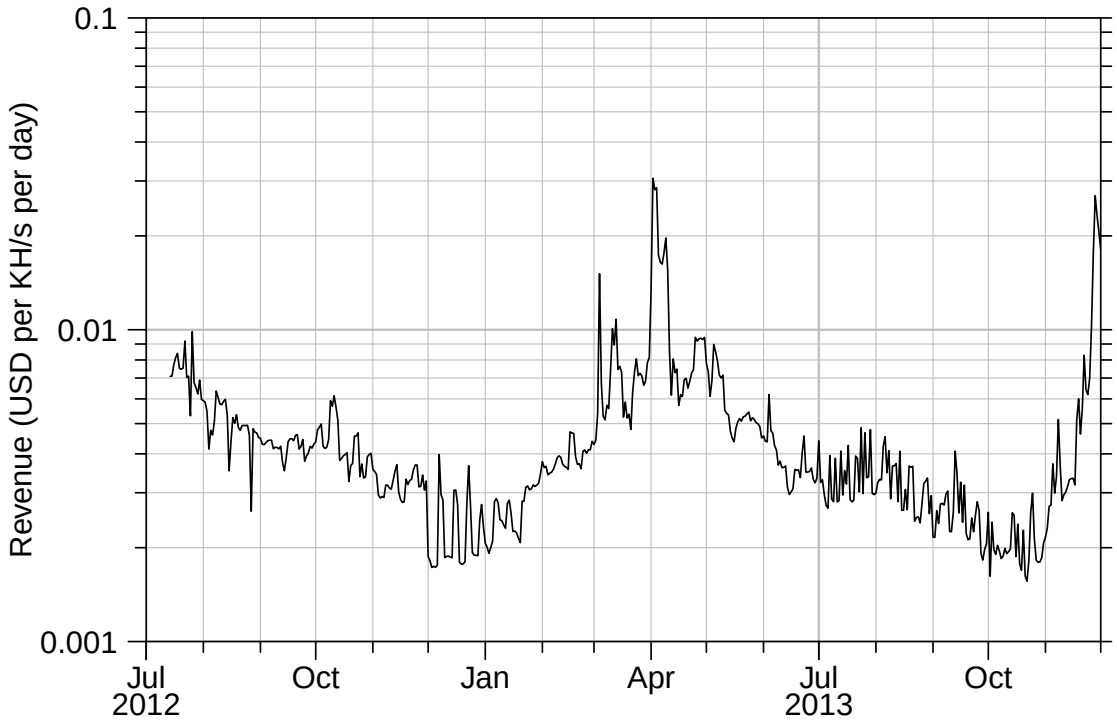


Figure 2.10. Daily revenue per KH/s of Litecoin mining capability.

With acquisition and infrastructure costs paid for by another activity, the remaining costs are software development and additional management overhead associated with illicit Bitcoin mining. Because these costs can benefit from economies of scale, we expect Bitcoin mining to remain at least marginally profitable for large botnets.

2.7 Summary

In this chapter, we used the blockchain to understand bitcoin-mining botnets. It was a new phenomenon where botnets monetized stolen computational resources by mining bitcoins without paying for the energy costs. By analyzing mining malware and tracking bitcoin movements on the blockchain, we showed that botnets generated at least $118,000 of revenue, in addition to existing monetization schemes in their infrastructure. For some botnets, the energy costs to the infected victims were at least twice the botnets' revenue. However, as specialized mining hardware gained increasing popularity, the

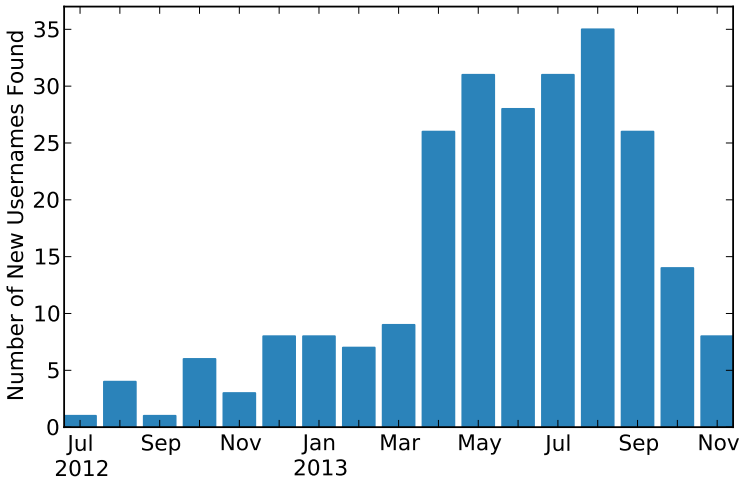


Figure 2.11. Monthly new usernames for Litecoin-mining malware.

revenue per unit of computation diminished over time, as we found by analyzing the level of difficulty in the blockchain.

While bitcoin-mining was unlikely to be profitable for botnets, botnets could choose other crypto-currencies for mining to monetize the computational resources, such as *Litecoin*. Litecoin is another decentralized virtual currency based on Bitcoin code. The only significant changes for Litecoin are a difficulty parameter that produces blocks four times faster and replacing the SHA-256 proof-of-work function with Scrypt, which has a higher performance on CPU rather than GPU/ASIC miners. The release of Litecoin gave new opportunities for botnets. From a botmaster's point of view, Litecoin mining is the same as Bitcoin mining, differing only in the executable, mining pools, and profitability. The BMControl bot has already begun mining litecoins, and contacting the pool server operator indicates that the botnet has received 453 LTC (about $900) between April and August 2013. Another botnet reported online received 6700 LTC (present value about $13,000), mostly between December 2012 and March 2013.

The interest can be observed in other ways. Figure 2.11 shows the number of new usernames for Litecoin-mining malware discovered in the Emerging Threats

database every month, where the first Litecoin-mining malware was found in July 2012. Beginning in April 2013, its clear that the significant uptick in Litecoin-mining malware suggests increased botnet interest in Litecoin. Figure 2.10 is the recent Litecoin analog to Figure 2.9, showing the revenue per day per KH/s for a Litecoin miner. In contrast to Bitcoin, which witnessed a collapse in revenue due to the influx of ASIC miners, the revenue per KH/s for Litecoin has experienced fewer drastic fluctuations.

In addition to Litecoin, even today, there is anecdotal evidence of botnets mining other crypto-currencies, such as Monero [32]. We argued that the actual choice of crypto-currency did not matter. As long as there existed available computational resources, miscreants could monetize them via mining.

To conclude this chapter, botnets are not the only ones mining Bitcoin, Litecoin, and similar crypto-currencies. In fact, there are more than 1,400 such crypto-currencies. They are typically used as investment vehicles in mining and speculation. We will show how to track economic activities involving crypto-currency investment in the next chapter.

Chapter 2, in full, is a reprint of the material as it appears in the Proceedings of Network and Distributed System Security Symposium, 2014. Huang, Danny Yuxing; Dharmdasani, Hitesh; Meiklejohn, Sarah; Dave, Vacha; Grier, Chris; McCoy, Damon; Savage, Stefan; Weaver, Nicholas; Snoeren, Alex C.; Levchenko, Kirill. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Estimating the Profitability of Crypto-currency Investors

The previous chapter alluded to botnets' use of crypto-currencies, other than Bitcoin, for mining. In fact, there are more than 1,400 such crypto-currencies that are alternatives to Bitcoin. Known as *altcoins*, most of them share the same code base as Bitcoin, with a few key difference in blockchain parameters, such as the number of blocks per hour. Unlike Bitcoin, which is accepted by merchants online and in the darkweb, few altcoins are known to be used in commerce. Rather, most of them are used as investment vehicles, with a market capitalization ranging from hundreds to millions of dollars.

In this chapter, we use the blockchain and trade datasets of 18 altcoins to measure the potential profitability of altcoin investment. By developing a new technique to estimate the mining cost, we show the wide range of sunk cost that miners have invested across various coins. Furthermore, we simulate altcoin investment — mining or buying/selling coins — to estimate the potential profitability of various investment strategies. We show retrospectively that the altcoin market, in general, is highly risky, as the potential profitability of an altcoin can drastically change depending on the investment strategy.

3.1 Introduction

In Chapter 2, we showed that as the revenue from bitcoin-mining decreased, there was increasing evidence of botnets mining crypto-currencies, such as Litecoin, that were alternatives to Bitcoin. Today, there are over 1,400 of such currencies. Also known as *altcoins*, most of these crypto-currencies share the same code base as Bitcoin, which itself is open source. The main differences are key blockchain parameters, such as the number of blocks per day, the amount of miner reward per block, or the hash function used for mining. For instance, whereas Bitcoin generates a new block approximately every 10 minutes, Litecoin (LTC) does so every 2.5 minutes; whereas Bitcoin uses the SHA-256 hash function, LTC uses Scrypt [57].

In addition to technical differences, most altcoins are different from Bitcoin in usage. In contrast to Bitcoin, which can be used for commerce both online and on the darkweb [47], few altcoins are known to be accepted by merchants. In fact, most altcoins are used for speculative investment. Some altcoins, such as Auroracoin (AUR) and Dogecoin (DOGE), are worth millions of dollars in market capitalization. This high market capitalization suggests that despite the lack of inherent value in most of these coins (e.g. inability to be used in commerce) investors actually assign high value to the coins.

However, investment in altcoins can be risky, as altcoin prices are highly volatile. For coins with high market capitalization, high volatility may cause some investors to gain or lose significant amounts of money. By following the movement of altcoins, analyzing the blockchain, and examining the trade data, we strive to estimate the gains and losses for different types of investors, and to retrospectively identify the most profitable investment strategies.

To this end, we first identify the stakeholders in the altcoin economy. An investor

can enter the market by either mining a coin or buying a coin. His actions are a mixture of mining and speculating. We decouple mining from pure currency speculation by identifying two logically separate roles: *miners*, who mine a particular altcoin and immediately sell it; and *speculators*, who buy a particular altcoin and sell it later. In reality, an investor can be both a miner and speculator, but both are mutually exclusive in our model. Thus, an investor chooses to become a miner or a speculator at a particular time for a particular coin. Each choice would result in a different profit, as the cost of mining and the price of coins constantly change. Our goal is to retrospectively measure the profit for these choices.

To estimate the profit, we need both the cost and the revenue. While the cost for speculators is simply the market price of the coin purchased, the cost for miners is difficult to estimate. A miner can buy or rent different mining hardware; he can even steal computational cycles, as illustrated in Chapter 2. All these options are associated with different costs. To sidestep the issue, we introduce a method to estimate cost of mining across all miners of a particular coin, based on the opportunity cost of mining a different coin, such as Bitcoin or LTC, that have a much larger market capitalization. For example, mining $1 worth of Dogecoin on January 1, 2014 required more than 7 billion hash computations on average; the same amount of effort could be spent mining Litecoin, which uses the same hash algorithm as Dogecoin, for an expected revenue of $0.61. Regardless of the costs associated with operating the mining hardware, on that day, Dogecoin offered an additional $0.39 of revenue compared with Litecoin, for doing the same 7 billion hash computations. By mining Dogecoin, the miner has given up the opportunity to earn Litecoin; thus, the opportunity cost of mining Dogecoin relative to Litecoin $0.61. We use this opportunity cost to study the relative profitability of each coin as viewed by a rational profit-oriented miner with perfect information.

To estimate the revenue, we use simulation to sidestep the challenge of identifying

the behaviors of individual miners/speculators in reality. In this way, our combined use of the blockchain and trading data reveals what happens in the altcoin economy: the potential profitability of different investment strategies with mining and speculating.

In summary, the contributions of this work are as follows. First, we develop a methodology for estimating a miner's investment in an altcoin by calculating her opportunity cost of mining a more stable alternative like Bitcoin or Litecoin. Second, we compare the profitability of mining and speculation of different coins through simulations under varying conditions. Third, we find that miners who mine an altcoin immediately after it is listed on exchanges tend to enjoy higher potential returns than miners who mine on subsequent days. In contrast, speculators who buy an altcoin shortly after it is listed on exchanges are likely to generate lower returns than speculators who buy at a later point, and they also generate lower returns than miners in the same period.

The rest of this chapter is organized as follows. Section 3.3 describes our dataset, which includes both the blockchain and trade data. Section 3.4 explains our methodology of computing the opportunity cost of mining, as well as using simulations to estimate the profit of mining and speculation. Using this methodology, in Section 3.5 we estimate the opportunity cost of mining for the 18 altcoins in our dataset, followed by Section 3.6, where we use the opportunity costs and market prices to estimate the profitability of mining and speculating. We discuss two altcoins in detail in Section 3.7 to highlight common observations across altcoins, along with exceptions to such patterns. Section 3.8 discusses limitations to our methodology. Finally, we conclude our chapter in Section 3.9.

3.2 Background on Altcoins

Based on data we collected from exchanges, at least 1,400 altcoins were mined and traded after Bitcoin was released in 2008. Others, like Ripple [5] and Ethereum [4], were developed completely independently. The vast majority, however, are adapted from

Bitcoin, with a variety of minor differences. For example, the developers of LTC were dissatisfied with Bitcoin's average block rate of 10 minutes; thus, LTC has a block rate of 2.5 minutes in an apparent effort to speed up transaction processing. Furthermore, BTC uses SHA-256 as the proof-of-work (PoW) function. As Bitcoin miners were increasingly using dedicated hardware like ASICs to mine bitcoins, LTC used Scrypt as PoW, in an apparent attempt to discourage the use of ASICs in mining. [57] In short, these "cloned" coins, like LTC, form the basis of our study, as their shared structure admits straightforward comparisons.

We consider two logical participants in an altcoin ecosystem—although any single person or enterprise may play either or both roles: miners, who provide the computational resources to generate altcoins by mining; and speculators, who simply buy and sell altcoins on the open market. A typical coin's lifetime starts when the developer releases the coin's client to the public. A miner downloads the client, connects to the coin's peer-to-peer network, and mines the coin with CPU, GPU, or dedicated hardware. Once there is sufficient mining power toward the coin, or once there is enough interest in the community, an exchange may decide to list the coin, in a process similar to initial public offerings. At this point, a market for the coin exists. Miners can sell their coins, and speculators can start buying the coins.

Because we want to measure the profitability of miners and speculators, we need a way to quantify the costs and revenues of each. Our speculation analysis relies on trading data. To analyze mining revenue, we compare one coin against a more popular base currency with the same technical attributes. Here, two attributes of a coin are important:

Types of proof: Each coin defines a mechanism to allow a miner to prove that they have successfully mined a block. Bitcoin, for instance, uses PoW. Recall, from Section 1.3, that to generate this proof, miners must find partial hash collisions via brute

force. Miners with higher computational power are more likely to be rewarded. To compensate for the time-varying hashing capabilities of the active miners, PoW coins constantly adjust their difficulty, which dictates the expected number of hashes to mine a block, in an attempt to keep the rate at which coins are mined constant. The cost of PoW mining, therefore, is a time-varying function unique to each miner, dependent on that particular miner's ability to acquire and operate the requisite computational resources, often referred to as the mining gear.

In contrast, altcoins employing *proof-of-stake* (PoS) reward miners based on the number of the coins they already possess. Typically, a miner computes a very small number of hashes on the order of every second. For the purpose of this chapter, the computational resources required for PoS mining are effectively zero. For a typical PoS coin, the first few blocks are mined using PoW to bootstrap the currency, and subsequent blocks are mined with PoS, or they alternate between PoS and PoW.

Lastly, a coin may also be mined using *auxiliary proof-of-work* (AuxPoW). Also known as merged-mining, this approach allows a coin to be generated alongside a *parent coin* (which are typically popular coins such as Bitcoin or Litecoin), while the miner computes hashes only for the parent coin. In this way, miners need not dedicate their mining power to a particular AuxPoW coin; they can mine the AuxPoW for free while generating the parent coin. A typical AuxPoW coin starts with only PoW blocks in the blockchain, and it eventually switches to AuxPoW blocks.

