

Analysis of Cryptocurrency Desktop Wallet Software

Preston Miller

Marshall University

Summer '14

Introduction

Often synonymous with providing anonymity for the acquisition of nefarious goods online, cryptocurrencies, like Bitcoin, have been brought to the forefront after gaining traction due to recent media attention. Increased exposure has publicized the utility of cryptocurrencies and spurred the production of new currencies, like Litecoin and Darkcoin. As more consumers turn to online markets, such as Silk Road, for the purchase of both legal and illegal goods, understanding the channels through which cryptocurrencies travel will be vital in future casework. In anticipation of this, Bitcoin, Litecoin and Darkcoin desktop wallet software was examined for forensic artifacts including software GUID, transaction logs, wallet addresses, access times and user and account settings. To ensure a holistic evaluation the examination was conducted on multiple operating systems in three states: fresh installation, intermediate use and deleted.

Basics of Cryptocurrency Design

Bitcoin's creation is credited to the alias Satoshi Nakamoto, whose identity is still unknown. The idea behind Bitcoin was initially proposed in October 2008 as a "purely peer-to-peer version of electronic cash" that would cut out the middle man, i.e. financial institutes¹. Satoshi's idea would

¹ Nakamoto, Satoshi. [Bitcoin: A Peer-to-Peer Electronic Cash System](http://nakamotoinstitute.org/bitcoin/). 31 October 2008. <<http://nakamotoinstitute.org/bitcoin/>>.

circumvent the financial institution's trust based model in favor of a cryptographic proof of work model.

The benefits of this model are twofold:

- Non-reversible transactions protecting sellers from fraud
- Agreement of address escrow mechanism to protect buyers

One issue that needed to be dealt with was double spending. In the trust model, a trusted third party verifies that the money has not already been spent before allowing a transaction. This obstacle led to Bitcoin's greatest innovation, the blockchain². The blockchain is essentially a public ledger that is made up of blocks containing all previous transactions of the currency (Fig. 1).

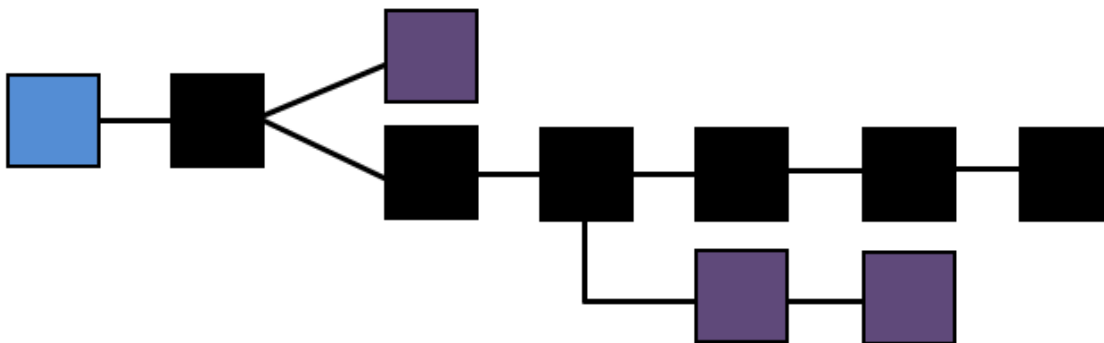


Figure 1. The blockchain is made up of blocks. Each block is made up of hundreds of transactions. The start of the chain, the Genesis block, is blue. The main chain is black and forks in the chain are purple, referred to as "orphan" blocks.

This blockchain prevents double spending by the verification process each transaction must undergo. This process is designed to take ten minutes to complete and is composed of these basic steps:

1. Assign the transaction to a block that is in a queue to be verified
2. Confirm coins were signed with the sender's address private key
3. Confirm coins have not already been spent by checking the blockchain
4. Repeat for all transactions in the block

² Bitcoin Wiki. [Block chain](https://en.bitcoin.it/wiki/Block_chain). 21 April 2014. <https://en.bitcoin.it/wiki/Block_chain>.

5. Calculate a difficult SHA256 hash of the block plus “nonce”
6. Add the block to the blockchain

All the transactions in a given block are considered verified once that block is followed by five blocks. If a block never reaches five following blocks, it is considered an “orphan,” as shown in Fig. 1 above.

The majority of the verification process is spent computing an arbitrarily difficult SHA256 hash of the block (step 5, above). Bitcoin uses the hashcash proof of work function for this step³. A SHA256 hash of the block must be calculated with a certain number of preceding zeroes. As more nodes are present (i.e., more Bitcoin users) the number of preceding zeroes increases, effectively increasing the difficulty of the calculation. After every 2,016 verified blocks a system automatically adjusts the difficulty of the hash if the average time of the verification process deviates from ten minutes. Difficulty increases as more Bitcoin nodes are added to the network (Fig. 6). Computers compete by appending a “nonce”, a random number, to the block and rehashing it until the correct hash is obtained. Once the correct hash is obtained, the nonce used is broadcasted to the network where each node appends the nonce to the block and hashes it, if a majority of the nodes agree then the block, and all of its transactions, is added to the chain. Once there are five blocks above that block, the transactions are verified. The verification process is referred to as “mining”^{4,5}.

³ Bitcoin Wiki. [Hashcash](https://en.bitcoin.it/wiki/Hashcash). 8 June 2014. <<https://en.bitcoin.it/wiki/Hashcash>>.

⁴ Perry, David. [Bitcoin Mining in Plain English](http://codinginmysleep.com/bitcoin-mining-in-plain-english/). 6 September 2012. <<http://codinginmysleep.com/bitcoin-mining-in-plain-english/>>.

⁵ Bitcoin Wiki. [Mining](https://en.bitcoin.it/wiki/Mining#The_Computationally-Difficult_Problem). 1 April 2014. <https://en.bitcoin.it/wiki/Mining#The_Computationally-Difficult_Problem>.

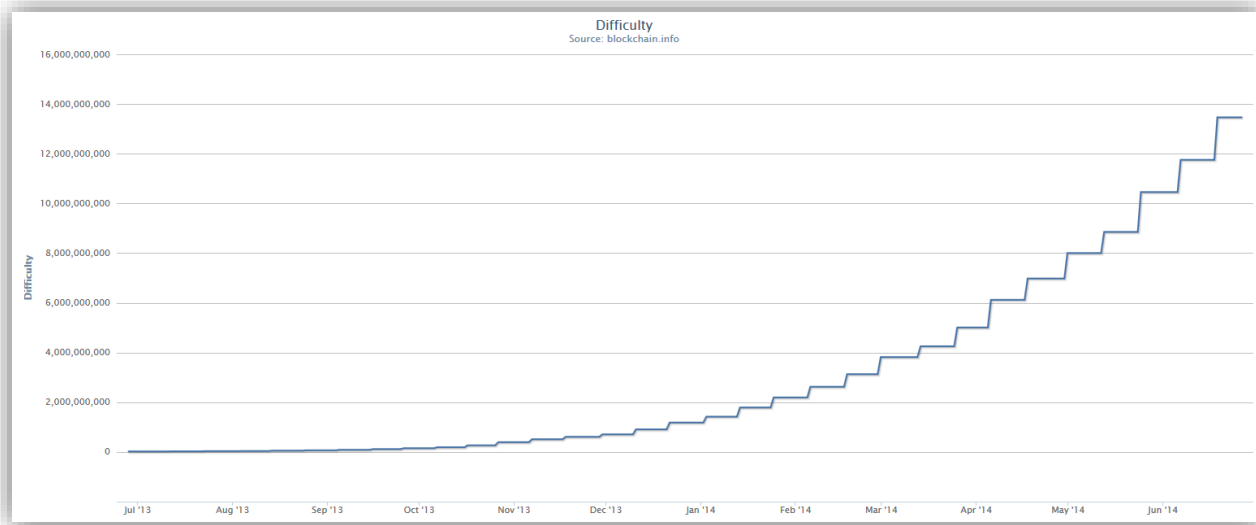


Figure 2. Difficulty is determined every 2,016 blocks and usually increases to prevent the verification process deviating from 10 minutes as more nodes join the network.