Hash functions: Mining a PoW or AuxPoW block requires searching for collisions in some prefix of a fixed hash function's output. (The hash function is selected by the coin's developer.) For Bitcoin, that hash function is defined to be SHA-256 for all blocks, as described in Section 1.3; for Litecoin, it is Scrypt [56]. For most altcoins, each block is similarly mined using the same hash function, either SHA-256, Scrypt, or

Table 3.1. Overview of the coins that we study.

	(a) Overall Statistics								(b) Opportunity Cost Analysis			
Coin	*Blockchain (Days)*	*Trade (Days)*	*Market Cap ($)*	***Trade Vol ($)***	V_1 *(%)*	V_7 *(%)*	*HF*	*Pf*	*Anlys (Days)*	*Market Cap ($)*	*Opp Cost ($)*	*Corr*
BTC	2,711	2,128	9,129,946,498	30,750,001,190	0.00	0.81	SH	W	2,128	8,112,097,758	1,764,083,690	1.00
LTC	1,691	1,122	182,338,574	1,542,156,841	-0.16	-0.96	Sc	W	1,122	106,266,194	179,931,265	1.00
DOGE	919	915	26,938,503	236,217,401	-0.52	-2.48	Sc	A	276	17,476,815	43,769,744	0.45
PPC	1,375	1,045	8,511,846	187,487,304	-0.39	-1.76	SH	S	1,375	8,129,715	4,137,808	0.98
AUR	852	825	2,625,706	20,455,696	-0.95	-4.73	C	W	827	2,828,901	1,357,342	0.87
DGC	1,101	1,045	270,634	4,756,876	-0.50	-4.01	C	W	563	171,734	831,855	0.96
VIA	678	677	125,390	4,426,677	-1.16	-4.84	Sc	A	160	403,478	240,058	0.94
UNO	950	888	371,620	4,204,613	-0.15	-1.63	SH	A	566	491,855	436,511	0.62
RPC	878	686	5,561	938,403	-1.90	-6.96	Sc	W	691	4,323	276,416	0.93
ARG	1,079	858	8,630	733,145	-1.12	-5.61	Sc	W	892	7,540	91,499	0.90
EFL	797	796	152,200	372,766	-0.11	-1.04	Sc	W	797	162,355	77,047	0.91
WBB	464	454	192,722	317,571	-1.83	-5.74	Sc	W	464	164,643	61,213	0.86
CURE	754	744	436,351	250,942	-0.61	-4.60	SH	S	754	421,326	37,761	0.73
XJO	976	701	4,368	183,473	-0.85	-6.25	SH	W	789	3,519	114,994	0.97
BTA	383	379	32,229	30,163	-0.84	-3.38	Sc	W	383	31,293	8,020	0.93
HAM	694	497	14,166	25,042	-1.20	0.59	SH	S	575	12,698	5,575	0.70
SWING	269	230	3,091	23,354	-1.55	-5.40	SH	S	269	4,102	3,441	0.84
TROLL	177	57	22,144	2,639	0.05	0.85	Sc	S	58	24,776	280	0.70
VCN	378	246	3,390	982	1.61	7.38	SH	W	259	4,316	852	0.82
DOT	775	331	17,656	917	-1.31	-5.46	Sc	W	346	4,974	3,198	0.78

some other hash functions. Some altcoins, however, such as Auroracoin (AUR), allow a portion of their blocks to be mined with one hash function, and another portion of their blocks to be mined with a different hash function [1].

3.3 Data sets

In this section, we describe our blockchain and trade datasets. We explain how we collected the data, and we provide an overview of both dataset.

3.3.1 Blockchain

We obtained the blockchain data for 153 altcoins from the CryptoID website, which hosts and displays blockchain data in a way similar to blockchain.info [3]. Additionally, we also obtained the blockchains of Litecoin (LTC) and Bitcoin (BTC) by running the clients ourselves. By checking against altcoin mining pools such as ZPool and PoolSwitch, we were able to determine the hash algorithms and types of proofs for

every block in 42 of the coins. We have also manually sampled blocks from CryptoID, ZPool and PoolSwitch and confirmed that they are consistent with one another. Out of these 42 coins, 18 coins, along with BTC and LTC, exclusively use SHA-256 or Scrypt as the hash function in parts of their blockchains. As we can compare these coins against BTC (based on SHA-256) and LTC (based on Scrypt), our study focuses on the 18 altcoins.

Table 3.1(a) summarizes the coins by hash functions (Column "HF") and types of proofs (Column "Pf") as of November 8, 2016. For hash functions, "SH" is a shorthand for SHA-256, "Sc" for Scrypt, and "C" for Combined, where some blocks are mined with SHA-256, and some blocks are mined with Scrypt (e.g. AUR). For types of proofs, "W" is a shorthand for PoW, "S" for PoS, and "A" for AuxPoW. We note that any of these properties may change at any time. For example, BTC has consistently been a PoW coin that is mined with SHA-256. In contrast, for the first 827 days in the 852-day blockchain in our dataset, Auroracoin (AUR) is a proof-of-work coin that could be mined with Scrypt. Afterwards, blocks can be mined with SHA-256, Scrypt, or some other hash functions.

3.3.2 Trade

We obtained daily trade data of the 20 coins (18 altcoins plus LTC and BTC) from CryptoCoinCharts, which aggregates daily altcoin trades across 1,436 altcoins over 57 exchanges since 2010 [2], along with blockchain.info, which records daily BTC prices since 2011. We also scraped a well-known exchange, Cryptsy (now defunct), to verify altcoin prices on CryptoCoinCharts were the same as those on Cryptsy. The 57 exchanges include some of the most well-known altcoin exchanges like Cryptsy and Poloniex, and we believe that they have processed a representative sample of all altcoin trades, although we make no claims of completeness. While our volume data is therefore a lower bound on

actual trade volume, we assume that any markets we do not capture offer roughly similar prices on a day-to-day basis. For each coin, our dataset includes the exchange(s) where the coin was listed, along with the daily trade volumes at each exchange. For each altcoin c, we compute the mean price on a given day t as $p(c,t) = v(BTC,t)/v(c,t) * p(BTC,t)$, where $v(BTC,t)$ is the trade volume of BTC against c at t (as altcoins are typically traded against Bitcoin), $v(c,t)$ is the trade volume of c against BTC at t, and $p(BTC,t)$ is Bitcoin's daily USD price at t.

Table 3.1(a) presents trade-related statistics for the 18 altcoins we study, along with BTC and LTC. In particular, we show the lengths of trading activities in the "Trade" column, which counts the number of days from when a coin is first listed at an exchange to when it is last traded at some exchange. We contrast this value with the lengths of blockchains ("Blockchain" column) — the number of days from when the first block was mined to the last block in the data. In all cases, a coin is listed some time after the first block is mined.

The market capitalization, shown in the "Market Cap" column, reflects the market value of all units of an altcoin as of the last day of that coin in our dataset [9]. We calculate it as $(p \cdot q)$, where q is the total number of coin units ever mined up to the last day in our dataset (which differs across altcoins), and p is the USD price of the altcoin as of the last day the coin was listed in our dataset. In comparison, to compute the trade volume of an altcoin, we sum up the total number of bitcoins traded against the altcoin every day. Using daily BTC-USD exchange rates, we calculate the equivalent US dollar value for the bitcoins. Summing up these US dollars, we obtain the "Trade Vol" column. Typically, a high trade volume is strongly correlated with a large market capitalization, with the Pearson correlation coefficient of 0.9995. The total amount of trade for all 1,436 altcoins in the CryptoCoinCharts dataset is $38.5 billion. Bitcoin accounts for the majority of the trade with a volume of $30.8 billion. Of the remaining $7.7 billion of

trade, the 18 altcoins we study, plus LTC, account for $2.0 billion of trade volume.

One feature of altcoin markets is the price volatility. To compare volatility across different altcoins, we examine the percentage difference between daily prices over time. We define volatility V_d as the median percentage difference between prices on Day i and Day $(i+d-1)$ for all possible i values within the trading period. A larger absolute value of V_d indicates a higher level of volatility. For instance, the Euro/USD exchange rate is the past year is associated with $V_1 = V_7 = 0.00\%$ [59]. In the same period, Google's stock price (GOOG) has $V_1 = 0.08\%$ and $V_7 = 0.67\%$ [67]. For altcoins, we present the V_d values in Columns "V_1" and "V_7" in **Table 3.1**(a). BTC and LTC, for example, have amongst the lowest absolute values for V_1 and V_7.

3.4 Methodology

Our goal is to estimate the potential profitability of mining and speculating across different altcoins. Using the data we collected in Section 3.3, we discuss a methodology to estimate the cost of mining (Section 3.4.1) with the concept of opportunity costs. Using the cost estimate, we describe a method to estimate the profitability of mining, as well as speculating in Section 3.4.2.

3.4.1 Estimating Cost of Mining

For miners to decide whether to mine a particular coin through PoW, they first need to estimate the cost of mining. However, the precise value is difficult to calculate. The fixed cost of mining, such as investing in mining equipment, can vary across coins, depending which hash functions coins use. For instance, while coins based on SHA-256 are best mined with custom ASICs, hash functions such as X11 are designed to be ASIC-resistant and should be computed on general CPUs or GPUs [27]. We also need to account for variable costs, such as the cost of upgrading to faster mining equipment,

as well as the (significant) energy costs of operating the gear that differ widely across geographical regions.

An alternative to using direct cost is the *opportunity cost*. The opportunity cost of an activity is the revenue lost by engaging that activity *rather than the best alternative*. We can apply the idea to altcoin-mining. In particular, for two coins based on the same hash function, the costs of mining one versus the other are nearly equal, because the underlying unit of work, computing a hash, is the same. Thus, other than some initial software set-up cost, computing a million hashes of SHA-256 toward Bitcoin costs the same as computing a million hashes of SHA-256 toward XJO, another SHA-256-based currency.

Thus, the opportunity cost of mining XJO is the revenue a miner can expect from mining another SHA-256 currency *instead* of XJO over the same time period. To be a meaningful concept for the miner, this alternative revenue should be something a miner can reasonably expect to receive *a priori*. In other words, to say that a miner chose to mine A units of XJO rather than receive D US Dollars for mining another currency, the miner must be certain that he could get D US Dollars by choosing the alternative *before* choosing one or the other. In our comparisons, we use the least volatile alternative currencies with the highest trade volumes. For SHA-256-based coins, this is Bitcoin; for Scrypt-based coins, this is Litecoin. We call currency whose opportunity cost we are computing (e.g. XJO) the *target currency*, and Bitcoin or Litecoin the *base currency*.

Formally, we define the opportunity cost of mining a unit of a currency on a given day as follows. First, we determine the expected number of hashes, H, necessary to mine a unit of the target currency based on the difficulty of mining the currency that day. Next, we determine the expected number of units of Bitcoin (for SHA-256 currencies) or Litecoin (for Scrypt currency) that could be mined with H hashes. Finally, we convert this expected number of bitcoins or litecoins to US Dollars at the exchange rate that day.

Thus, the opportunity cost of mining a unit of currency X is $\text{OppCost}_X = D_X \cdot R_B / D_B$, where D_X is the expected number of hashes required to mine a unit of X based on the day's difficulty, D_B is the expected number of hashes required to mine a unit of the base currency (Bitcoin or Litecoin) based on the day's difficulty, and R_B is the exchange rate of the base currency, in US Dollars per unit of the currency, on that day. We compute D_X and D_B based on the blockchain data of the target and base currencies, while we obtain R_B from our trade data.

An alternative way to understand opportunity cost is to consider an investor who would like to mine \$100 worth of currency X, say one based on SHA-256. He can approach a miner mining Bitcoin and offer him \$100, plus a nominal fee, to spend the same number of hashes it would take to mine \$100 worth of Bitcoin to mine X instead. The miner gets \$100 plus a fee, the investor gets $\$100/\text{OppCost}_X$ units of X. In an efficient market, the fee paid to the miner is negligible, so OppCost_X is the cost of acquiring a unit of X by mining. Even though such an efficient market for hashing power may not exist today, opportunity cost is a powerful *analysis tool* for comparing the relative profitability of mining a currency versus buying it outright.

3.4.2 Estimating Profitability

Our goal is to estimate the profitability of both miners and speculators for different coins. However, computing the profitability of individual actors is difficult. First, it is hard to identify the miners based on wallet addresses on the blockchain. Moreover, even if we did so, our trade data is anonymous and independent of the blockchain. More vexingly, we do not even have lot information: we do not know which coin units were sold when. The trade data only presents daily aggregates of how many coin units were bought and sold, and at what price. Instead, we focus on the potential profitability of a dollar invested, either through mining (i.e., \$1 of opportunity cost) or speculating (e.g.,

literally spending $1 to buy into the market of a given altcoin).

We separate the act of altcoin investment into two logical roles: (a) the creation of the coin units (i.e. mining), and (b) continuing to hold the coin units and selling at the opportune moment (i.e. speculation). Specifically, we restrict mining to the act of dedicating computational resources to the coin's blockchain to obtain a reward, and then presume the miner sells immediately (if possible). Hence, any gain or loss a miner experiences is due entirely to intra-day arbitrage between the valuation of the currency he chooses to mine and the alternative base currency he could have mined instead (i.e. opportunity cost). The mined coin units are logically transferred from miners to speculators, who would then hold the units and sell them at the opportune moment (e.g. when the price rises).

We estimate the profitability of miners through simulation. Using historical blockchain and trade data, we simulate the investment of $1 worth of opportunity cost in mining across different altcoins, start dates, and durations. For speculators, we use a similar simulation, varying the time when an investor enters the market by buying $1 worth of an altcoin's units, as well as the holding positions of the investor. In this way, we can compute the profitability of mining/speculating depending on the participant's strategy. This profitability analysis, albeit retrospective, assesses the risks and rewards for each coin and across multiple coins.

To simplify our analysis, we assume that any action that a miner or speculator takes is sufficiently insignificant so that it is unable to affect the market price. Specifically, a miner spends $1 of opportunity cost on mining an altcoin and sells the mined units on the same day. Similarly, each speculator purchases a $1 worth of a given altcoin and sells all units some time later. An altcoin market typically has a trade volume much higher than $1, such that $1 of mining or speculating is unlikely to change the price significantly.

3.5 Estimating Cost of Mining

In this section, we approximate the cost of PoW mining by computing the opportunity cost. We are aware that some coins start as simple SHA-256/Scrypt PoW crypto-currencies, but later they change the hash functions or types of proof. To this end, we consider the history of the currency up to the day of change, so that we can use BTC or LTC as the base currency for calculating opportunity costs.

Table 3.1(b) presents the opportunity cost of mining. For each currency, we show the the length of the blockchain — since the beginning of the chain — for which mining involves PoW blocks based on SHA-256 or Scrypt; this length is our analysis period ("Anlys" column). For coins like WBB and EFL, the values in the "Anlys" column are the same as the actual lengths of the respective blockchains, because the entire blockchain accepts PoW mining with SHA-256/Scrypt. In contrast, DOGE blocks are based on Scrypt in the first 276 days; afterwards, DOGE allows AuxPoW mining. As a result, we only consider the first 276 days of the 919-day blockchain for DOGE.

We include PoS coins in our opportunity cost analysis as long as they also allow PoW mining. For every PoS currency we consider in **Table 3.1**(b), the majority of the altcoin units were mined with PoW. For instance, only 1.5% of PPC units were mined with PoS, and 0.3% of CURE units were with PoS. As the computational cost of mining PoS coins is negligible, we effectively consider these altcoins as PoW and compute their opportunity cost accordingly.

Focusing on the analysis period, we compute the total opportunity cost of mining ("Opp Cost" column). The total opportunity cost reflects the amount of sunk cost into a given altcoin by all the miners.[1] For example, DOGE miners would have potentially made more than $43.8 million if they had been mining LTC instead. We contrast the total

[1]The opportunity costs of mining BTC and LTC are simply the revenue of selling mined BTC/LTC on the same day.

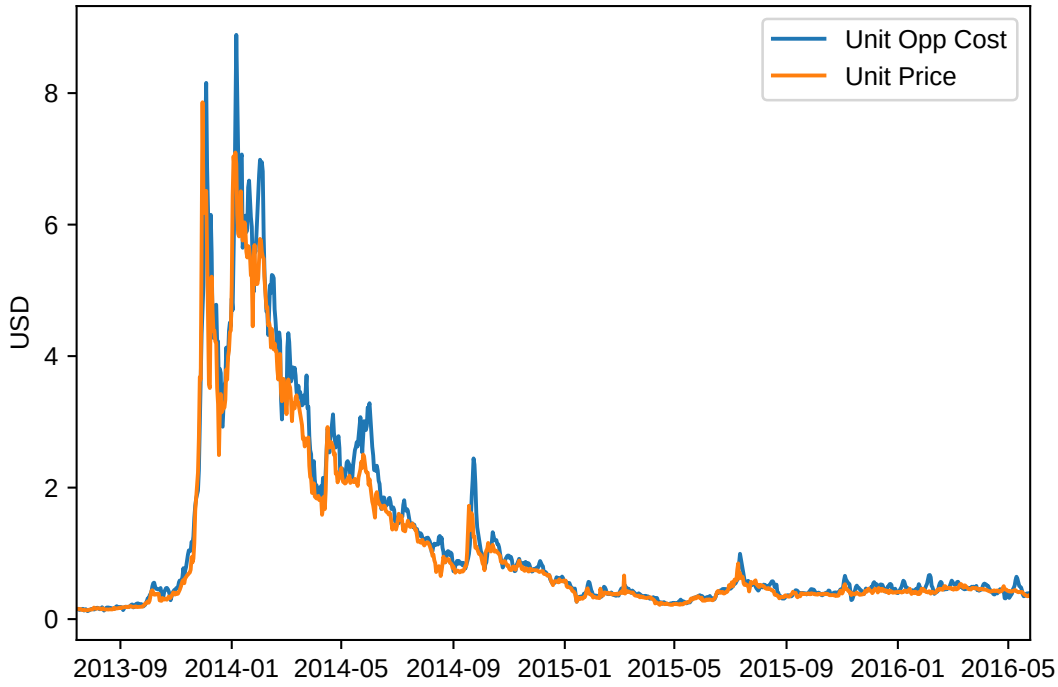


Figure 3.1. The opportunity cost of mining a unit of Peercoin (PPC), along with the market price on the day. For clarity, we have truncated the x-axis to show the prices between May 2015 and May 2016.

opportunity cost with the market capitalization, as shown in the "Market Cap" column in **Table 3.1**(b). For a given altcoin, we compute the market capitalization as $(p \cdot q)$, where p is the price of the altcoin on the last day in the analysis period, and q is the total number of coin units mined during this period. The market capitalization reflects the value of all the coin units at the end of the analysis period. If this value is greater than the total opportunity cost, then all the coin units are worth more than what mines have collectively invested at the end of the analysis period. For example, miners invested $4.1 million worth of opportunity cost in mining PPC, while PPC's market capitalization is almost double this amount. Potentially, PPC could be overvalued and miners could make a profit.

We argue that opportunity cost is an effective estimate of the actual mining cost,

as it is closely correlated to the market price; we show one example in **Figure 3.1**. One possible explanation is that as more hype is created around a coin, the market price increases, which in turn attracts more miners. This increases the difficulty of mining and also the opportunity cost, so the opportunity cost goes up along with the price. Conversely, a coin that has attracted a significant amount of hashing capacity has a high difficulty and thus opportunity cost. Thus, miners expect to sell the mined coin units at a higher price. In general, the opportunity cost for mining a unit of a coin is correlated with the coin's market price on the same day. The "Corr" column in **Table 3.1**(b) shows the Pearson correlation coefficient for the market price and the unit opportunity cost of a given altcoin on the same day. Across the 18 altcoins in the table (except BTC and LTC), 12 altcoins are associated with a correlated coefficient > 0.80. As a comparison, the Dow Jones Industrial Average and the S&P 500 Index between May 2012 and 2017 are correlated with a coefficient of 0.99 [30].

3.6 Estimating Profitability

In this section, we compute the profitability of miners and speculators. In our model, a miner always expends \$1 worth of opportunity cost in mining per day and sells the mined coin units on the same day. In contrast, a speculator purchases \$1 of an altcoin's units on one day, and he sells all of the coin units later at a different price. Since our dataset does not offer details on exactly what individual miners or speculators did in the past, we can only use the historical data to simulate the behaviors of hypothetical miners and speculators. In this way, we compute the respective profitability under various conditions.

Table 3.2. Mining continuously for 7 and 30 days. All units are in percentages unless otherwise stated.

	(a) 7 Days of Mining					(b) 30 Days of Mining				
Coin	*1st Day r*	$E(r)$	$\sigma(r)$	P	$T_{r\geq 0}$ *(Days)*	*1st Day r*	$E(r)$	$\sigma(r)$	P	$T_{r\geq 0}$ *(Days)*
ARG	4.41	2.61	7.22	76.28	8	-0.51	0.71	1.50	80.71	0
AUR	10.17	0.63	2.25	61.35	19	1.37	0.13	0.35	62.31	18
BTA	6.67	2.01	5.18	69.33	14	0.99	0.45	1.07	68.97	13
CURE	6.23	-6.68	5.92	14.54	18	0.72	-1.54	1.01	8.79	8
DGC	3.24	1.16	2.79	61.66	111	0.77	0.27	0.57	64.89	207
DOGE	70.63	4.22	10.51	100.00	N/A	8.88	0.76	1.31	100.00	N/A
DOT	10.48	18.29	12.12	99.21	14	2.94	4.92	1.15	100.00	N/A
EFL	16.48	2.00	2.07	89.29	30	1.61	0.47	0.34	94.89	18
HAM	13.49	-2.19	6.43	18.82	46	3.10	-0.51	1.30	14.89	65
PPC	0.09	-1.13	1.37	16.50	1	0.02	-0.26	0.17	3.40	4
RPC	10.52	0.74	3.08	59.54	6	0.82	0.17	0.46	59.85	32
SWING	2.74	-2.46	3.77	19.09	3	-0.04	-0.53	0.55	21.83	0
TROLL	-0.01	4.37	1.31	98.08	0	0.87	0.99	0.06	100.00	N/A
UNO	8.44	-5.43	5.52	20.99	79	1.26	-1.34	1.23	15.98	72
VCN	56.98	-2.22	11.77	27.18	34	7.40	-0.75	1.87	30.23	25
VIA	-1.68	2.76	2.09	95.07	0	0.71	0.67	0.33	100.00	N/A
WBB	6.21	6.58	4.45	97.87	83	0.75	1.54	0.78	100.00	N/A
XJO	3.03	-5.51	4.00	4.48	15	0.18	-1.28	0.75	0.83	4

3.6.1 Miners

We start by considering the profitability of miners. Using the unit opportunity cost that we computed in the previous section, we construct a simulator in which a miner continuously mines over a duration of d days, starting on Day i. Every day, he invests \$1 worth of opportunity cost in mining. At the end of each day, he sells all the coin units mined on that day. At the end of the d days, his total revenue would be v dollars. We vary i between 1 to $N-d$, where N is the length of the trading period; for instance, $i=1$ means that our simulated miner starts mining on the day when an altcoin is first listed on an exchange. We also vary d for $d=1,7,30$ days. For all these i and d combinations, we compute the daily average returns, r, by solving for r in this equation: $d(1+r)^d = v$.

Table 3.2(a) shows the result of our first simulation, in which a miner continuously mines for $d=7$ days. Our analysis covers 18 altcoins during the same period as shown in **Table 3.1**(b). The "1st Day r" column shows the value of r if the miner starts mining a coin on Day 1 and continues until Day 7, selling any mined units on the same day. In

contrast, the "$E(r)$" column computes the expected/mean r if the miner starts mining on a random day. The standard deviation is shown in the "$\sigma(r)$" column. The larger the standard deviation, the higher the risk of investment if a miner starts on a random day. For instance, mining DOT on a random day results in an expected daily return of $18.29 \pm 12.12\%$, which potentially presents a higher reward and risk than mining AUR, with an expected daily return of $0.63 \pm 2.25\%$. We stress that the r value is computed based on a simulated investment of \$1 worth of opportunity cost. In total, DOT has \$917 of trade volume and \$3,198 of total opportunity cost (**Table 3.1**(b)). Even though its r value is high, the amount of actual profit extracted is likely to be limited. In contrast, a PPC miner can generate an expected return of $-1.13 \pm 1.37\%$. Given that PPC is associated with millions of dollars worth of trade volume and total opportunity cost, an actual miner has the potential to suffer significant losses.

Another way to measure risk is to compute the probability, P, of achieving a positive r if a miner starts mining an altcoin for 7 days on a random start day. As shown in the "P" column, miners of altcoins like DOGE and WBB will earn positive returns on any random start day. XJO miners, by contrast, will earn positive returns on a random start day with a probability of only 4.48%.

We observe that of the 18 altcoins in the table, the 1st Day r values are higher than the corresponding $E(r)$ values in 14 altcoins, which suggests that miners of these 14 coins obtain higher returns if they start mining as soon as an altcoin is listed on an exchange, relative to the average case (i.e. $E(r)$). One possible reason is that when an altcoin is first listed, the amount of mining capacity it has attracted is still on the rise, as there is friction for miners to reconfigure their equipment to mine a new-to-market altcoin; thus the opportunity cost of mining tends to be low in the beginning. Furthermore, the market price is often high when an altcoin is listed — a period typically associated with hype. The gap between high price and low opportunity cost creates a potential for

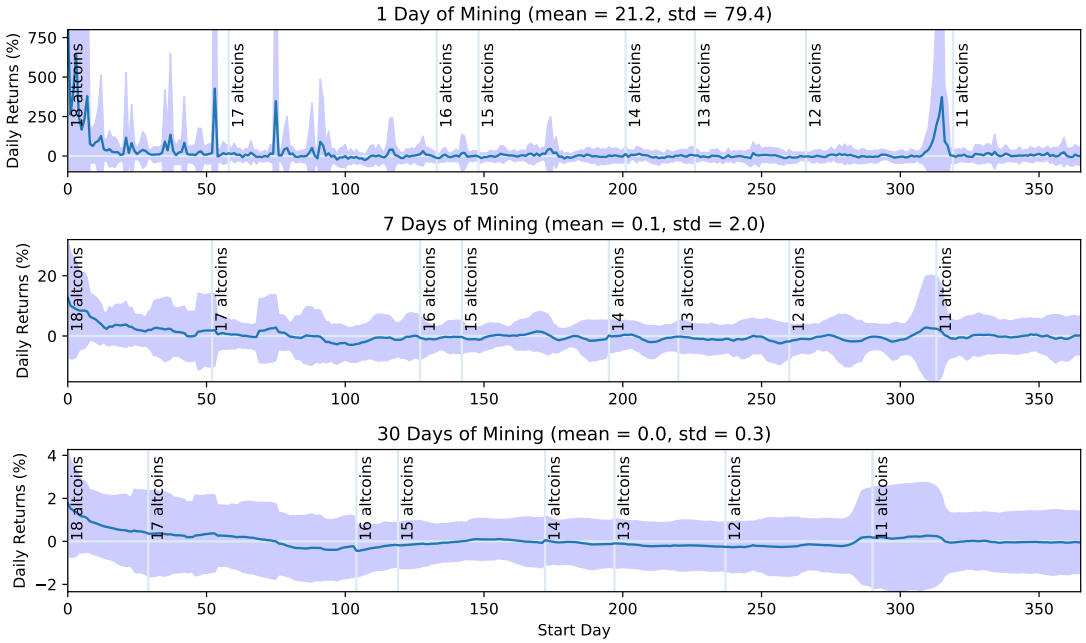


Figure 3.2. Mining a random altcoin.

miners to profit during this period.

In fact, miners can potentially profit during the first few consecutive days after an altcoin is listed on exchanges. Column "$T_{r\geq 0}$" shows the number of consecutive days since Day 1, such that a miner who starts mining on one of these days and continues for 7 days will not encounter a negative r. For example, an ARG miner who starts mining on any day of the first 8 days will receive $r \geq 0$; if she starts mining on Day 9, she would receive $r < 0$ for the first time (although she could still obtain positive returns subsequently). For TROLL and VIA, the 1st Day r is already negative, so $T_{r\geq 0} = 0$. For DOGE, all r values are positive regardless of the start day; thus $T_{r\geq 0} =$ N/A. Finally, we repeat the simulation above, changing $d = 30$ days. We show the results in **Table 3.2**(b). Again, for 14 out of the 18 altcoins, mining on Day 1 will yield a higher r than the expected case. The expected returns are lower for $d = 30$ than $d = 7$ in 11 of the altcoins, while all the $\sigma(r)$ values are smaller.

So far, we have examined the daily average returns for individual altcoins. This approach assumes that a miner already knows which altcoin to mine. Our next simulation departs from such an assumption, and it instead looks at a case where a miner randomly picks one of the 18 altcoins in **Table 3.2** and start mining on Day i. Again, $i = 1$ means the miner starts on the same day when a coin is listed on an exchange. The miner will stick to mining this altcoin for d days, devoting \$1 of opportunity cost of mining every day, and selling all mined units at the end of each day. Since the miner picks a coin at random for each i, our goal is to compute the distribution of daily returns for these 18 coins for given i and d values.

First, suppose we set $d = 1$ day. The top chart in **Figure 3.2** shows the result of our new simulation. The x-axis shows the start day of mining, i (relative to the first day of listing at an exchange for each coin). The y-axis shows the distribution of daily average returns, r. The solid line represents the expected (i.e. mean) r for picking a random coin and starting to mine on Day i for $d = 1$ day. The band above and below the solid line indicates the standard deviation of r. For i between 0 and 57, a miner can randomly mine one of the 18 altcoins on Day i. At its peak, the mean r is $885.1 \pm 2,781.0\%$ on Day $i = 0$. The expected r value decreases over time. Between $i = 58$ and $i = 132$, the miner can only pick one of 17 altcoins, as we do not have the trading data for one of the altcoins beyond 57 days. In general, the mean of the expected r values between $i = 0$ and $i = 365$ is $21.2 \pm 79.4\%$ (i.e. the expected returns for a miner who picks a random coin on a random start day).

The middle and bottom charts in **Figure 3.2** show the results for $d = 7$ and 30 days. As d increases, the expected r and its standard deviation both decrease. Effectively, with lower expected returns, the risks are potentially lower, too. Furthermore, as d increases, a persistent pattern is that the returns tend to be higher when i is low, regardless of the d values. This implies that a miner who picks a random altcoin and mines it shortly

Table 3.3. A speculator that holds for 7 and 30 days. All units are in percentages unless otherwise specified.

	(a) 7 Days of Speculating					(b) 30 Days of Speculating				
Coin	*1st Day r*	$E(r)$	$\sigma(r)$	P	$T_{r<0}$ *(Days)*	*1st Day r*	$E(r)$	$\sigma(r)$	P	$T_{r<0}$ *(Days)*
ARG	10.72	-0.32	5.56	39.02	0	-0.48	-0.41	2.90	31.48	26
AUR	6.69	-0.22	5.87	41.38	0	3.39	-0.52	2.46	40.22	0
BTA	-26.24	0.35	6.87	47.31	11	-11.71	0.56	3.06	54.35	31
BTC	-1.97	0.44	2.51	55.17	6	-0.72	0.42	1.43	57.59	18
CURE	-3.74	-0.45	3.76	41.90	22	-6.30	-0.40	1.77	40.19	47
DGC	-5.33	0.00	4.44	41.26	10	-1.90	-0.06	2.21	37.04	10
DOGE	86.27	-0.06	4.53	39.06	0	5.24	-0.06	1.48	37.84	0
DOT	-17.47	-0.37	20.13	41.09	2	-18.67	-1.68	7.79	45.28	23
EFL	17.07	0.02	4.60	47.95	0	0.21	-0.18	2.06	47.83	0
HAM	-19.35	0.46	7.15	51.05	8	-1.12	0.35	2.35	64.24	1
LTC	-4.12	0.08	3.54	45.45	6	-0.34	0.05	1.63	39.55	6
PPC	-1.66	0.14	3.08	44.25	6	-0.16	0.11	1.60	46.99	3
RPC	-1.99	-1.13	4.89	39.08	1	-1.46	-1.25	2.24	28.50	88
SWING	1.61	-0.52	5.20	42.28	0	-1.04	-0.66	2.05	48.59	51
TROLL	-1.93	-0.69	4.38	51.92	1	-2.25	-0.16	1.22	51.72	13
UNO	3.34	-0.09	3.74	47.45	0	-2.08	-0.10	1.39	44.65	20
VCN	-43.26	0.80	7.99	58.97	5	-13.06	1.18	2.44	73.26	6
VIA	2.72	-0.21	3.53	44.74	0	1.49	-0.39	1.43	33.19	0
WBB	-20.82	0.17	5.77	41.97	22	-7.41	0.25	2.41	53.05	17
XJO	7.36	-0.82	3.37	35.68	0	-2.37	-0.86	1.58	29.90	25

after the altcoin is listed is likely to receive higher returns than later.