Miners are individuals who verify transactions by computing difficult hashes. Miners are incentivized to verify transactions with Bitcoin rewards. More miners increase the integrity of transactions by making it more difficult for a bad actor to have control of a majority of the verification process.

There are three ways of obtaining Bitcoins: purchase from an exchange, transfer among individuals, and mining. Mining has two main functions:

- Verify and add blocks to the blockchain
- Release new Bitcoins to the market

Once the verification processes is completed, the computer that calculated the correct hash is rewarded with freshly 'minted' Bitcoins called the coinbase. The current reward is roughly 25 Bitcoins per block. Users typically pay fees when making transactions. These fees will also be awarded to the miner to incentivize them to include their transactions in the block. These fees will be increasingly important to miners as the coinbase decreases over time. Miners work in pools, as the likelihood of calculating the hash on their own is slim. When a member of a pool calculates the winning hash the

reward is split amongst the group based on each individual's contributed computation power. To stimulate scarcity, Bitcoin has a finite amount of coins that can ever be created, 21 million Bitcoins. There are currently over 13 million coins in circulation. As more coins are mined the reward decreases (Table 1).

Table 1. Depicts diminishing returns on Bitcoins per block as more Bitcoins are in circulation. Each decrease in reward is referred to as a block reward-halving event.

Block	BTC/block	Year	Start BTC	BTC Added	End BTC	BTC Increase (%)	BTC Supply % of Limit
0	50.00	2009	0	2625000	2625000	Infinite	12.50
52500	50.00	2010	262500	2625000	5250000	100.00	25.00
105000	50.00	2011	5250000	2625000	7875000	50.00	37.50
157500	50.00	2012	7875000	2625000	10500000	33.33	50.00
210000	25.00	2013	10500000	1312500	11812500	12.50	56.25
262500	25.00	2014	11812500	1312500	13125000	11.11	62.50
315000	25.00	2015	1312500	1312500	14437500	10.00	68.75

Users who possess a full Bitcoin client act as nodes on the Bitcoin network⁶. Each node possesses its own up-to-date version of the blockchain. Each new block added to the chain contains the hash of the header of the block below it, as well as newly verified transactions, when a majority of the nodes verify the transactions and a set hash is computed (Fig. 3-5). This chain of blocks has one path leading to the first block created, the Genesis block. Forks in the chain are possible if two blocks are created within seconds of each other. Because new blocks always have a hash of the block below it,

⁶ Barber, Simon, et al. "Bitter to Better - How to Make Bitcoin a Better Currency." Financial Cryptography and Data Security (2012).

forks never merge further up the chain. Only blocks in the long chain will continue to grow. The unused block is referred to as an Orphan block. The transactions in the Orphan block are added to another block and go through the verification process again to get added to the long chain. Additionally, miners receive no reward for Orphan blocks. Transactions will not be verified until five other blocks follow a block. This way, modification of the blockchain is impractical due to the need to regenerate the hash of all the blocks above it.

Height	Age	Transactions	Total Sent	Relayed By	Size (kB)
307973	8 minutes	1553	11,879.93 BTC	Discus Fish	764.11
307972	42 minutes	327	2,615.49 BTC	BTC Guild	192.24
307971	49 minutes	589	2,960.05 BTC	GHash.IO	324.36
307970	57 minutes	946	7,214.04 BTC	BTC Guild	453.07
307969	1 hour 16 minutes	816	6,343.98 BTC	BTC Guild	440.46
307968	1 hour 30 minutes	262	1,167.27 BTC	Discus Fish	128.27

Figure 3. There are numerous websites that allow individuals to view the blockchain, such as blockchain.info. Each block in the blockchain is assigned a number in the chain and is verified along with the transactions within it.

One concern with the blockchain is that if 51% of the nodes are ever controlled by one or one group of individuals the system breaks. Only a majority of nodes are required to agree to verify a transaction. With 51% of the nodes, they can push through or deny any transaction. However, this inherent flaw is only a concern with large mining pools as an individual is unlikely to have the resources to possess 51% of the nodes at a given time.

Transactions Transactions contained within this block		
6a10efc5e048da5d3ef6d7e0d2fd1e8ffdb09e4667f8183ee86bc6729c195da0	2014-06-26 20:15:21	
No Inputs (Newly Generated Coins)	1KFHE7w8BhaEN... (F2Pool Mining Address)	25.24677656 BTC 25.24677656 BTC
d7e60b82ec09a16a3115cbd4d57a4fc83190cb9d3071d6c5648419588df18451	2014-06-26 19:52:42	
1HwRCRaBVeiEC4S2wq34P1g1vUIGQu5KR	1P0r6uz4IP294h9Dw3jQ2BVD5JP1QkckZ5 1HwRCRaBVeiEC4S2wq34P1g1vUIGQu5KR	1.7 BTC 9.81378235 BTC 11.51378235 BTC
1ab5d13931b6338298f99b2f5e419d464a3c916b7b2eb70671f37010345aa0	2014-06-26 19:49:09	
12ssZJSYwCsVBe2dv8pN9fa25Tn7m5jFz	1HSVYxCNVBkU6hgqwh7emgmdCbdpCyk92 12ssZJSYwCsVBe2dv8pN9fa25Tn7m5jFz	1 BTC 2.63772873 BTC 3.63772873 BTC
d5701d547366f8181848056d97c8051a91b28d5b4ada9c65e184353b278fcd00	2014-06-26 20:15:09	
1HxkTyu6DKJF1ePe3UISwCgaN7Ep6QTwx	1NBcTKC2673oq4Bep6uEchaPDnUYNyxKf 1HxkTyu6DKJF1ePe3UISwCgaN7Ep6QTwx	0.10048988 BTC 3.11819596 BTC 3.21868584 BTC

Figure 4. Each transaction in a block can be examined to see the addresses and amount of Bitcoins involved in each transaction.

Transaction View information about a bitcoin transaction

23972b7f1672bf30b326ff992c6a126e9cce273f0296718203ff665d823c4db6

No Inputs (Newly Generated Coins) → 1CjPRZ5ZSyWk6... (ghash.io) - (Unspent) 25.08264529 BTC
2 Confirmations 25.08264529 BTC

Summary		Network Propagation (Click to view)	
Size	157 (bytes)		
Received Time	2014-06-26 20:51:38		
Reward From Block	307977		
Scripts	Hide scripts & coinbase		
Relayed by IP	176.9.227.249 (whois)		
Visualize	View Tree Chart		

Figure 5. Each individual transaction in a block can be examined for even more data including “Received Time”.

Bitcoin for Individual Users

Bitcoin is categorized as a cryptocurrency because of its cryptographic roots^{7,8}. Bitcoins are stored in web, mobile, or desktop wallets and, within a wallet, are assigned to certain user created addresses. For each address, a public and private key is generated. Users can dictate which address to use for any transaction. This is confirmed in the verification process with the user's public key that is mathematically related to the private key.

Bitcoins wallet addresses and public and private keys are mathematically linked through a series of algorithms (Fig. 6). This process is described in detail in Ken Shirriff's blog⁹. Private keys are randomly generated 32-byte numbers. Bitcoin uses Elliptic Curve Digital Signature Algorithm (ESDCA) cryptography to generate a 64-byte public key with a 0x04 prefix from the private key. The SHA-256 and RIPEM 160 algorithms create a 20-byte hash of the public key. The address associated with the public and private key is simply a 1 byte prefix, the custom base58 encoding of the 20-byte public key hash, and a 4 byte check code. This process can be replicated forward starting with a random private key to arrive at an address. However, it has not been proven possible to use an address and derive its private key. In a transaction, the sender signs the coins with the used address' private key. The blockchain then confirms that the coins being used have not already been spent, through the verification process.

⁷ Katz, J and Y Lindell. Introduction to Modern Cryptography. CRC Press, 2007.

⁸ IEEE. Standard Specifications For Public-Key Cryptography. 10 October 2008. <<http://grouper.ieee.org/groups/1363/>>.