3.6.2 Speculators

In contrast to miners who acquire altcoins through mining, speculators acquire altcoins by buying from exchanges. Also, while miners mine and sell on the same day, speculators buy coins on one day and sell them later. We design similar simulations to measure the potential profitability for speculators. Specifically, we require that a speculator enter the market on a random day i, buy \$1 worth of coins, hold them the next $d-1$ days, sell all the coins at the end of the d-day period for a total of v dollars. Again, we compute the average daily return by solving for r in this equation: $1 \cdot (1+r)^d = v$.

Table 3.3 shows the results of our simulation. Similar to **Table 3.2**, **Table 3.3** shows the returns on the 1st Day, as well as the expected r values for $d = 7$ and 30 days. However, the trend is the opposite. Across the 18 altcoins, plus LTC and BTC, a speculator who enters the market on Day 1 receives *lower* returns than the average case

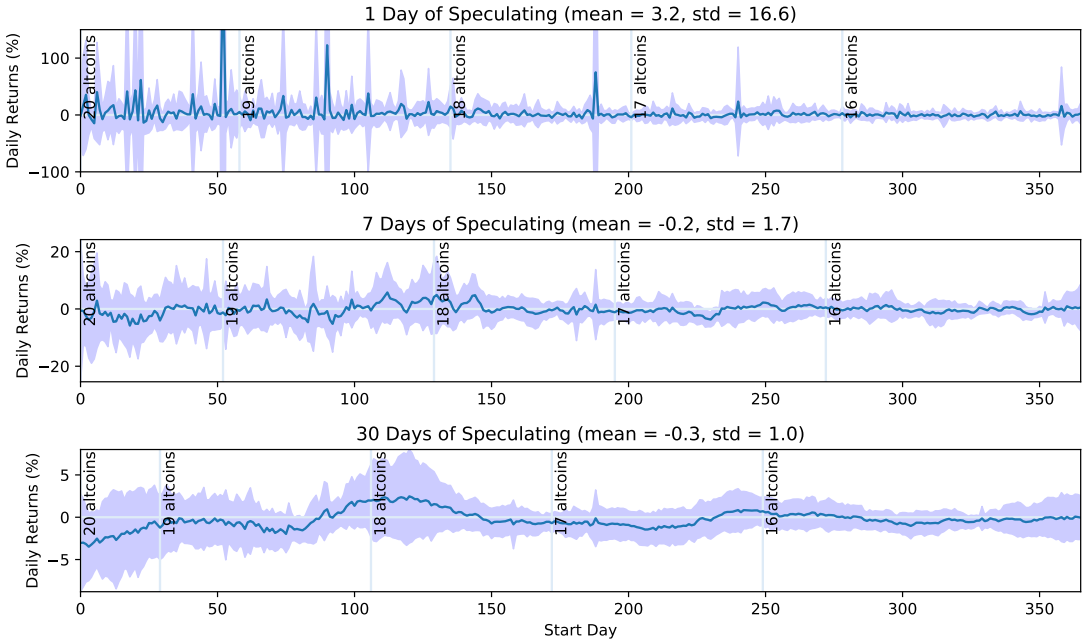


Figure 3.3. Speculating a random coin

for 12 of the coins for $d = 7$; for 30 days, we observe the same trend across 16 of the coins. In contrast to the $T_{r \geq 0}$ metric in **Table 3.2**, we compute $T_{r<0}$ here in **Table 3.3**, which counts the number of consecutive days since Day 1, such that if a speculator enters the market on one of these days for a given d value, she will receive $r < 0$. For instance, for $d = 7$ days, if a CURE speculator enters the market on any day between 1 and 22, she will receive negative returns; on Day 23, she will receive positive returns for the first time.

In addition to analyzing the returns for individual altcoins, we examine the case where a speculator picks an altcoin at random. **Figure 3.3** shows the result. Similar to **Figure 3.2**, **Figure 3.3** shows that as d increases from 1 Day, 7 Days, to 30 Days, the mean of the expected r values decreases, and so does the standard deviation; in other words, as the holding time increases, the potential returns and risks decrease. Contrary to **Figure 3.2**, **Figure 3.3** shows that a speculator who picks a random altcoin and enters

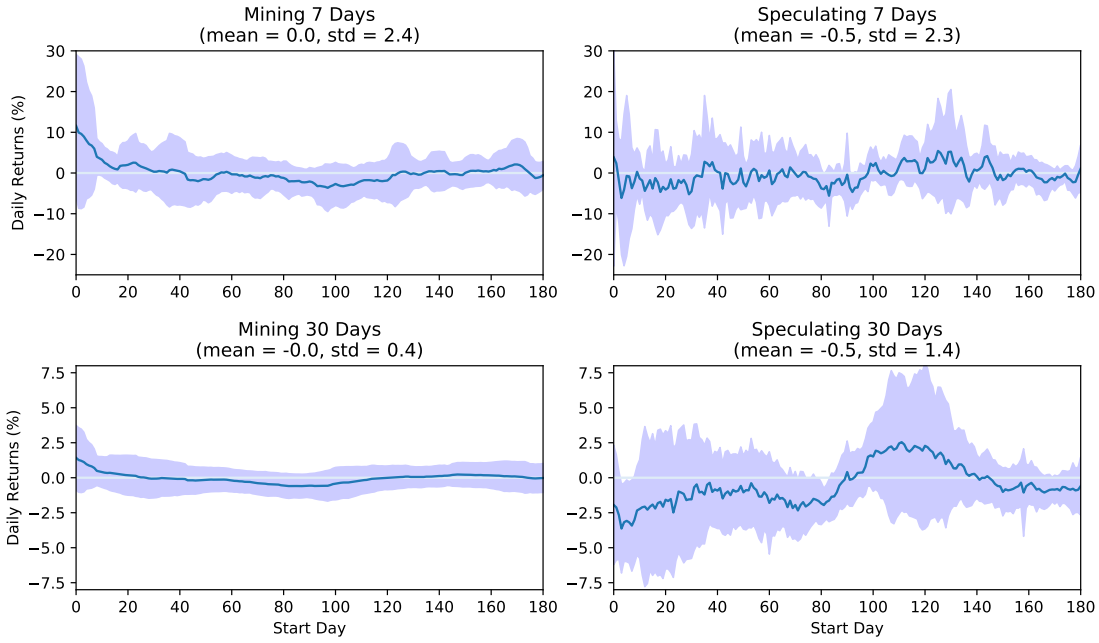


Figure 3.4. Comparing mining and speculating for the same set of 14 altcoins.

the market soon after the altcoin is listed on exchanges is likely to receive *lower* returns than if she enters the market later.

To compare the returns between mining and speculating, we identify 14 out of the 18 coins, such that all of them can be involved in mining and speculation for $d = 7, 30$ days and $i = 0, \ldots, 180$ days. Again, we assume that an investor randomly picks one of the 14 coins on Day i, enters the market either by mining or buying, and exits d days later. We show the results in **Figure 3.4**. For both $d = 7$ and 30 days, if an investor who picks a random altcoin decides to enter the market early, her expected returns will be higher if she becomes a miner. For $d = 30$, if an investor decides to become a speculator, she can potentially receive higher returns in the best case (e.g. more than 6% around $i = 120$) than the best-case returns in mining (i.e. less than 5% at $i = 0$). However, the risk of speculation is also higher, as indicated by the larger standard deviation for the expected r value.

3.7 Case Studies

Using WBB and DOGE as examples, we illustrate two observations that are common across other altcoins. First, as shown in Section 3.6, many coins are associated with high r values. While the price of coins tend to closely correlate with the opportunity cost, there are periods in which the price is higher than the opportunity cost, thus creating the potential for profitability. Second, coins like DOGE attract a significant amount hashing power, to a point where it is comparable to LTC. We show that using LTC as the base currency for estimating the opportunity cost is associated with decreased correlation between the price and the opportunity cost. This suggests that the resultant opportunity cost may not be an accurate estimate for the actual mining cost.

To visualize these observations, we plot two graphs for each coin: (a) the unit price against the unit opportunity cost every day, which would allow us to visualize the correlation, or the lack thereof, between price and opportunity cost; and (b) the expected number of hashes across the network over the same time frame. Although we are unable to obtain the actual number of hashes, the expected number of hashes, derived from the difficulty value, can approximate the relative number of hashing power at different points in time for a given coin, or across multiple coins. It is an indicator of the relative supply of hashing power from miners.

3.7.1 WBB

Between March to May 2016, the unit price of WBB is consistently higher than the unit opportunity cost, as shown in the top chart of **Figure 3.5**. A miner during this period is likely to be profitable. For example, on April 26, 2016, the unit opportunity cost is 8.3 cents, while the unit price is 16.9 cents. In other words, expending the same energy on LTC-mining (i.e. the base currency) would produce only 8.3 cents of revenue; mining

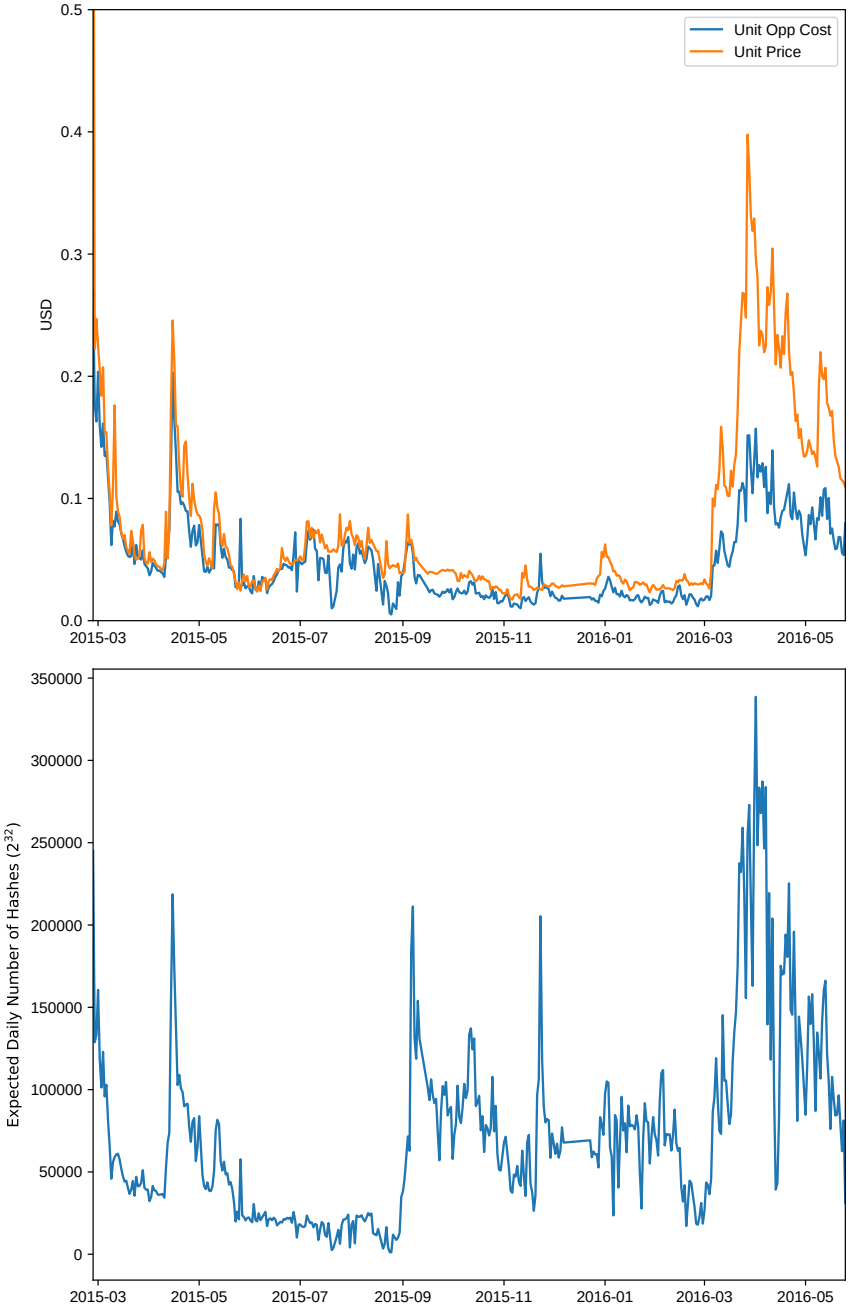


Figure 3.5. Top: The opportunity cost of mining a unit of WBB, compared with the market price. Bottom: The expected number of hashes per day. The x-axis starts on the same day as WBB's first block.

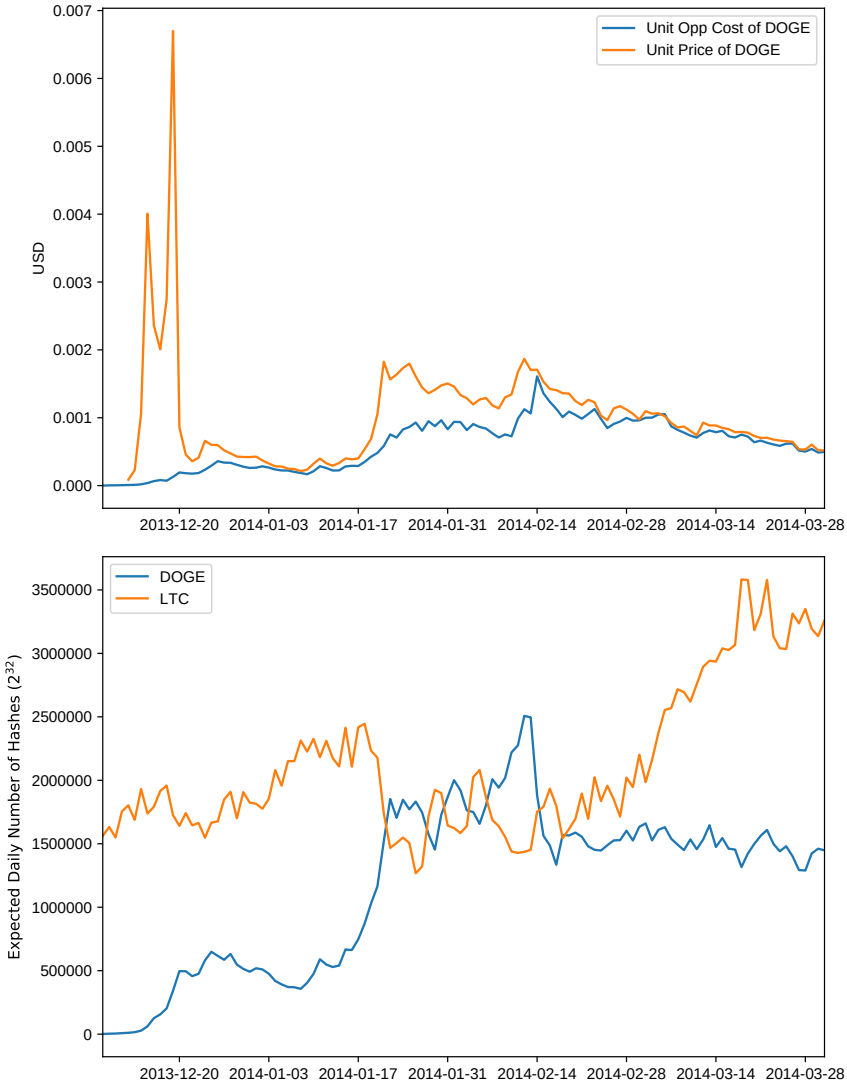


Figure 3.6. Top: The opportunity cost of mining a unit of DOGE, compared with the market price. Bottom: The expected number of hashes per day for DOGE and its base currency LTC. The x-axis starts on the same day as DOGE's first block.