⁹ Shirriff, Ken. Bitcoins the hard way: Using the raw Bitcoin protocol. <<http://www.righto.com/2014/02/bitcoins-hard-way-using-raw-bitcoin.html>>

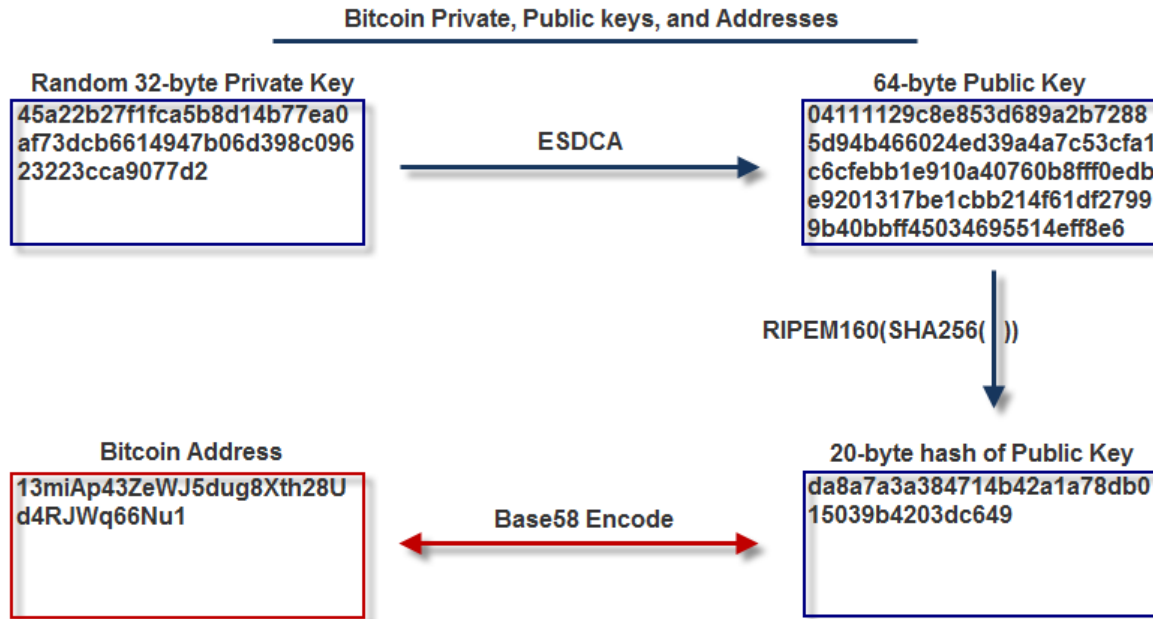


Figure 6. Demonstrates the process through which an address is generated. This process is primarily only one way. To protect the address used in this figure, the private and public key values were made up.

Litecoin

Litecoin is based on Bitcoin's node system and was created by Charles Lee in October 2011¹⁰. The currency was created to make improvements on the Bitcoin system. Overall, the currency works in the same manner as Bitcoin. Litecoin has a peer-to-peer network where each node has an updated blockchain. Litecoins are stored in wallet addresses with public and private keys. The sender's private key is used to sign the currency when making a transaction. Verification proceeds with miners packaging transactions in blocks, confirming a sender's legitimate private key was used to sign the currency by comparing it to the sender's public key, and processing a difficult hash. Miners that calculate the correct hash are rewarded with Litecoins. Litecoin has four main variations on the Bitcoin model¹¹:

- Scrypt based hash

¹⁰ Litecoin Wiki. [Litecoin](https://litecoin.info/Litecoin). 20 January 2014. <<https://litecoin.info/Litecoin>>.

¹¹ Litecoin Wiki. [Comparison between Litecoin and Bitcoin](https://litecoin.info/User:Iddo/Comparison_between_Litecoin_and_Bitcoin). 22 January 2014. <https://litecoin.info/User:Iddo/Comparison_between_Litecoin_and_Bitcoin>.

- Four times faster verification time
- More frequent difficulty adjustments
- Four times total Litecoin limit

Script is a memory-hard function whose purpose is to maximize the amount of memory it uses for every process¹². Script was originally created to make brute forcing a password hash more time consuming, however it has been implemented in Litecoin to decrease entry cost. Litecoin is more memory intensive rather than just arithmetic based, like Bitcoin. Application-specific integrated circuits, ASICs, are chips that are designed for specific purposes, such as Bitcoin mining. ASICs, which are more commonly used in Bitcoin, would be considerably more expensive for Litecoin mining. Additionally, return benefits from ASIC hardware would also be decreased by 10-fold in Litecoin mining. CPU mining is no longer truly viable with Bitcoin. It is, however, still an option for Litecoin miners. The difference between CPU vs. GPU mining in Litecoin is less extreme than in Bitcoin and because of these factors; Litecoin is more inclusive in that relatively inexpensive hardware can be used to productively mine.

Litecoin boasts a two and a half minute verification time. This also means that miners are rewarded more frequently as blocks are verified more often. Because the 2,016 blocks are processed four times quicker than in Bitcoin, this means that forks in the blockchain and difficulty readjustments happen four times more frequently. As forks happen more frequently, there is a greater waste of computational power as nodes are more likely to hash orphan blocks before the location of the correct block is broadcasted. With the potential for there to be 84 million Litecoins in circulation, the market cap and purchasing power of Litecoin would surpass Bitcoin, if the value of a Litecoin were greater than a quarter of a Bitcoin; however, currently a Litecoin is only worth two percent of a Bitcoin.

Darkcoin

¹² Percival, Colin. "Stronger Key Derivation Via Sequential Memory-Hard Functions." (2009).

Darkcoin is a relative newcomer to the cryptocurrency field, but has grown quickly perhaps owing to its increased focus on anonymity¹³. It does this through the custom implementation of CoinJoin, called Darksend. CoinJoin is a popular structure that many mixing services are based on. While CoinJoin is not built into Bitcoin, Dark wallet is a CoinJoin implementation built on the Bitcoin network. CoinJoin works by mixing multiple transactions into a pool and then sending money from that pool to intermediaries, other pools, and eventually to the recipients. However, this mixing service is provided by a centralized server, which if it were compromised 'anonymous' transactions would be revealed. Another mixing service based on CoinJoin is ShareCoin and was recently discovered to not be as anonymous as users believed¹⁴. Since Darksend is built into the Darkcoin client, their mixing service is decentralized and is unlikely that enough users would be compromised to threaten anonymity on the server. As yet, no weakness has been discovered in Darksend's capability to provide anonymous transactions.

Darkcoin has three unique properties: denominations, master nodes, and an inflationary currency. As of Release Candidate 4, to use Darksend users must allow their wallet to 'anonymize' their Darkcoins first and then only send transactions using the anonymized Darkcoins. The anonymizing process is configurable and shows up in the wallet's transaction log with the type "Darksend Denominate" with an associated fee of 0.001 Darkcoins for each mixing cycle. Master nodes are simply nodes that provide the mixing service using the Darksend technology. These master nodes are elected by a pseudorandom algorithm and receive twenty percent of the block reward. Becoming a master node

¹³ Duffield, Evan and Kyle Hagan. "Darkcoin: Peer-to-Peer Crypto-Currency with Anonymous Blockchain Transactions and an Improved Proof-of-Work System." (2014).

¹⁴ Southurst, Jon. [Blockchain's SharedCoin Users Can Be Identified, Says Security Expert](http://www.coindesk.com/blockchains-sharedcoin-users-can-identified-says-security-expert/). 10 June 2014. <<http://www.coindesk.com/blockchains-sharedcoin-users-can-identified-says-security-expert/>>.

is meant to be difficult, a node must have at least 1,000 Darkcoins and the user must be technically savvy enough to set up the master node.

Darkcoin utilizes a two and a half minute verification process, similar to Litecoin. There are only a maximum of 22 million Darkcoins that can be created. Darkcoin uses X11 as its proof of work hash. X11 is a chained hashing algorithm that is composed of 11 different hashes. This hashing function is more complicated than the SHA256 hash in Bitcoin and prevents current ASICs from being utilized efficiently. Darkcoin, like Litecoin, favors CPU and GPU mining. Darkcoin reevaluates the difficulty of the hash every block using DarkGravityWave. In contrast to Bitcoin and Litecoin, Darkcoin features a block reward curve where reward decreases gradually as more blocks are mined. The reward curve will eventually reach a steady 7% annual release of Darkcoins. This makes Darkcoin an inflationary currency, unlike the deflationary Bitcoin and Litecoin.