WBB would double the revenue.

Given the potential profitability, one would expect the market to react to close the gap between the price and the opportunity cost. Intuitively, a higher profitability would draw more miners, which would increase the difficulty of mining and thus the opportunity cost of mining. Miners would keep coming until the market reaches equilibrium, when the price is equal to the opportunity cost.

This intuitive behavior is partly observed in WBB, as shown in the bottom chart of **Figure 3.5**. As the gap between the price and opportunity cost widens beginning of March 2016, the expected daily number of hashes also begins to rise, which, in turn, causes the opportunity cost to rise. However, this rise in hash computations is not high enough to further drive up the opportunity cost and close the price-cost gap. We see this behavior not only in WBB, but also in coins like AUR and VIA, where the price is consistently higher than the opportunity cost during certain periods.

3.7.2 DOGE

DOGE is similar to WBB, in that around December 2013, the price is consistently higher than the opportunity cost, as shown in **Figure 3.6**. The expected number of daily hashes also rises, but not enough to drive up the opportunity cost and close the price-cost gap.

Furthermore, there is another observation unique to DOGE. Between Jan 17 and Feb 14, 2014, the price is higher than the opportunity cost, and the expected number of hashes of DOGE surpasses that of LTC, the base currency for DOGE. Throughout our analysis, we have always used LTC as the base currency for computing opportunity costs, as we assume LTC is associated with a higher trade volume and hash rate than a typical altcoin, and that miners are more likely to view LTC as an alternative when deciding to mine a Scrypt-based coin.

This assumption is invalidated during the Jan-Feb period of DOGE. As the expected number of hashes toward DOGE-mining increases, the expected number of hashes toward LTC-mining decreases by similar amounts. It is likely that the hashing power originally dedicated to LTC-mining was shifted to DOGE. As a consequence, using LTC to compute the opportunity cost results in an inaccurate estimate of cost and a decreased correlation between the price and opportunity cost.

3.8 Discussion

In this section, we discuss a number areas where our methodology might be limited. First, our choice of base currencies are BTC and LTC, as they have the highest trade volumes and their prices are relatively stable. However, they are by no means the only candidates for base currencies. DOGE and PPC, for instance, trail behind BTC and LTC in terms of total trade volumes, but they, too, enjoy relatively stable prices. Furthermore, as we show in **Figure 3.6**, there were more hashes toward mining DOGE than LTC for close to a month. Using LTC as the base currency is associated with a decreased correlation between the price and the opportunity cost. During this period, an alternative analysis for Scrypt-based altcoins might use DOGE rather than LTC as the base currency. In general, using coins other than BTC and LTC as the base currencies may potentially produce different opportunity costs and present a different analysis of the market characteristics.

Second, the opportunity cost equates each hash of mining coin X to mining coin Y, provided X and Y both use the same hash function. In reality, such hash-to-hash comparison may be inaccurate. For example, a miner could be still computing hashes for an old block when a new block is announced. Depending on the network latency, it may take some time for the miner to receive the new block announcement. In the meantime, hashes are wasted on the old ("stale") blocks.

Third, in characterizing various altcoin markets, we estimate the profitability of mining or speculating by assuming $1 of investment. While this is a sufficiently small amount that would be unlikely to affect the market size, the resultant r values that we compute are not necessarily applicable to much larger investments. Even a few hundred dollars of investment may be enough to swing the price in small markets with only thousands of dollars of trade volume. Even in markets with much higher trade volumes, the actual amount of liquidity may be significantly less. A participant could be trading with herself to create an illusion of supply and demand in the market. Given the data we have, there is no way to rule out such "shadow" trades, thus making it difficult to estimate the true supply and demand.

Finally, in converting altcoin values into equivalent USD values, we first compute the value of altcoins in BTC and subsequently convert BTC into USD. While typically the markets with the highest trade volumes tend to be between altcoins and BTC, this is by no means the only way to sell altcoins. Many exchanges such as Cryptsy allowed speculators to trade altcoins with more well-known altcoins such as PPC, LTC, and DOGE. Our analysis assumes that the market is perfectly efficient and that altcoin prices are the same regardless the trading pair. In relatively small markets with low trade volumes, however, this assumption may not be true.

3.9 Conclusion

In this chapter, we tracked the movement of altcoins, analyzed the blockchain, and examined the trade data to measure the profitability of mining and speculating in 18 altcoins across multiple investment strategies. By comparing against BTC and LTC, we use opportunity cost to estimate the miners' effort in the 18 coins, and we design simulations to estimate the daily returns per $1 of investment, either through mining or speculating, under various conditions. These simulations show that a miner who

starts mining shortly after an altcoin is listed can potentially earn higher returns than the average case, whereas a speculator who enters the market shortly after an altcoin is listed on exchanges might potentially earn lower returns. We also show that returns from mining a random altcoin tend to be lower with smaller standard deviations — less risky — than from speculation.

Chapter 3, in full, is currently being prepared for submission for publication of the material. Huang, Danny Yuxing; Levchenko, Kirill; Snoeren, Alex C. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Tracking Spam Transactions in Bitcoin

So far, we have considered scenarios where crypto-currencies are used to make money through mining or speculation. Aside from these use cases, crypto-currencies can be used for commerce. However, as described in Chapter 1, Bitcoin transactions are prone to delays. A transaction can be held in the mempool from several minutes to hours if there is a large volume of transactions waiting to be confirmed. As service providers are typically unwilling to deliver until transactions are confirmed, customers may need to wait a long time in extreme cases.

In this chapter, we study one cause to such delays: spam transactions. These transactions typically send the smallest amount allowed by Bitcoin's protocol, or they might have a disproportionately large number of transaction outputs with respect to the average case. Such transactions consume what is already limited space in each block, thus preventing other transactions from being confirmed in a timely fashion. Our goal is to use the blockchain and mempool to identify spam transactions and measure their effect on other transactions. To this end, we cluster likely spam transactions and show that on average spam transactions raised transaction fees by 51% and increased the confirmation time 7-fold. At the same time, we show that the miscreants expended a total cost of $49,000 — a modest amount with respect to the daily Bitcoin transaction volume of at least $10 million in 2015 [15].

Due to increased fees and confirmation time, spam transactions can potentially disrupt normal commercial activities on the Bitcoin network. One mitigation is for merchants to not wait for the confirmation time, and instead to deliver the goods or services once the payment reaches the mempool. We will explore some of the associated risks in Chapter 5.

4.1 Introduction

Whereas Visa and PayPal typically clear a transaction within seconds, the waiting time for Bitcoin can range from a few minutes to several hours. Each Bitcoin block holds at most 1 MB of transactions. Even though a block is mined approximately every ten minutes, a large volume of transactions in the mempool leads to a significant backlog of transactions waiting to be confirmed. A typical merchant would be unwilling to deliver the goods or services until the relevant transactions are confirmed for fear of double-spending attack. As such, transaction delays can be disruptive to commercial activities that use Bitcoin.

The wait time depends on how a transaction is prioritized relative to others in the mempool. Recall from Section 1.2.2 that transactions are prioritized based on two rules. For transactions less than 1 KB whose outputs are at least 0.01 bitcoins each, transaction fees are unnecessary, and the priority is based on the P value (Equation 1.1). All other transactions are prioritized based on the fees per KB. It follows that transactions with low P values or low per-KB fees tend to experience higher delays (assuming that the miner uses the default Bitcoin client).

These rules can be gamed to disrupt transactions. One instance that we will focus in this chapter is *spam* transactions, which take up space without useful purposes, e.g. by having disproportionately (with respect to the average transaction) high number of outputs back to the sender. Some of these transactions pay zero fees because the size is

less than 1 KB with at least 0.01 bitcoins per output. As time goes by, P automatically goes up, leading to the confirmation of these transactions into blocks and therefore taking up the valuable block space. On the other hand, some spam transactions pay low fees, but their per-KB fees might be high enough that non-spam transactions with lower per-KB fees are de-prioritized.

As such, we assume that spam transactions are unlikely to serve any real commercial purpose. They take up mempool and block space, and cause non-spam transactions to be de-prioritized. Our goal is to identify spam transactions and measure their impact on non-spam transactions. To this end, we analyze Bitcoin's blockchain.

In particular, we examine the Bitcoin mempool and Blockchain transaction graph from July 7 to 17, 2015, a period in which proponents of higher block sizes were reported to have sent spam transactions, in an effort to show the limitations to the current 1 MB block size [43, 6]. We use k-means clustering and a set of features we identified to differentiate spam from non-spam transactions, such as transactions with low transaction fees per byte, transactions that sent to invalid outputs, and transactions that sent to a large number of outputs. We identify 385,256 (23.41%) out of 1,645,667 total transactions as spam during the study period. Further analysis of transactions in these clusters allow us to identify four distinct motifs of spam transactions. Based on our identification of spam and non-spam transactions, we are able to estimate the cost of this spam campaign as $49,000 at least. In comparison, at least $10 million worth of transactions occurred on a daily basis in 2015. The $49,000 cost of spam is thus considered modest, while the effect is significant. The campaign led to an increased average transaction fee by 51% (from 45 to 68 Satoshis/byte) and processing delay by 7 times (from 0.33 to 2.67 hours).

Our study makes several contributions, including proposing and empirically validating a method to identify spam transactions, characterizing the spam transactions, measuring the impact of this spam campaign on Bitcoin, and proposing changes to

transaction fees that would mitigate the effectiveness of spam transactions.

This chapter is structured as follows. In Section 4.2, we describe how we collected our data and how to use k-means to cluster spam transactions. Using the cluster information, we measure the impact of spam on non-spam transactions, in terms of increased fees and delays, in Section 4.3. We discuss limitations to our approach, and we propose potential mitigations in Section 4.4. We conclude in Section 4.5.

4.2 Methodology

We describe how we collected the mempool and blockchain data in Section 4.2.1. We use this data to identify spam transactions in Section 4.2.2.

4.2.1 Data Collection

In our study, we set up a server connected to a public-facing network. We installed the Bitcoin client (Core v0.11) and kept it running between June 19 and September 23, 2015. We collected three main data sets using Bitcoin daemon's JSON-RPC interface.

1. Bitcoin Blockchain: On September 23, we downloaded the entire blockchain using the `getblock` and `getrawtransaction` methods. This returned details for all blocks and transactions, such as the timestamps of blocks, the timestamps at which we received the transactions, the number of transaction inputs and outputs, as well as the input and output amount. We stored the data as plain-text JSON strings. As a result, the total data size is 350 GB.

2. Mempool: Between June 19 and September 23, the `getrawMempool` method was invoked every minute. This returned a list of unconfirmed transaction IDs currently in the Mempool. These would be either committed to the blockchain or later discarded by the P2P network. We saved this list of transaction IDs, along with

Table 4.1. Overview of our datasets. All data sets cover a period between June 19 and September 23.

Data	Period	Size
Blockchain	Between Jan 9, 2009 and Sept 23, 2015	350 GB
Memory pool	Between June 19 and Sept 23, 2015	250 GB
Unconfirmed transactions	Between June 19 and Sept 23, 2015	1.3 TB

the timestamp of the RPC call, on the Hadoop file system. During this period, we captured 12 million distinct transactions in the mempool, which amounts to 250 GB of plain-text data.

3. Unconfirmed transactions: For every unconfirmed transaction ID that we had obtained above, we *immediately* looked up the transaction details using the `getrawtransaction` method, since the Mempool could discard the transaction any moment. To optimize for speed and storage, we ignored transactions that we had previously seen. Finally, we saved all the transaction details, along with the data collection timestamp, on Hadoop. Between June 19 and September 23, we captured 1.3 TB of unconfirmed transactions in plain text.

The total size of the data collected is 2 TB, which we saved as plain-text JSON strings on the Hadoop file system and analyzed with Spark. We summarize our data sets in Table 4.1.

As we collected data using only a single node, our perspective of the P2P network — and thus the transactions in the Mempool — is potentially biased. In particular, network propagation takes time. For transactions in the mempool, the timestamps that we observed are always later than the originating timestamps, but we have no knowledge of how much later. Furthermore, whether a transaction is relayed is up to individual nodes. A transaction created a few hops away is not guaranteed to reach our node. It is, however, beyond the scope of this paper to adjust for such biases. We assume that our observation

of the network is largely consistent with the rest of the network.

4.2.2 Clustering Transactions

Our collaborator, Khaled Baqer, uses unsupervised machine learning techniques to identify possible spam transactions. Even though the spammers publicly announced the spam period (i.e. July 7 to 17, 2015), the they did not disclose the precise transactions they sent [43, 6]. Without this ground truth, we can only cluster spam transactions based on their *motifs*. In other words, if we translate transactions into graphs — where each vertex represents a transaction, an incoming edge represents an input, an outgoing edge represents an output, and the edge weight represents the input/output amount — then patterns of incoming and outgoing edges become useful features for our clustering algorithm.

In particular, we use k-means clustering. After manually inspecting distinct motifs before and during the spam period, we set $k = 10$. Our paper contains more details on how we do the clustering [11]. In short, we have found distinct clusters that represent 1.6 million possible spam transactions during the spam period. Some of the transactions' features include the following:

- **Fan-in:** a disproportionately (with respect to average transactions) large number of inputs but only one output. There are about 14,000 *fan-in* transactions.
- **Fan-out:** one or two wallet addresses sending funds to a disproportionately large number of outputs. For example, we have seen transactions that sent from a single address to more than 3,000 output addresses. Such transactions increase of outputs that are unspent (see Section 1.2.2) at the time of transactions. Since the Bitcoin client caches unspent outputs in case they are spent later, this cache can potentially grow in size and crash some clients on memory-constrained devices [7].

- **Invalid output wallet addresses:** one input address sending bitcoins to an invalid output address (i.e. which cannot be decoded). In other words, there are no valid private keys that correspond to the output wallet addresses. Effectively, no one can use the output in subsequent transactions. There are 425,000 such transactions and this is the largest type of spam by transaction count.

- **Dust output:** Transactions with disproportionately small (compared to the average transaction) output amounts — e.g. sending 0.00000001 bitcoins, the smallest allowed unit.

It is important to note we do not have the ground truth. The features of clustering are based on our observation — in particular, how the motifs in the transaction graph differed before and during the spam campaign. For the rest of this chapter, "spam" refers to transactions that display the features above and thus are identified as spam. We do not have any external data source or spam list to validate our clustering. Instead, we apply the clustering algorithm to transactions before the spam period, between June 24 and July 7, 2015. As we will discuss in Section 4.3, the number of possibly spam transactions are smaller during the pre-spam period than the spam period.