This introduction is not meant to address every nuance of these currencies, but should provide sufficient working knowledge of these cryptocurrencies for this report. A reference table below summarizes the differences between the three cryptocurrencies under examination. (Table 2).

Table 2. Summarizes the main differences between the three cryptocurrencies examined in this paper.

	Bitcoin	Litecoin	Darkcoin
Current value (USD)	600	9	9
Coins in circulation	13,000,000	30,000,000	4,500,000
Currency type	Deflationary	Deflationary	Inflationary
Maximum coins	21,000,000	84,000,000	22,000,000
Verification time (min)	10	2.5	2.5
Hash used	SHA256	Scrypt	X11

Reward	25 BTC	50 LTC	5 DRK
Reward type	Reward halving	Reward halving	Reward curving
Difficulty adjustment	2,016 transactions	2,016 transactions	1 transaction

Materials & Methods

Materials used for this research project include:

- One external hard drive
- VM Workstation and ISO files for operating systems analyzed
- Bitcoins, Litecoins, and Darkcoins for trading purposes
- Access to forensic software programs (listed in Table 3)

Virtual machines (VMs) were created and maintained by VMware Workstation version 9.0.0.

Each operating system analyzed had a negative control and three states: no use, intermediate use, delete. Each cryptocurrency wallet had a unique VM to avoid potentially confusing artifacts from a different wallet software. Negative controls were created with fresh installations of the operating systems being analyzed. The control's memory and hard drive were preserved for later analysis by duplicating the VM's VMEM and VMDK files, respectively. Controls were then copied using VMware's "Full Clone" option to create independent VMs for testing.

The first batch of clones, labeled the "Before" state, had their respective wallets installed and up to date with the blockchain, but were not used to trade. Clones of the "Before" state, labeled the "After" state, were used to conduct trades with a variety of other wallets. In order to obtain a comprehensive dataset, wallets sent to and received from included desktop, mobile, and web wallets. Clones of the "After" state, labeled the "Delete" state, employed a variety of deletion methods and the resulting artifacts upon deletion of the software.

On Windows VMs, Regshot version 1.8.3 was used to analyze differences in the registry for all VM states. Additionally, Wireshark version 1.6.1 captured network activity for all VM states. Each VM's VMEM file was preserved before shutting it down. Afterwards a copy of their VMDK was also preserved for analysis. VMDK files were converted to an E01 image with FTK Imager Version 3.1.3.2. A variety of tools were used for memory and hard drive analysis (Table 3).

Table 3. Tools used, version, and their purpose in analysis.

Software	Version	Purpose
Memory Analysis		
FTK Imager	3.1.3.2	String searching
Volatility	2.3.1	Overall memory analysis toolkit
MantaRay	1.2.9	Overall memory analysis toolkit
Hard Drive Analysis		
EnCase	6.17.0.90 & 7.08.00.137	Overall case analysis toolkit
Linkil	1.0.0.1	Lnk file analysis

Contemporaneous notes were maintained throughout the entire research project. Where possible, results were placed in an Excel workbook on separate spreadsheets for convenience. Relevant artifacts searched for include: transactions with date and time stamps, GUID, user settings, installation information, and usage.

The "Deleted" state simulated deletion of cryptocurrency software and their artifacts using multiple methods. The methods used were: uninstall program with the provided uninstaller or CCleaner version 4.11.4619 uninstall and clean modules. The memory and hard drive were preserved and analyzed to identify any new or remaining artifacts resulting from a range of shallow to deep deletion activity.

Results & Discussion

Bitcoin – MultiBit 0.5.18

Multiple transactions were conducted with the MultiBit 0.5.18 client on Windows 7 and Ubuntu 14.04 VMs. After imaging and analysis of the virtual drives, the resulting artifacts were similar on both

operating systems. The majority of pertinent artifacts were found in C:\Users\[User]\AppData\Roaming\MultiBit and /home/[User]/MultiBit directories on Windows and Ubuntu, respectively. There was also a MultiBit-0.5.18 folder in C:\Program Files(x86) and /home/[User] directories. However the MultiBit-0.5.18 folder did not contain any user activity artifacts. Limited transaction data was stored in multibit .info files, located in the previously mentioned directories (Fig. 7). In my research, these .info files contained only transactions sent from the wallet.

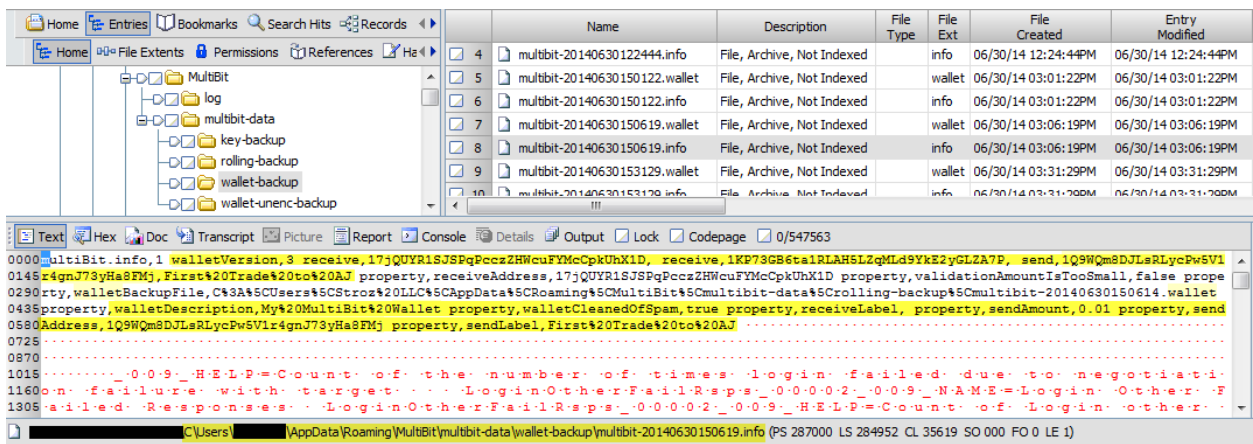


Figure 7. The multibit .info files stored in the MultiBit\multibit-data\wallet-backup\ directory contain some transaction data. New .info files appear to be generated as part of MultiBit’s wallet backup service.

The MultiBit 0.5.18 and the Uninstall MultiBit 0.5.18 Ink files contained the SID of the account that had the application installed (Fig. 8). This information is helpful in determining ownership of the software program.

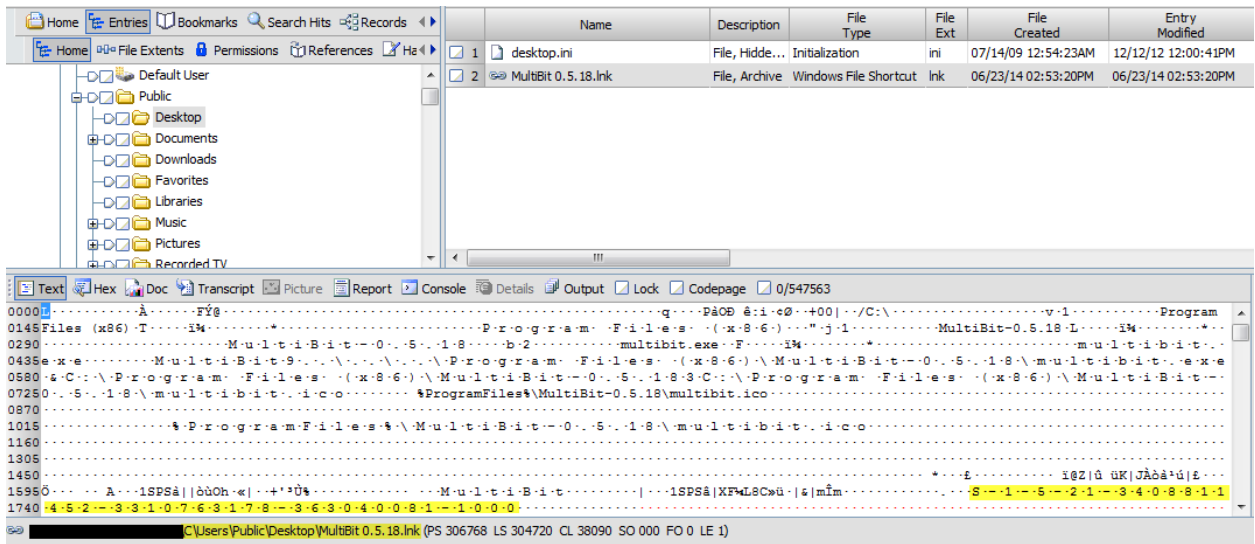


Figure 8. The MultiBit 0.5.18 and Uninstall MultiBit 0.5.18.lnk files contain the SID of the user's account that the program is installed on.