4.3 Analysis

We now describe the effects of spam campaigns on the Bitcoin network — especially on users who send non-spam transactions, as well as the miners. For the users, we measure the change in transaction fees and transaction delays (i.e. the time between when we first observe a transaction in the Mempool and when the transaction is committed to the blockchain). A large amount of spam is likely to increase the backlog of unconfirmed transactions. As a result, transactions are delayed for longer time periods. With more intense competition, senders pay higher fees, in the hope that their transactions will be

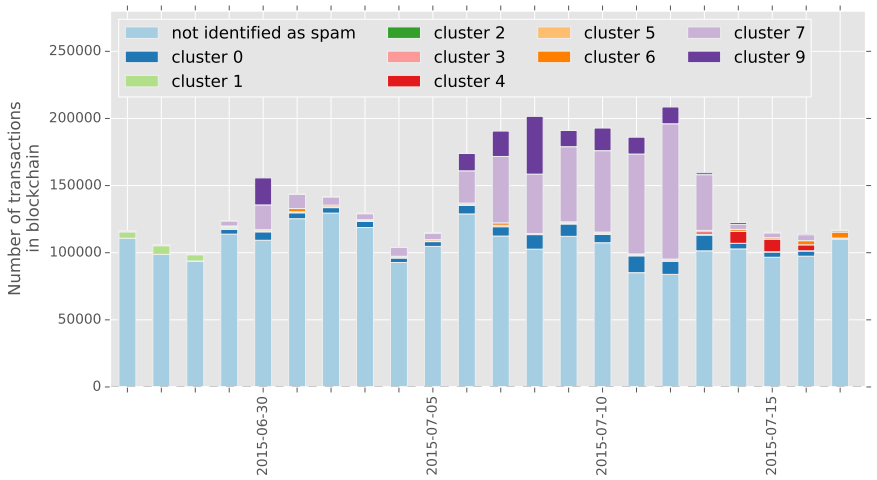


Figure 4.1. A stacked bar chart that shows the number of transactions per day in the blockchain. Note that the spam period is from July 7 to 17, and that Cluster 8 is not shown because we have merged it with Cluster 7. See the paper for details [11].

included in blocks sooner. For the miners, we measure the corresponding increase in the block reward.

Figure 4.1 shows the clustering results before and during the spam period (July 7 to 17, 2015); for a full description of individual clusters' features, refer to our original paper [11]. During the spam period, as much as half of a day's transaction volume was attributed to spam (e.g. July 12). In contrast, before the spam period, the number of transactions that we identify as spam is significantly smaller.

To highlight periods of the spam campaign, we measure the number of unconfirmed transactions in the Mempool, which indicates the amount of backlog in the network. Every minute, we take a snapshot of the Mempool and count the number of unconfirmed transactions. We take the average on a daily basis and plot the result in Figure 4.2.

Each major spike in the graph refers to a period of significant backlog. The first spike, which happened between July 7 and 17, corresponds to the spam campaign in our

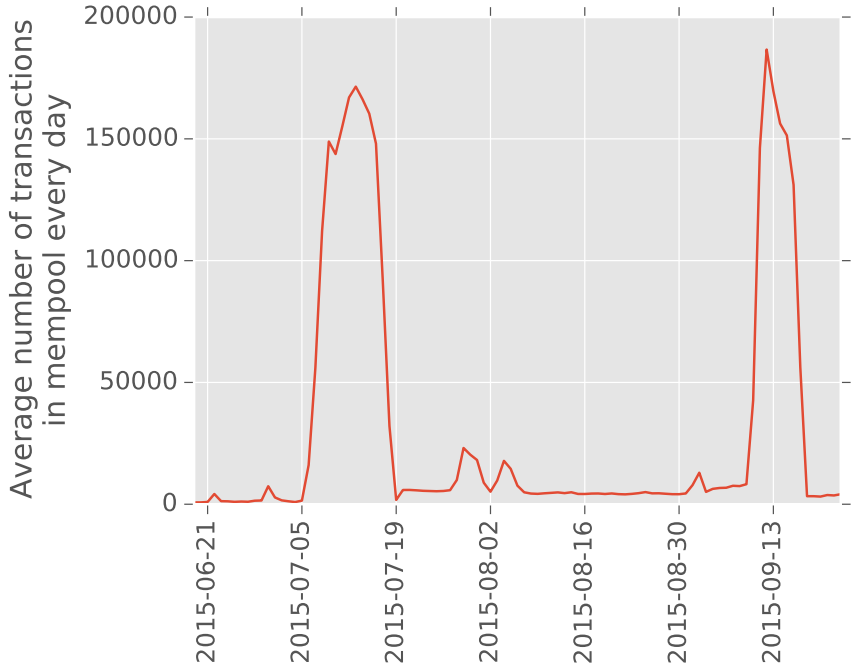


Figure 4.2. The average number of unconfirmed transactions in the memory pool every day.

study. There are sporadic spikes between July and August, but we do not have sufficient insight on the cause. Finally, a spike appeared around September 13, when an anonymous group conducted another spam campaign to show the limitations to Bitcoin's 1 MB block size.. As a result, a large number of transactions were created to compete for the free bitcoins, although only a few of them would be included in the blockchain eventually. Such a deluge of transactions caused the second backlog in Figure 4.2.

Focusing on the July 7 to 17 spam period, we next examine the number of transactions that were committed to the blockchain. We are interested in how each block allocates its scarce 1 MB of real-estate space to spammers and non-spammers. As shown in Figure 4.3, the number of transactions surged during the spam period. Between a quarter to half of the daily transactions have been identified as spam. As a baseline comparison, we also show that the number of spam transactions before the spam period

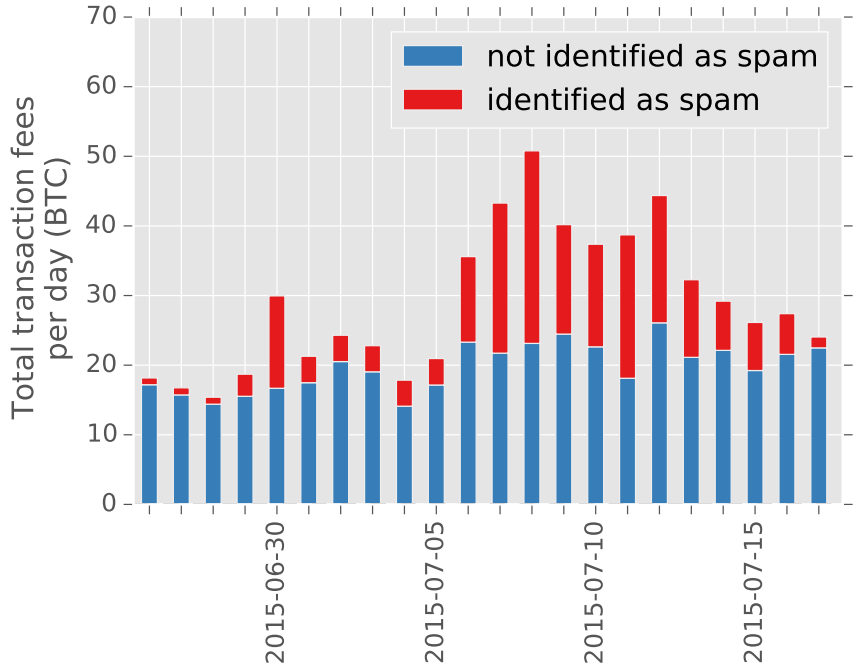


Figure 4.3. A stacked bar chart showing the total amount of transaction fees every day.

is significantly lower, with the exception of June 30. Based on anecdotal evidence, some users were attempting to stress-test the Bitcoin network on a small scale, which resulted in a brief rise in spam [55].

For the non-spammers, the spam period was a time when both transaction fees and delays were higher than normal. We show the comparison in Figures 4.4 and 4.5. On average, the delays in processing non-spam transactions increases by 7 times, from 0.33 to 2.67 hours. Likewise, the average non-spam transaction fees also surged, increasing from 45 to 68 Satoshis for every byte of transactions (or from $0.11 to $0.17 USD per kilobyte of transactions) — an uptick of 51%.

While non-spammers suffered, miners slightly benefit from the fee hike. As shown in Figure 4.3, miners were earning twice their normal fee-based revenue during the mid-July spam period, as compared with the non-spam period. However, even on days with maximum fees, the amount of extra mining income from fees was less than

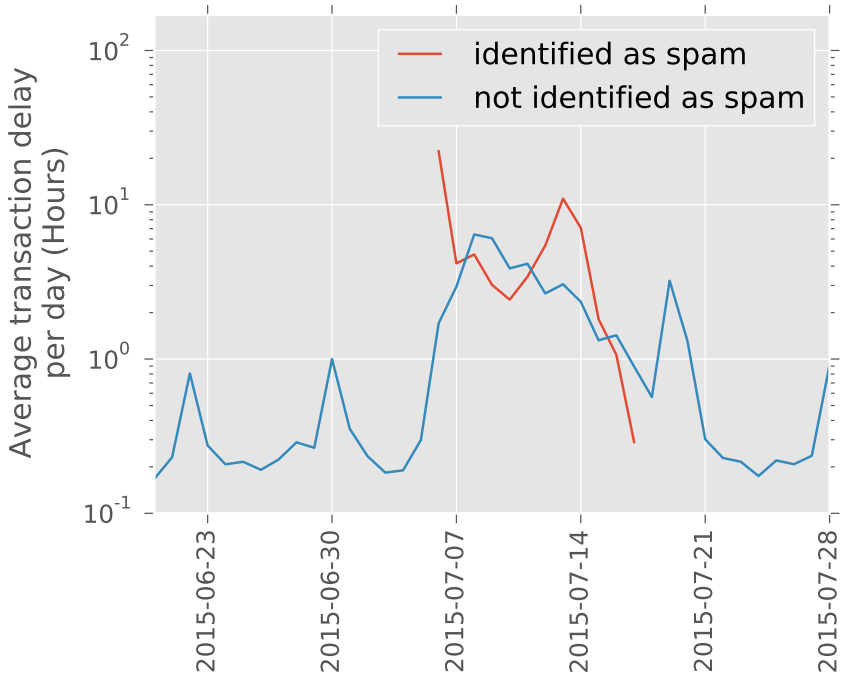


Figure 4.4. Average transaction delay between when a transaction appears in the Mempool and when it is committed to the blockchain.

1% of the block reward (which is 25 BTC per block, and about 3,600 BTC per day). The total transaction fees that spam transactions paid amounted to 201 BTC (or about $49,000 USD) over a 10 day time period — a modest sum that caused a rather noticeable disruption to the network.

4.4 Discussion

One surprising result in this work is that spammers were able to cause significant disruption at a relatively small cost. Considering the millions of dollars transacted daily in 2015, the $49,000 fees associated with spam transactions are modest. We speculate two factors were correlated with such a low cost. First, not all transactions need to pay for fees. As explained in Chapter 1, transactions less than 1 KB and whose outputs are at least 0.01 bitcoins each are prioritized not based on their fees but instead on the *P*

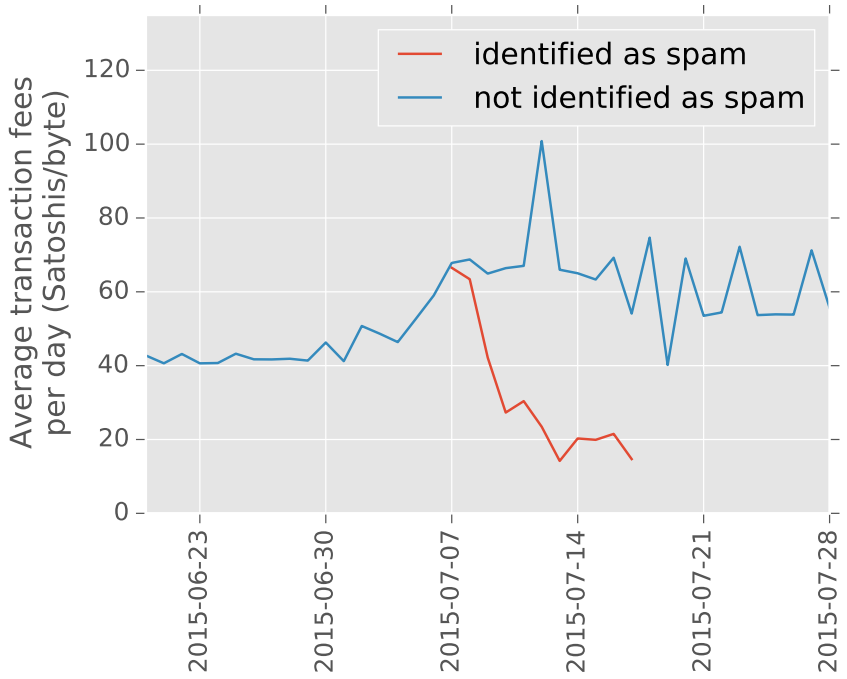


Figure 4.5. Average transaction fees per transaction per day. Note that the fees are normalized against the size of each transaction.

value, which depends on the output value and the age. We suspect that spammer created such transactions at no cost, and when their P values were high enough given sufficient time (as shown in **Figure 4.4**), these transactions confirmed into the blockchain, thereby taking up block space.

The second factor has to do with the miners. If a miner is running the default Bitcoin mining client, then transactions are prioritized based on the protocol described in Section 1.2.2. However, a miner could be running off-the-shelf software released by third-party developers. Whether these developers closely adhered to the standard protocol is beyond our knowledge. We speculate that some low-fee transactions were confirmed as a result.

Even in the absence of spam, it is difficult for senders of transactions to set optimal fees. If the fee for a transaction, at the time of entering the mempool, is lower

than average, the transaction may get delayed. If the fee is too high, the transaction may be confirmed upon the next block, but at an unnecessary cost to the sender. Worst, the average fees dynamically change. What appears to be a suitable fee at one second may be lower than average in the next second. As such, setting an optimal fee that takes into account the cost of fees and the confirmation time remains an interesting future research area.

4.5 Conclusion

In this chapter, we showed how to use the Bitcoin mempool blockchain to identify spam transactions and measure their effects. By analyzing transactions and clustering with k-means, we identified 1.6 million possibly spam transactions, about 27% of which (425,000) were transactions with invalid output addresses, during a 10 day period. We also showed that this spam campaign had a negative impact on non-spam transactions, increasing average fees by 51% (from 45 to 68 Satoshis/byte) and processing delay by 7 times (from 0.33 to 2.67 hours). This suggests that an adversary who is willing to expend modest amounts of bitcoin (at least $49,000 USD) to pay higher fees can disrupt Bitcoin's network. By analyzing Bitcoin's transaction graphs, we have shown how to detect such adversaries and discussed possible ways to thwart future attempts.

Given the risk of potential disruptions due to transactions delays, some merchants have decided to ship products or deliver services before their customers' payments are confirmed in the blockchain. In exchange for the added convenience to the customers, these merchants are taking the risk of double-spending attacks. A notable example includes Backpage.com, an online classified ads portal where ad authors can pay for ads in Bitcoin, and the portal would post the ads once the payment transaction enters the mempool. In Chapter 5, we will discuss how this approach leaks information about the ad authors.

Chapter 4, in part, is a reprint of the material as it appears in the Proceedings of The Third Workshop on Bitcoin and Blockchain Research, 2016. Baqer, Khaled; Huang, Danny Yuxing; Weaver, Nicholas; McCoy, Damon. The dissertation author was an author of this paper.

Chapter 5

Using Bitcoin's Side Channel to Cluster Backpage Ads

In Chapter 4, we showed that Bitcoin transactions are prone to delays. Merchants who wait for transactions to be confirmed before shipping the products may face delays from a few minutes to several hours. While waiting for confirmation reduces the likelihood of double-spending attacks, as explained in Chapter 1, the delay may cause inconvenience to the paying customers. Hence, some merchants are willing to bear this risk. A notable example, which we will discuss in this chapter, is Backpage, an advertisement portal similar to CraigsList; it is also known for ads that are indicative of human trafficking [45]. Ad authors pay for ads in Bitcoin. Once the payment transaction enters the mempool, Backpage posts the ad within seconds, without waiting for the transaction to be confirmed in the blockchain.

Because Backpage ads appear almost immediately after Bitcoin payments, this process has inadvertently created a timing side channel, where the timestamp of a Backpage ad is closely coupled with the corresponding Bitcoin transaction. This chapter shows that we can take advantage of this side channel, and that we can establish one-to-one mappings between ads and potential transactions that paid for the ads, based on the timestamps in the mempool. If two ads are mapped to transactions with the same input

wallet addresses (i.e. same wallet address paying for two transactions), we identify these ads as potentially having the same payer. Using this technique, we have found 18 groups of related ads (26 ads per group on average) across 6 wallet addresses that potentially paid for these ads — out of 700,000 ads we scraped from Backpage. Such ad clusters could be a useful dataset for researchers and law enforcement agencies to conduct further investigations on human trafficking and potentially identify the criminals behind such operations.