The multibit.log file found in the MultiBit\log\ folder on both Windows and Ubuntu VMs contains a wealth of information including program start and stop times and detailed transaction logs. Time stamps in the log were primarily in local time, unless otherwise stated. For example, when a user starts MultiBit, a “Starting MultiBit at” followed by a date and time stamp in UTC appears in the log. The last entry added to the log when a user closes MultiBit is “Making request to shut down socket” preceded by a local time stamp (Fig. 9). It is possible for a start entry to not be followed by a corresponding stop entry, and this indicates improper termination of the application such as a forced shutdown of the operating system. Only the most recent transactions will be found in the log as it has a finite storage capacity.

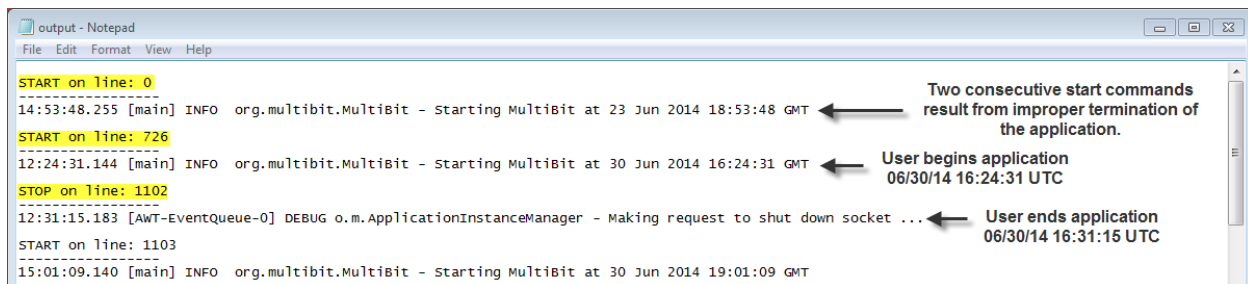


Figure 9. Start and stop commands from the multibit.log file can help an examiner establish a user activity timeline.

Detailed transaction data was stored in the multibit.log file. Each transaction sent from the wallet begins with a “SendBitcoinConfirmAction” preceded by a local timestamp. The entries following contain the transaction hash, addresses, and the amount of BTC involved. Note, to obtain the public addresses, one must convert the RIPEM-160/SHA-256 representation of the public address found in the log into Base58. This process will be demonstrated in the network capture section.

A “Peer announced new transaction” is indicative of a transaction being received by the wallet. Like the above, a local timestamp, transaction hash, addresses, and the amount of BTC involved are preserved (Fig. 10). A python script searching for the terms indicating start and stop activity, and transactions would be ideal for analyzing this lengthy log file.

```

15:05:53.906 [AWT-EventQueue-0] DEBUG org.multibit.network.MultiBitService - MultiBitService#sendCoins - Sent coins has completed
15:05:53.906 [AWT-EventQueue-0] DEBUG org.multibit.network.MultiBitService - MultiBitService#sendCoins - Sent coins. Transaction hash is 9ad3169a
15:05:53.906 [AWT-EventQueue-0] DEBUG org.multibit.network.MultiBitService - Added bitcoinController 20511156 as listener to tx = 9ad3169acc99e22
15:05:53.906 [AWT-EventQueue-0] DEBUG org.multibit.file.FileHandler - Start of securedelete

TRANSACTION Received on line: 1699
15:19:11.161 [NioClientManager] INFO com.google.bitcoin.core.MemoryPool - [83.40.36.156]:8333: Peer announced new transaction [1] 4146a0b6fdebb2
15:19:12.081 [NioClientManager] DEBUG com.google.bitcoin.core.wallet - Saw relevant pending transaction 4146a0b6fdebb2235d05de99e5de8ff7dfb6c14
in [30440220421d7f587a9f8d6e7a8cc286a39bf1d5dced3a3b5b2bb1ea0bff56bae76dcffa02204cec588a4e3c22caf6ce15abd0f7c5aa875a62b9f9b05f6850f5f9e062b5e
outpoint:9ad3169acc99e22e2f8f9f26d5b349e1b986985723e1d762dc3ceefe7696788c:0
out DUP HASH160 [c9a074d45ac907c1d861d3855602e272ceb2b5e0] EQUALVERIFY CHECKSIG 0.0095 BTC ← Output Addresses and Values
out DUP HASH160 [fd6ec54b05f81ab98e8f808bf2c60eb6791f4782] EQUALVERIFY CHECKSIG 0.0004 BTC

15:19:12.081 [NioClientManager] INFO com.google.bitcoin.core.Peer - [83.40.36.156]:8333: Downloading dependencies of 4146a0b6fdebb2235d05de99e5d

TRANSACTION Sent on line: 1773
15:27:19.785 [AWT-EventQueue-0] DEBUG o.m.v.s.a.SendBitcoinConfirmAction - just about to complete the tx (and calculate the fee)...
15:27:19.785 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - Completing send tx with 1 outputs totalling 1000000 satoshis (not includin
15:27:19.785 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - with 0.0134103 coins change
15:27:19.785 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - with a fee of 0.0001
15:27:19.785 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - completed: 2f495764adff13ee641eb055f943542198e0aa5dbc1c421a1c5b22982eb
in
outpoint:9ad3169acc99e22e2f8f9f26d5b349e1b986985723e1d762dc3ceefe7696788c:1 hash160:49d5d227d6143d21def6ca5e165df97f3cedffb0 ← Input Address
out DUP HASH160 [1e64eb9041df47cb0186b31e5b6be1c825447cb9] EQUALVERIFY CHECKSIG 0.01 BTC ← Output Addresses and Values
out DUP HASH160 [c9a074d45ac907c1d861d3855602e272ceb2b5e0] EQUALVERIFY CHECKSIG 0.0134103 BTC

15:27:19.785 [AWT-EventQueue-0] DEBUG o.m.v.s.a.SendBitcoinConfirmAction - The fee after completing the transaction was 10000
15:27:25.401 [AWT-EventQueue-0] DEBUG org.m.v.s.action.SendBitcoinNowAction - Sending from wallet C:\Users\Stroz LLC\AppData\Roaming\MultiBit\multi
in
outpoint:9ad3169acc99e22e2f8f9f26d5b349e1b986985723e1d762dc3ceefe7696788c:1 hash160:49d5d227d6143d21def6ca5e165df97f3cedffb0
out DUP HASH160 [1e64eb9041df47cb0186b31e5b6be1c825447cb9] EQUALVERIFY CHECKSIG 0.01 BTC
out DUP HASH160 [c9a074d45ac907c1d861d3855602e272ceb2b5e0] EQUALVERIFY CHECKSIG 0.0134103 BTC

15:27:25.401 [AWT-EventQueue-0] DEBUG org.multibit.network.MultiBitService - Ping: [98.186.167.15]:8333
15:27:25.526 [AWT-EventQueue-0] DEBUG org.multibit.network.MultiBitService - MultiBitService#sendCoins - just about to send coins
15:27:25.900 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - committx of 2456f14833634389d12380be0215ba55002ac18f67cf88c154ba6b7f46a688
15:27:25.900 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - marked 9ad3169acc99e22e2f8f9f26d5b349e1b986985723e1d762dc3ceefe7696788c:
15:27:25.900 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - 9ad3169acc99e22e2f8f9f26d5b349e1b986985723e1d762dc3ceefe7696788c_prevtx
15:27:25.900 [AWT-EventQueue-0] INFO com.google.bitcoin.core.wallet - ->pending: 2456f14833634389d12380be0215ba55002ac18f67cf88c154ba6b7f46a6883
Transaction Hash
Transaction Hash

```

Figure 10. Transactions sent from the wallet appear after a “SendBitcoinConfirmAction” entry. Transactions to the wallet appear after a “Peer announced new transaction” entry. These transactions contain the time, transaction hash, address inputs and outputs, and amount traded. This can also be compared with the start and stop times to determine transactions in particular sessions of the application.