5.1 Introduction

There are existing techniques that offer coarse-grained de-anonymization of Bitcoin wallet addresses [47]. For example, recall from Section 1.2.3 that we can cluster co-spent input wallet addresses. By repeatedly sending and/or receiving bitcoins from known entities, such as exchanges or marketplaces, we can discover a small number of wallet addresses that belong to these entities, along with the cluster of addresses to which they belong. While this technique produces a mapping between organizations and their addresses, it does not link individual actors at these organizations with their addresses. An darkweb marketplace, for instance, may allow individual users to pay Bitcoin for advertisements on illegal services. If law enforcement agencies could link such advertisements to the Bitcoin transactions responsible for the payment, then the agencies can trace the payers' wallet addresses — one step closer to identifying the criminals. We call this process *per-transaction* de-anonymization. To our knowledge, there is no known technique for such fine-grained de-anonymization of Bitcoin.

This chapter describes a technique for per-transaction de-anonymization on Backpage, where users pay bitcoins to post ads. An ad typically costs less than $10, but the actual cost depends on the number of cities in which to post the ad, and the number of times during the day/week the ad is posted on Backpage's website. After an ad author

composes the ad and pays the bitcoins, Backpage posts the ad upon seeing the payment transaction in the mempool. Hence, an ad poster does not need to wait for transaction confirmation before his ad appears.

This payment model runs the obvious risk of double-spending attacks. Furthermore, this model has created a timing side channel that makes per-transaction deanonymization possible. Specifically, an ad appears on Backpage within a few seconds of someone paying for the ad. By correlating the time and price of each ad with the time and cost of each transaction in the mempool, we can establish possible mapping between ads and transactions.

We show how to use this timing side channel to map ads to transactions and then cluster ads with potentially the same payers. To this end, we first obtain Bitcoin's mempool, identify transactions that sent bitcoins to Backpage, and extract their timestamps as well as output values. At the same time, we scrape Backpage's website. For every paid ad, we extract the timestamp and reconstruct the price. In this way, we couple the timestamp and price of an ad on Backpage with one or more possible transactions in the mempool. If two ads are associated with transactions that share the same input wallet address, then we consider the two ads to be in the same cluster. The same person has potentially paid for both ads. We then validate the cluster by examining the writing style or contact information for each ad in the cluster; we expect them to be similar. In the end, such clusters might be indicative of large human trafficking networks. We pass these clusters on to human trafficking researchers and law enforcement agencies for further analysis.

Out of the 700,000 Backpage ads that we scraped over a 4 week period, we have identified 18 groups of related ads (5 ads per group on average) across 6 wallet addresses that potentially paid for these ads. We have shared the clusters and our technique with NGOs which specialize in fighting human trafficking. The remainder of the chapter is

structured as follows. First, we describe how Backpage works in Section 5.2, where we show how an ad author can post and pay for an ad in Bitcoin, and how Backpage processes the Bitcoin payment. Next, we discuss how we collected the mempool and Backpage datasets in Section 5.3. Section 5.4 explains how we use the dataset to establish potential mapping between a Backpage ad and the likely payer's Bitcoin wallet address, and how we use this mapping to cluster ads. Using this methodology, we present our results in Section 5.5 and discuss how we could improve our technique in Section 5.6, before concluding the chapter in Section 5.7.

5.2 Background on Backpage

Like CraigsList, Backpage is an online platform where anyone can post ads. Unlike CraigsList, Backpage users can pay for ads with bitcoins. The ad will appear on Backpage within a few seconds of the payment transaction entering the mempool, without waiting for confirmation [58].

Specifically, here is the process for composing and paying for an ad.

1. Ad author composes an ad on Backpage's web form.

2. Ad author chooses ad features to purchase. Similar to CraigsList, Backpage is a local classified ads service. Users have to visit pages of particular geographical areas to see ads in those areas. As such, examples of ad features include which urban areas to display the ad, and how many times to automatically promote the ad to more prominent positions on the page. Each ad feature has a fixed price in US Dollars. For instance, posting an ad in a city costs exactly $1 as of July 2017.

3. Ad author proceeds to the payment page, which shows the following:

    - Bitcoin is the only acceptable form of payment.

- The page shows the total amount of bitcoins needed, converted from the US Dollar price in Step 2. Backpage computes this bitcoin value based on the instantaneous exchange rate provided by GoCoin, Backpage's payment processor [58]. Backpage then rounds the bitcoin value to four decimal places.
- The page also shows a wallet address, which we shall call *deposit address*, to which the ad author should send bitcoins. The wallet address is dynamically generated for every ad payment. As GoCoin is Backpage's payment processor, we assume that the deposit address belongs to GoCoin.

4. Using a Bitcoin client or an online exchange, the ad author sends the required number of bitcoins to the wallet address in the previous step.[1]
5. The ad author's transaction enters the mempool a few seconds later.
6. The ad appears on Backpage a few seconds later.

5.3 Dataset

We collected three datasets, which we will analyzing using the methodology in Section 5.4.

Backpage: A collaborator of ours, Rebecca S. Portnoff, scraped Backpage from December 11, 2016 to January 9, 2017, collecting all adult ads placed in the United States every hour, which contains a total of 741,275 unique ads. As Backpage does not show the price of individual ads, the collaborator had to reconstruct each ad's price. See

[1] For our technique to work, the payer must pay the exact amount of bitcoins in a single transaction. We discuss the limitations in Section 5.6.

the original paper for details [58] . In this way, we construct a dataset of Backpage ads that include their timestamps and prices.

Blockchain: On February 1, we downloaded Bitcoin's blockchain by running the default Bitcoin client. This dataset contains all transactions in the history of Bitcoin.

Mempool: Similar to Section 4.2.1, we took a snapshot of the mempool every minute using the Bitcoin client hosted at University of California, San Diego. From the snapshots, we computed the smallest timestamp for every transaction. This is the time when our Bitcoin client first observed the transaction in the mempool. We assume that other Bitcoin clients would first observe the transaction around the same time, as previous research has shown that the Bitcoin peer-to-peer network is a highly connected graph [19].

USD/Bitcoin Exchange Rate: When an ad is about to be purchased, Backpage displays its price in both USD and Bitcoin. The exchange rate is updated sometimes once per minute, and GoCoin provides an API that announces this exchange rate. We scraped this API every minute to compute the USD values for all transactions in the mempool.

5.4 Methodology

In this section, we show how we use the Bitcoin and Backpage datasets for our analysis. First, we show how to identify transactions that are likely to represent ad purchases on Backpage in Section 5.4.1. In Section 5.4.2 we map ads with transactions that are likely to have paid for the ad. We group these ads by the potential payer in Section 5.4.3, and we validate our result in Section 5.4.4.

5.4.1 Finding Transactions to GoCoin

As GoCoin is Backpage's Bitcoin payment processor, all ad purchases can be mapped to transactions in which one of the output wallet addresses belongs to GoCoin. Our goal is to find all such transactions, so that we can map them with ads in Section 5.4.2.

As the first step, we use data from Chainalysis.com, a company which de-anonymizes Bitcoin wallet addresses by constantly sending/receiving bitcoins to/from known entities [8]. By applying the technique in Section 1.2.3 on GoCoin, Chainalysis identifies some of the wallet addresses that belong to GoCoin. We call this set of wallet addresses the "strict" set. While this technique is not known to have false positives, it may not be able to discover all of GoCoin's wallet addresses (i.e. false negatives). As such, the GoCoin wallet addresses we obtain in this step thus far may be incomplete.

To compensate for the lack of coverage in the first step, our collaborator, Rebecca S. Portnoff,paid for 32 dummy ads on Backpage, using 32 distinct Bitcoin transactions at different times on different days.[2] In each of these transactions, we transferred the required amount of bitcoins (typically less than $5) into a distinct deposit address, which GoCoin controls. We observe the following features in these 31 GoCoin wallet addresses.

1. A GoCoin wallet address appears in exactly two transactions: one where it receives bitcoins from us, and one where its bitcoins are sent to somewhere else.

2. The amount of bitcoins a GoCoin wallet address receives (and equivalently sends) is always less than 1 bitcoins with exactly four decimal places. This amount is the same as that displayed on Backpage's payment screen just before we pay for an ad. We suspect that this feature is specific to Backpage transactions. GoCoin serves as the payment processor for other merchants; we have no knowledge whether other

[2]Unless specifically stated, the text in this chapter refers to work that this author was solely responsible for during the collaboration.

merchants adopt such this pricing format.

3. When a GoCoin wallet address sends bitcoins somewhere else, it is co-spent with wallet addresses such that at least 80% of them also display Features 1 and 2.

Based on the three features above, we identify other possible GoCoin wallet addresses on the blockchain; we call them the "heuristic" set of wallet addresses. In total, we identify 753,929 "strict" and "heuristic" wallet addresses that had transactions during our Backpage scrape period. About 1.5% of the addresses are exclusively in the "strict" set, and 69.6% exclusively in the "heuristic" set. The remaining 29.0% of the addresses are in both sets. Given that a wallet address is in the "strict" set, the probability of it being in the "heuristic" set is 95.5%. The remaining 4.5% includes, for example, Bitcoin output values less than 1 bitcoins and with only three decimal places. We suspect that the fourth decimal might be a zero, and thus some of these transactions do not satisfy Feature 2.

In the rest of the analysis, we take a union of the "strict" and "heuristic" addresses and refer to them as simply GoCoin wallet addresses. While taking the intersection would give us higher confidence that a transaction is a Backpage transaction, the resultant set of transactions which we can map to ads would be smaller. Ideally, we would like to maximize the candidate set of GoCoin transactions.

5.4.2 Mapping Ads to Transactions

Our collaborator, Rebecca S. Portnoff, developed a technique for mapping ads to transactions based on their timestamps and prices [58]. Formally, suppose A is the set of ads on Backpage where $a \in A$ are individual ads, $t \in T$ is the set of bitcoin transactions that sent bitcoins to GoCoin, and $w \in W$ are input wallet addresses for each $t \in T$. We construct an undirected bipartite graph, $G = (V, E)$, where the set of vertices

is $V = A \cup T \cup W$ and the set of edges is E. There is an edge between w and t if w is an input wallet address for t. Furthermore, there is an edge between a and t if t is a possible transaction that paid for a. We consider t to be a possible transaction for a if both of the following are true:

- The USD price of a is within 2% the output value of t (thus allowing for small inconsistencies in the USD- Bitcoin price).
- The difference between the timestamp of a's appearance on Backpage and the timestamp when t is first observed in the mempool is less than a minute. [3]

5.4.3 Grouping Ads by Payers

The edge between transaction t and ad a could be a false positive if the transaction, in reality, is not mapped to the ad. For example, suppose t_a is the actual transaction that pays for a, and that w_a is the input wallet address to t_a. It is possible that we fail to identify w_a as a GoCoin address (i.e. neither in the "strict" nor "heuristic" set). It is also possible that there is another transaction t' around the same time with the same cost. Our technique would map a with t', i.e. false positive. False negatives are also possible. Using the same example as above, t' may not exist. As such, we cannot map any transactions to a. Absent ground- truth, however, there is no way to establish false positives or negatives. Hence, an edge between a and t is only a probable or potential association between ads and transactions.

Our goal is to find a set of a's for which the corresponding t's all have the same input wallet address w. In practice, however, the mapping between a transaction t and an ad a is many-to-many. This many-to-many mapping between t and a makes it difficult to map a wallet address w to a. To this end, we construct a subgraph $G' \subset G$, where G is an

[3] Backpage shows ad timestamps to the minute. We cannot be more precise than this granularity.

undirected graph with A, T, and W as the vertices. In G, an edge between a w node and a t node exists if wallet w is the *sole input wallet address* of transaction t. We require subgraph G' to satisfy all of the following criteria.

1. Each t node should be adjacent to exactly one w node, because we already require every transaction in G to have a single input wallet address. However, we allow each w to be adjacent to one or more t, as a wallet address may be used across multiple transactions.

2. With exactly two hops (from w to t to a), each w node should be able to reach at least three a nodes with the same *hard identifier*. Hard identifiers are phone numbers and email addresses; two ads are related if they share the same advertised phone number and/or email address. This reachability suggests that w is likely to be the true owner for at least three of the a nodes; the presence of the shared hard identifier reduces the probability of having incorrect edges between t and a.

3. Each t should be adjacent to exactly one a. By transitivity, each a can reach exactly one w with two hops. In other words, one cannot find another wallet address, other than w, that can be mapped to ad a. This criterion attempts to further reduce the probability of incorrect edges between t and a.

4. With exactly two hops, each w should be able to reach at least two sets of a nodes with different hard identifiers. This suggests that these hard identifiers are likely to be related, in that they might have all used w to pay for the ads.

We define the resulting w in G' to be a Shared Hard Identifier wallet (SHI wallet). The last criterion allows us to find at least two sets of ads of different hard identifiers that are mapped to the same w. Even so, because of the potentially false positive/false

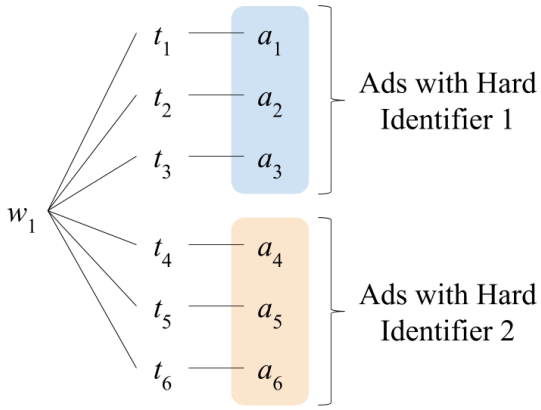


Figure 5.1. Grouping ads through Shared Hard Identifiers

negative problem we are unable to definitively conclude that w was used to pay for the ads.

One way toward validating SHI wallet addresses is to check the ad titles. Specifically, suppose our technique shows that two groups of ads, X and Y, have the same SHI wallet address. By definition, every ad $a_x \in X$ have the same SHI, and every ad $a_y \in Y$ have the same SHI. We say X and Y are potentially related if there exists some $a'_x \in X$ that has exactly the same title as $a'_y \in Y$. Even so, we still need further investigation to conclude that these ads are indeed related, which is beyond the scope of this chapter.

Figure 5.1 shows a hypothetical example of a subgraph G' that satisfies all our criteria. In particular, the wallet address w_1 is associated with two groups of transactions and ads that are mapped to two distinct hard identifiers. Within each group, there is a one-to-one mapping between each transaction and ad; for example, transaction t_i is mapped to ad a_i for $i = 1, 2, \ldots, 6$. In this way, w_1 is the shared hard identifier for these six ads.

5.4.4 Validation

Our collaborator, Rebecca S. Portnoff, paid for 32 dummy ads on Backpage. In all 32 cases, our transactions were observed in the mempool within 10 seconds, and the

ads appeared within 30 seconds afterwards.

Out of the 32 ads, Portnoff paid for 20 of them using Paxful, an exchange, such that each corresponding transaction has two input wallets and thus fails Criterion 1 (Section 5.4.2). For the remaining 11 ads, she paid with her personal wallet address, `1Ejb3`, across 11 distinct Bitcoin transactions. These 11 ads satisfy all four criteria above.

The price of the ads ranged from $2-$20. Using the methodology in Section 5.4.1, the output wallet addresses in the 11 transactions belong to GoCoin's "strict" and "heuristic" set simultaneously. Also, For the methodology in Section 5.4.2, 8 of the 11 ads are one-to-one mapped to our transactions. The remaining three ads all fail Criterion 3.

5.5 Results

Our collaborator, Rebecca S. Portnoff, applied the methodology above on the Backpage dataset, constructing both G and G'. After manually validating the results by checking the ad titles, she found 6 SHI wallet addresses associated with potentially related ads:

- SHI wallet address `1LYE` is associated with two groups of ads. The first group advertises Asian and Latina women in Los Angeles, while the second group advertises escort services in Los Angeles and New England areas.
- SHI wallet address `1Abg` is associated with two groups of ads that advertise Asian women in Illinois and the San Francisco Bay Area.
- SHI wallet address `1yVF` is associated with two groups of ads that advertise Asian women in San Francisco and Colorado.
- SHI wallet address `1BT6` is associated with five groups of ads that advertise college-age women in Hong Kong, Malaysia, and Taiwan.