RecentFileCache.bcf in the C:\Windows\AppCompat\Programs\ directory is another source that can reference recent execution of an application (Fig. 11). Previous research on RecentFileCache.bcf suggests that listed executables are typically new to the system and recently run by the user¹⁵.

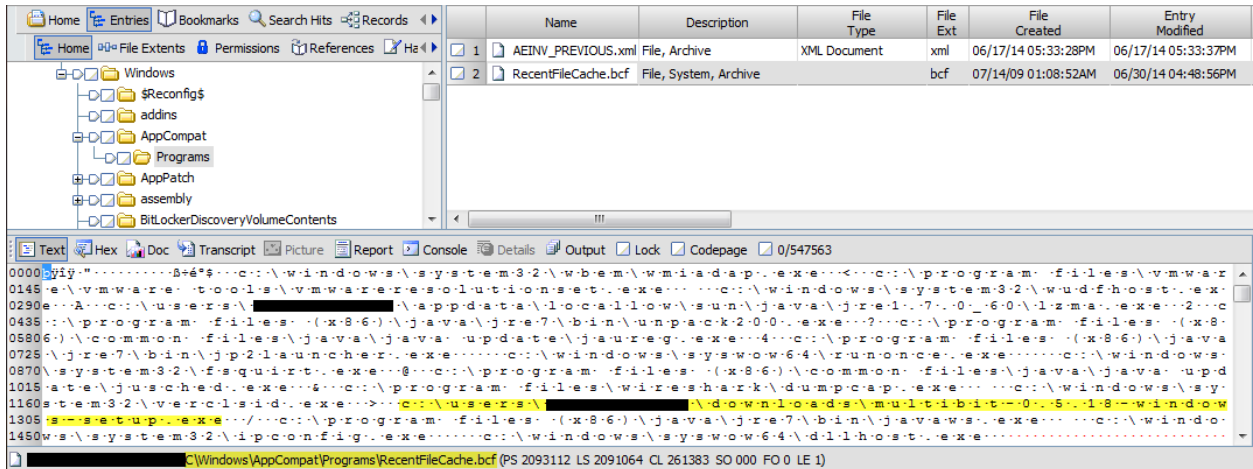


Figure 11. RecentFileCache.bcf contains recently executed applications and may contain reference to an executable even if it is no longer present on the system.

There are registry keys present in the SOFTWARE hive related to MultiBit, however no valuable user information was apparent. The following keys were associated with MultiBit installation:

- HKLM\SOFTWARE\Classes\bitcoin
- HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\MultiBit
0.5.18

After thorough analysis, MultiBit was uninstalled on both Windows and Ubuntu VMs. In Windows, the only pertinent files removed were lnk files and registry keys. In both cases, the MultiBit directory containing the .info and multibit.log files were still present. Equally as important, a log file titled izpack appended with a series of numbers, is created when the user uninstalls MultiBit (Fig. 12). This file appears in the C:\Users\[User]\AppData\Local\temp and /tmp/ directories on Windows and Ubuntu,

¹⁵ Harrell, Corey. [Revealing the RecentFileCache.bcf File](http://journeyintoir.blogspot.com/2013/12/revealing-recentfilecachebcf-file.html). 2 December 2013. <<http://journeyintoir.blogspot.com/2013/12/revealing-recentfilecachebcf-file.html>>.

respectively. IzPack is a packaging tool for Java applications. MultiBit's universal Java installer is based off of IzPack and is most likely the source of this deletion artifact. No reference to IzPack was found on the system prior to uninstalling MultiBit.

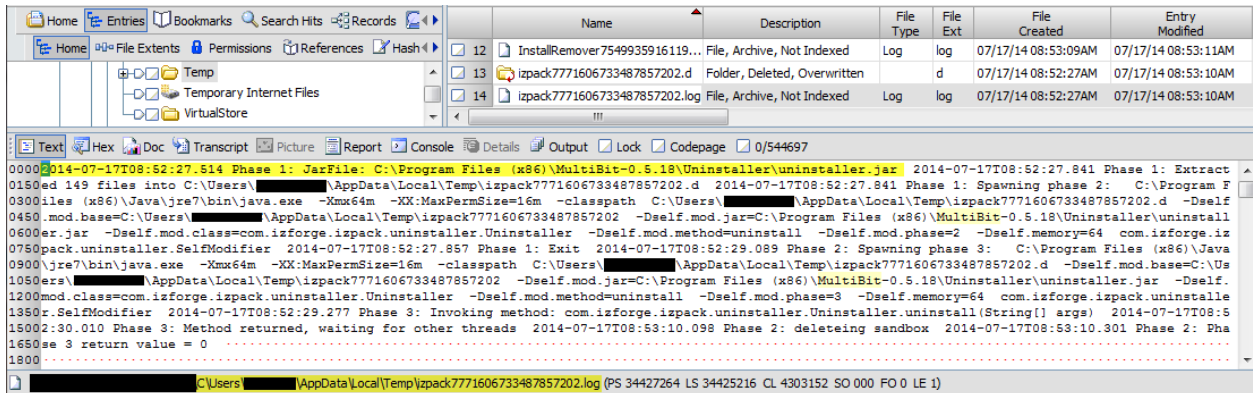


Figure 12. The izpack log appears to be generated upon the user uninstalling the MultiBit program from their system.

These artifacts can be used to determine the following: which account the program is installed on, recent usage, user activity timeline, date and time of uninstallation of the program, and detailed transaction logs.

Due to issues with the VM's VMEM files, string searching was the most extensive memory analysis possible. This revealed wallet addresses and sent/received address references, but did not allow for the recovery of transaction data (Fig. 13). This information could be used to indicate that MultiBit had been used during the current session and provide the examiner with additional addresses associated with the account.

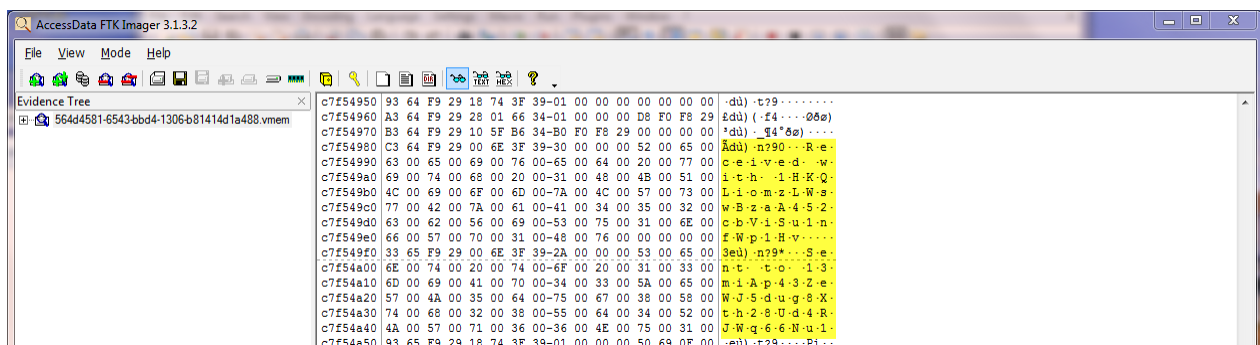


Figure 13. Received with and sent to addresses can be found in RAM. This can provide examiners with recent additional addresses that are associated with the account.

Capturing Bitcoin network traffic contained a great amount of transaction information. The latest version of Wireshark (Version 1.10.0) contains a Bitcoin protocol – greatly assisting the examiner in analyzing Bitcoin network traffic. The Bitcoin protocol, operating from port 8333, uses many message types, the most important for examinations being inv and tx¹⁶. The inv messages are typically the first sign of a transaction and an indication that the transaction has initially been accepted by the network. inv represents a node on the network advertising the knowledge of either a transaction (type 1) or a block (type 2) (Fig. 14). These nodes will use the same Payload checksum for this specific object. For message type 1 inv, the transaction hash is represented in Big Endian stored in the Data hash field.

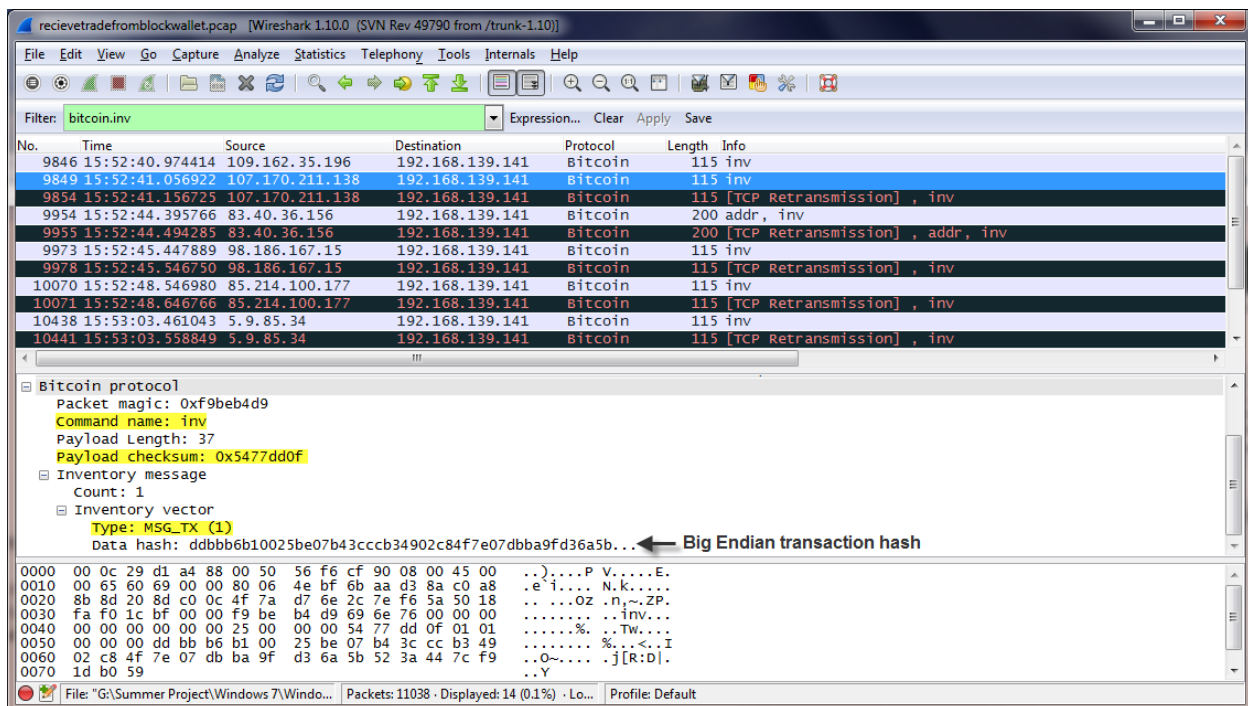


Figure 14. The inv message represents a node (107.170.211.138) acknowledging an object – typically a transaction (type 1) or a block (type 2). All inv and getdata messages for this particular transaction will share the same payload checksum.

The inv message is followed by a getdata message, which contains a matching payload checksum to its parent inv and the transaction hash. The getdata message is a response from the host to the node

¹⁶ Wiki, Bitcoin. [Protocol specification](https://en.bitcoin.it/wiki/Protocol_specification). <https://en.bitcoin.it/wiki/Protocol_specification>.

acknowledging the transaction (Fig. 15).

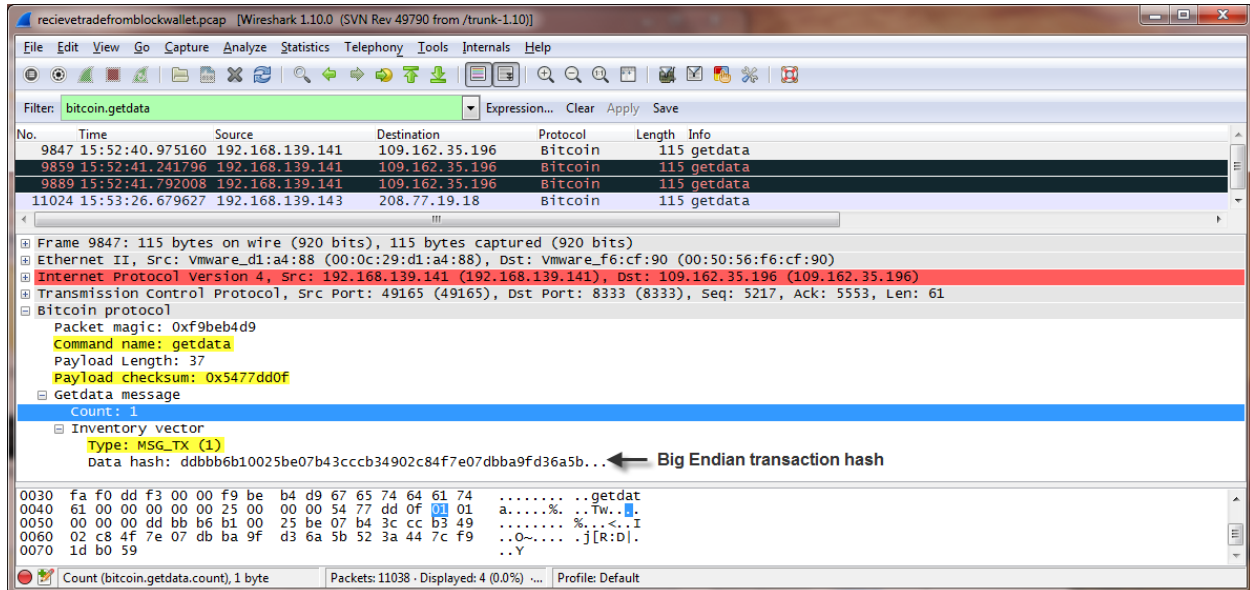


Figure 15. An inv will result in a getdata reply from the host (192.168.139.141) to the node (109.162.35.196).

The node then sends a tx message with the transaction details corresponding to the transaction hash in the previous inv and getdata messages. The tx message will have a payload checksum matching that of the first 4-bytes of the inv and getdata Data hash value. The tx message contains most of the information publically available on the blockchain (Fig. 16).

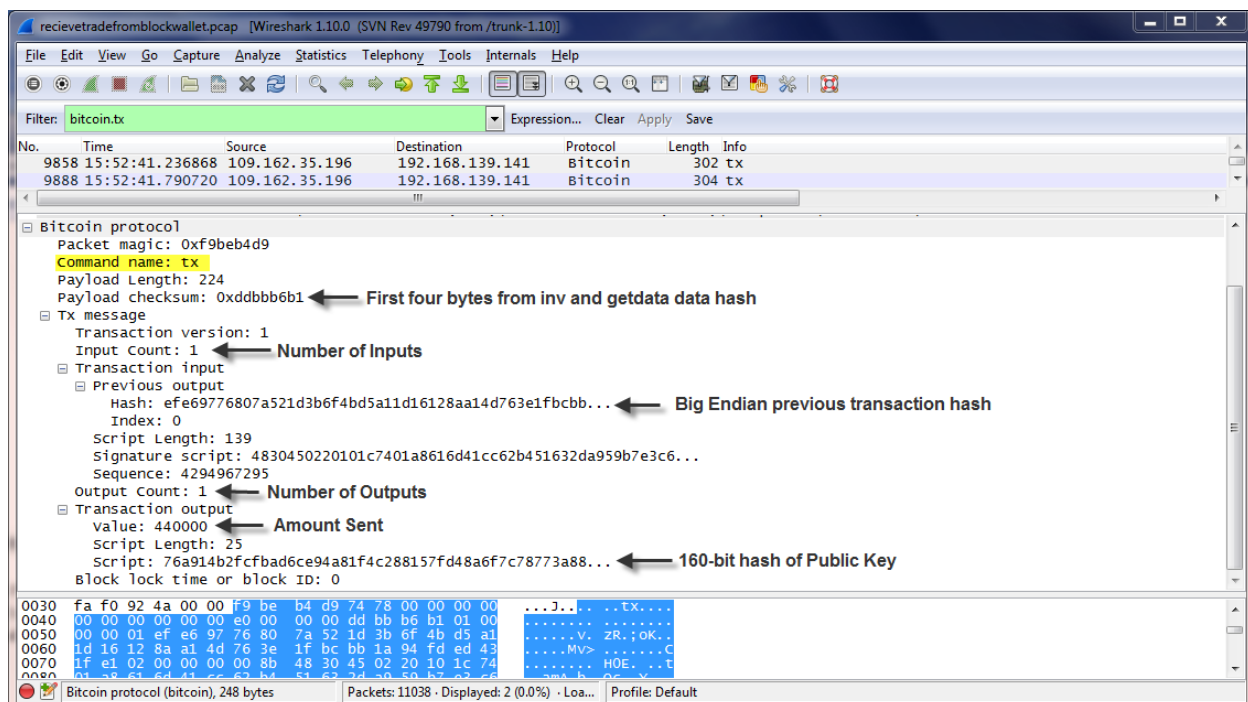


Figure 16. The tx message from the node (109.162.35.196) is in reply to the getdata message from the host (192.168.139.141). This message will have a payload checksum matching the first 4-bytes of the data hash in the previous inv and getdata messages. The tx message contains most of the information that would be publically viewable on the blockchain.