- SHI wallet address `1Mhe` is associated with four groups of ads that advertise new immigrant women in the US.
- SHI wallet address `1N7V` is associated with three groups of related ads.

Overall, we discover 18 groups of ads across 6 SHI wallets, with 26 ads per group on average. For a detailed discussion of the results, see our paper [58].

5.6 Discussion

Our methodology requires that w be the sole input wallet address for t in constructing the subgraph G'. This restriction ignores transactions with multiple input wallet addresses, which is a common practice if bitcoins are sent from an exchange to GoCoin. In future work, we plan to include such transactions in our analysis. We cannot, however, reuse the same methodology, as these transactions are likely to be created by exchanges. Whereas our methodology assumes that an ad payer runs his own Bitcoin client and sends payments from the same input wallet address, an exchange sends from different input addresses every transaction, and every transaction may include multiple inputs. For future work, we can group ads by possible exchanges, using the clustering algorithm as described in Section 1.2.3. Although this technique will be less fine-grained than the method described in the chapter, it is still useful especially if the exchanges are hosted in the US. In this case, law enforcement can use our results and subpoena these services.

An additional limitation to our methodology is that it requires an ad author to pay exactly the required amount of bitcoins to Backpage in single transactions. If an ad author overpays or if the payment is split into multiple transactions, our technique will not be applicable.

5.7 Conclusion

In this chapter, we took advantage a timing side channel in Bitcoin to map Backpage ads with potential payers' wallet addresses. We used this potential mapping to cluster ads, which would offer a useful datasets for human trafficking investigators toward identifying the actual people behind these operations. To our knowledge, we are the first in the research community to map individual Bitcoin transactions with real-world actions. We evaluated our technique on real-world escort ads on Backpage during a 4-week period and uncovered 6 SHI wallets across 18 groups of ads. We are currently collaborating with multiple NGOs and discussing ways to help them fight human trafficking with our technique.

Chapter 5, in part, is a reprint of the material as it appears in the Proceedings of the ACM SIGKDD Conference, 2017. Portnoff, Rebecca S.; Huang, Danny Yuxing; Doerfler, Periwinkle; Afroz, Sadia; McCoy, Damon. The dissertation author was an author of this paper.

Chapter 6

Conclusion

Bitcoin, along with other similar crypto-currencies, is fundamentally decentralized. A peer-to-peer network collectively maintains the blockchain and mines any new coins. Even though activities on the network are public, identities are represented as wallet addresses, which are decoupled from the real-world identities of individuals involved in transactions or mining. As such, activities that involve crypto-currencies are partially anonymous.

This dissertation shows that we can leverage the public information on the blockchain and in the mempool to track four kinds of activities: (i) mining bitcoins on botnets, (ii) speculatively investing in altcoins, (iii) sending spam transactions, and (iv) buying ads on Backpage. For these cases, this dissertation shows that we can measure the financial activities and uncover the potential identities of the players involved.

To measure financial activities — e.g. quantifying the cost and revenue — we analyzed information from the blockchain and the mempool, along with trading history from exchanges. For Case (i), we examined transactions between botnet wallet addresses and mining pools, and we showed that botnets generated a modest revenue of $118,000 between 2012–2013. Upon analyzing the surging level of mining difficulty on the blockchain, we found that these botnets were likely to experience an exponentially decreasing revenue. For Case (ii), we compared the levels of mining difficulty across

different altcoins. The relative difference in mining difficulty allowed us to estimate the opportunity cost of mining a particular altcoin. Combining our analysis with external trading data, we estimated the potential profitability for multiple investment strategies. For Case (iii), we looked at both the mempool and the blockchain and measured the transaction delays. We showed that spam transactions increased the delay of non-spam transactions from 0.33 to 2.67 hours and the fees by 51%, while incurring a modest daily cost of $4,900 on the spammers.

To uncover the potential identities of the players involved, we clustered co-spent input wallet addresses and linked these clusters with real-world organizations, such as mining pools and exchanges. In this way, we identified 10 Bitcoin-mining botnets for Case (i). Furthermore, for Case (iv), we discovered a timing side channel involving Bitcoin's mempool, which allowed us to map Backpage ads to transactions that potentially paid for the ads, and subsequently to the input wallet addresses. In this way, we identified 18 groups of ads potentially paid for by 6 wallet addresses, thus providing evidence for law enforcement agencies that investigate human trafficking.

In summary, we discussed techniques to analyze the blockchain and mempool datasets, along with trading data from exchanges. We showed that these techniques enabled us to measure the financial activities and potentially identify the actors involved in the four cases above. Our insight can potentially help future researchers stop some of the malicious activities we described in Chapters 2, 4, and 5. Moreover, we hope our analysis will help crypto-currencies investors, such as those in Chapter 3, make informed decisions.

Bibliography

[1] Auroracoin. *http://auroracoin.is*.

[2] CryptoCoinCharts. *https://cryptocoincharts.info/*.

[3] CryptoID. *http://chainz.cryptoid.info*.

[4] Ethereum. *https://www.ethereum.org*.

[5] Ripple. *https://ripple.com*.

[6] July 2015 flood attack. *https://en.bitcoin.it/wiki/July_2015_flood_attack*, 2015.

[7] With a 1Gb mempool, 1000 nodes are now down (compared to yesterday). *https://www.reddit.com/r/Bitcoin/comments/3ny3tw/with_a_1gb_mempool_1000_nodes_are_now_down/*, 2015.

[8] Chainalysis.com. *https://www.chainalysis.com/*, 2017.

[9] Definition of Market Capitalisation. *Financial Times, http://lexicon.ft.com/Term?term=market-capitalisation*, 2017.

[10] Syed Taha Ali, Dylan Clarke, and Patrick McCorry. *Bitcoin: Perils of an Unregulated Global P2P Currency*, pages 283–293. Springer International Publishing, Cham, 2015.

[11] Khaled Baqer, Danny Yuxing Huang, Damon McCoy, and Nicholas Weaver. Stressing out: Bitcoin "Stress Testing". In *International Conference on Financial Cryptography and Data Security*, pages 3–18. Springer, 2016.

[12] Jörg Becker, Dominic Breuker, Tobias Heide, Justus Holler, Hans Peter Rauer, and Rainer Böhme. Can we afford integrity by proof-of-work? Scenarios inspired by the Bitcoin currency. In *The Economics of Information Security and Privacy*, pages 135–156. Springer, 2013.

[13] Dmitry Bestuzhev. Skypemageddon by Bitcoining. *Securelist - Information about Viruses, Hackers and Spam*, 2013.

[14] Bitmain.com. Buy ASIC Bitcoin Miners and Bitcoin Mining Equipment. *https://shop.bitmain.com/main.htm;jsessionid=AAB4EEFF7CB054CAF322AF314CB6FE40?lang=en*, 2016.

[15] BlockChain.info. Estimated USD Transaction Value. *https://blockchain.info/charts/estimated-transaction-volume-usd?timespan=all*, 2017.

[16] BlockChain.info. Hash Rate. *https://blockchain.info/charts/hash-rate*, 2017.

[17] BlockChain.info. Market Price (USD). *https://blockchain.info/charts/market-price?timespan=4years*, 2017.

[18] BlockChain.info. USD Exchange Trade Volume. *https://blockchain.info/charts/trade-volume*, 2017.

[19] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A Kroll, and Edward W Felten. SoK: Research perspectives and challenges for Bitcoin and cryptocurrencies. *IEEE Symposium on Security and Privacy*, 2015.

[20] Taoufik Bouraoui. Stock spams: An empirical study on penny stock market. *International Review of Business Research Papers*, 5:292–305, 2009.

[21] BTC-e. *https://www.btc-e.com/*, 2017.

[22] Juan Caballero, Chris Grier, Christian Kreibich, and Vern Paxson. Measuring Pay-per-install: The Commoditization of Malware Distribution. In *Proceedings of the 20th USENIX Security Symposium*, Berkeley, CA, USA, 2011. USENIX Association.

[23] CIA. CIA World Factbook, Country Comparison: Internet Users.

[24] Coinbase.com. Buy and Sell Digital Currency. *https://www.coinbase.com/*, 2017.

[25] CryptoCurrency Market Capitalizations. 24 Hour Volume Rankings (Exchange). *https://coinmarketcap.com/exchanges/volume/24-hour/all/#BTC*, 2017.

[26] David Dagon, Cliff Changchun Zou, and Wenke Lee. Modeling Botnet Propagation Using Time Zones. In *Proceedings of the 13th Annual Symposium on Network and Distributed System Security*, 2006.

[27] Evan Duffield and Daniel Diaz. Dash: A Privacy-Centric Crypto-Currency. *https://github.com/dashpay/dash/wiki/Whitepaper*, 2015.

[28] Ittay Eyal and Emin Gun Sirer. Majority is not Enough: Bitcoin Mining is Vulnerable. http://arxiv.org/abs/1311.0243, 2013.

[29] Ittay Eyal and Emin Gun Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International Conference on Financial Cryptography and Data Security*, pages 436–454. Springer, 2014.

[30] Federal Reserve Bank of St. Louis. FRED Economic Data. *https://fred.stlouisfed.org/categories/32255?tg=gen*, 2017.

[31] John Paul Fraedrich. Signs and signals of unethical behavior. In *Business Forum*, volume 17, page 13. California State University, Los Angeles, School of Business and Economics, 1992.

[32] Goodin, Dan. Massive cryptocurrency botnet used leaked NSA exploits weeks before WCry. *ArsTechnica, https://arstechnica.com/information-technology/2017/05/massive-cryptocurrency-botnet-used-leaked-nsa-exploits-weeks-before-wcry/*, 2017.

[33] Chris Grier, Lucas Ballard, Juan Caballero, Neha Chachra, Christian J. Dietrich, Kirill Levchenko, Panayiotis Mavrommatis, Damon McCoy, Antonio Nappa, Andreas Pitsillidis, Niels Provos, M. Zubair Rafique, Moheeb Abu Rajab, Christian Rossow, Kurt Thomas, Vern Paxson, Stefan Savage, and Geoffrey M. Voelker. Manufacturing Compromise: The Emergence of Exploit-as-a-Service. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, October 2012.

[34] Hileman, Garrick and Rauchs, Michel. Global Cryptocurrency Benchmarking Study. *Cambridge Centre for Alternative Finance*, 2017.

[35] Inside Your Botnet. av.psybnc.cz (100k ngrBot hosted in France Paris Gandi). http://www.exposedbotnets.com/2011/11/avpsybnccz100k-ngrbot-hosted-in-france.html, November 2011.

[36] Inside Your Botnet. a.xludakx.com (ngrBot hosted in France Paris Gandi around 80k). http://www.exposedbotnets.com/2011/10/axludakxcomngrbot-hosted-in-france.html, October 2011.

[37] Inside Your Botnet. b.mobinil.biz (Silent BitCoin GPU Miner using Phoenix Miner). http://www.exposedbotnets.com/2011/07/bmobinilbizsilent-bitcoin-gpu-miner.html, July 2011.

[38] Inside Your Botnet. xD.a7aneek.net (80-100k ngrBotnet hosted in France Paris Gandi). http://www.exposedbotnets.com/2011/11/xda7aneeknet80-100k-ngrbotnet-hosted-in.html, November 2011.

[39] Inside Your Botnet. beast.darkogard.com (irc botnet hosted in Germany Frankfurt Am Main Sedo Domain Parking). http://www.exposedbotnets.com/2012/07/beastdarkogardcomirc-botnet-hosted-in.html, July 2012.

[40] Inside Your Botnet. d.xludakx.com (ngrBot hosted in Netherlands Amsterdam Leaseweb B.V.). http://www.exposedbotnets.com/2012/01/9521116562ngrbot-hosted-in-netherlands.html, January 2012.

[41] Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. Spamalytics: an Empirical Analysis of Spam Marketing Conversion. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 3–14, Alexandria, VA, October 2008.

[42] Aggelos Kiayias, Elias Koutsoupias, Maria Kyropoulou, and Yiannis Tselekounis. Blockchain mining games. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 365–382. ACM, 2016.

[43] KingAfurah. Ultimate Bitcoin Stress Test. *https://bitcointalk.org/index.php?topic=1094865.0*, 2015.

[44] Brian Krebs. Botcoin: Bitcoin Mining by Botnet. http://krebsonsecurity.com/2013/07/botcoin-bitcoin-mining-by-botnet/, 2013.

[45] Nicholas D Kristof. Where pimps peddle their goods. *The New York Times*, 11, 2012.

[46] Joshua Kroll, Ian Davey, and Edward Felten. The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries. In *Proceedings of WEIS 2013*, 2013.

[47] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoff Voelker, and Stefan Savage. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. In *Proceedings of the ACM Internet Measurement Conference*, 2013.

[48] Malte Möser and Rainer Böhme. Trends, tips, tolls: A longitudinal study of Bitcoin transaction fees. In *2nd Workshop on Bitcoin Research, affiliated with the 19th International Conference on Financial Cryptography and Data Security, Puerto Rico*, 2015.

[49] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[50] Namecheap. Namecheap now accepts Bitcoin. *https://www.namecheap.com/support/payment/bitcoin.aspx*, 2013.

[51] Ryan Naraine. Researchers Find Malware Rigged with Bitcoin Miner, 2011. http://www.zdnet.com/blog/security/researchers-find-malware-rigged-with-bitcoin-miner/8934.

[52] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies*. Princeton University Press, 2016.

[53] Nasdaq. Alphabet Inc. Class C Capital Stock Historical Stock Prices, 2017. http://www.nasdaq.com/symbol/goog/historical.

[54] Overstock.com. Bitcoin on Overstock.com. *https://www.overstock.com/bitcoin*, 2017.

[55] Pearson, Jordan. WikiLeaks Is Now a Target In the Massive Spam Attack on Bitcoin. *Motherboard, https://motherboard.vice.com/en_us/article/ezvw7z/wikileaks-is-now-a-target-in-the-massive-spam-attack-on-bitcoin*, 2015.

[56] Colin Percival. Stronger key derivation via sequential memory-hard functions. *The BSD Conference (BSDCan)*, pages 1–16, 2009.

[57] Colin Percival and Simon Josefsson. The scrypt password-based key derivation function. Technical report, 2016.

[58] Rebecca S Portnoff, Danny Yuxing Huang, Periwinkle Doerfler, Sadia Afroz, and Damon McCoy. Backpage and bitcoin: Uncovering human traffickers. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1595–1604. ACM, 2017.

[59] Quandl. Currency Exchange Rates - USD / EUR. *https://www.quandl.com/data/CUR/EUR*, 2017.

[60] Fergal Reid and Martin Harrigan. An analysis of anonymity in the Bitcoin system. In Yaniv Altshuler, Yuval Elovici, Armin B. Cremers, Nadav Aharony, and Alex Pentland, editors, *Security and Privacy in Social Networks*, pages 197–223. Springer New York, 2013.

[61] Dorit Ron and Adi Shamir. Quantitative Analysis of the Full Bitcoin Transaction Graph. In *Proceedings of Financial Cryptography 2013*, 2013.

[62] Christian Rossow, Dennis Andriesse, Tillmann Werner, Brett Stone-Gross, Daniel Plohmann, Christian J Dietrich, and Herbert Bos. SoK: P2PWNED: Modeling and Evaluating the Resilience of Peer-to-Peer Botnets. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2013.

[63] M. Bedford Taylor. Bitcoin and the age of bespoke silicon. In *2013 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*, pages 1–10, Sept 2013.

[64] Jovi Umawing. Fareit Goes Bitcoin Mining (ThreatTrack Security Labs IT Blog), 2013. http://www.threattracksecurity.com/it-blog/fareit-goes-bitcoin-mining/.

[65] James Wyke. The ZeroAccess Botnet - Mining and Fraud for Massive Financial Gain. Technical report, SophosLabs, 2012.

[66] Claud Xiao. Bitcoin Miner Malware.

[67] Yahoo. GOOG Historical Prices. *https://finance.yahoo.com/quote/GOOG/history?p=GOOG*, 2017.