The Input and Output Count fields indicate the number of addresses involved. The “Previous output Hash” is the previous transaction hash in big Endian of each input address. Each output address has the corresponding amount sent and the address sent to in the “Value” and “Script” fields, respectively. It should be noted that the actual value sent and the value observed in Wireshark is a difference of 1×10^8 . Additionally, the “Script” value must first be processed to arrive at the address as seen on the blockchain. This process involves taking the 25-byte “Script” value from Wireshark and removing the first 3-bytes and the last 2-bytes and running the remaining 20-bytes through a series of hashes (Fig. 17). The 5-bytes removed are simply Bitcoin protocol artifacts that are not necessary to decode the address.

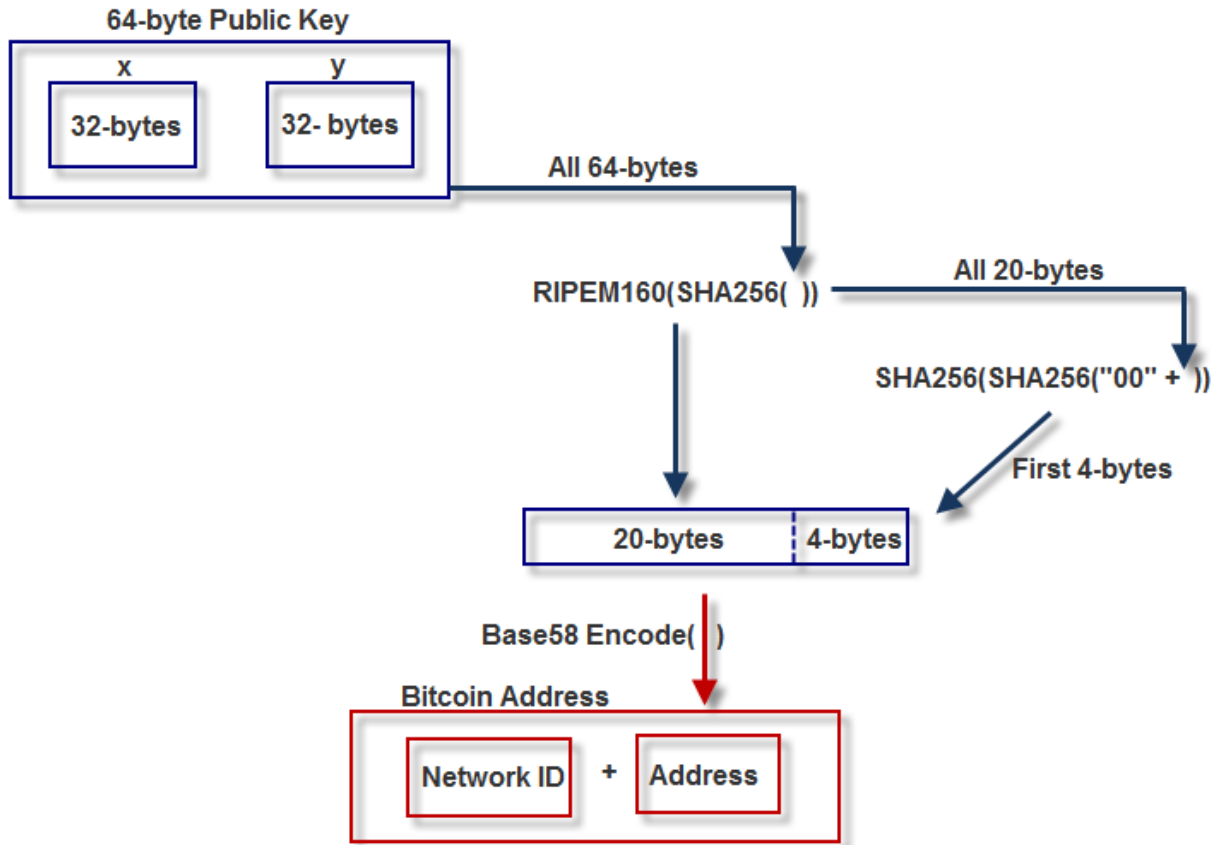


Figure 17. A more detailed version of Fig. 6 starting with the generated ESDCA public key, which is made up of its (x, y) position on the curve. The Script value from Wireshark is already the 20-byte product from the RIPEM160 and SHA256 hash of the public key. After removing the first 3-bytes and last 2-bytes from the Wireshark Script value the remaining 20-bytes must be processed. First a checksum must be calculated by sequential SHA-256 hashing of the 20-byte value with a prepended 00. The first 4 bytes of this value is appended to the original 20-bytes and base58 encoded. Then a Network ID must be prepended to arrive at the actual address. For Bitcoin, the Network ID is often 1 or 3. It is recommended that a script is created to automate this process.

There are many artifacts generated when a user uses MultiBit 0.5.18. The multibit.log and network captures contain the most valuable information to an examiner. Both of these will contain information needed to identify transactions on the user's computer. The multibit.log, RecentFileCache.bcf, and IzPack files can also help with creating a user activity timeline from daily activity to date of uninstallation. The MultiBit lnk files can pinpoint on which user account the application is installed. These artifacts not only assist the examiner with the MultiBit client, but in the case of the network captures, any client or even cryptocurrency using the Bitcoin protocol.

Litecoin – Litecoin 0.8.7.2

The Litecoin wallet software shares many similar artifacts with MultiBit. The main folder for Litecoin is in the C:\Users\[User]\AppData\Roaming\Litecoin and /home/[User].litecoin directories for Windows 7 and Ubuntu 14.04, respectively. The wallet.dat file located in this directory contains references to addresses received on and sent to (Fig. 18).

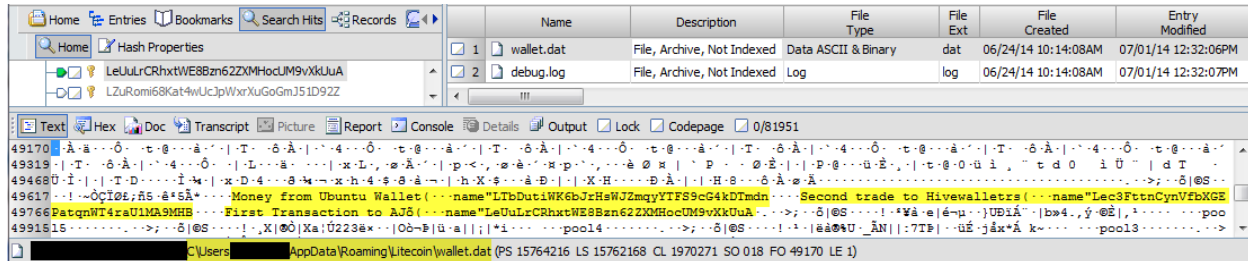


Figure 18. The wallet.dat file contains references to addresses received on and sent to. However, other transaction details such as a timestamp or an amount are not present.

On Windows, the Litecoin and Uninstall Litecoin Ink files contained the SID of the user account they were installed on (Fig. 19).

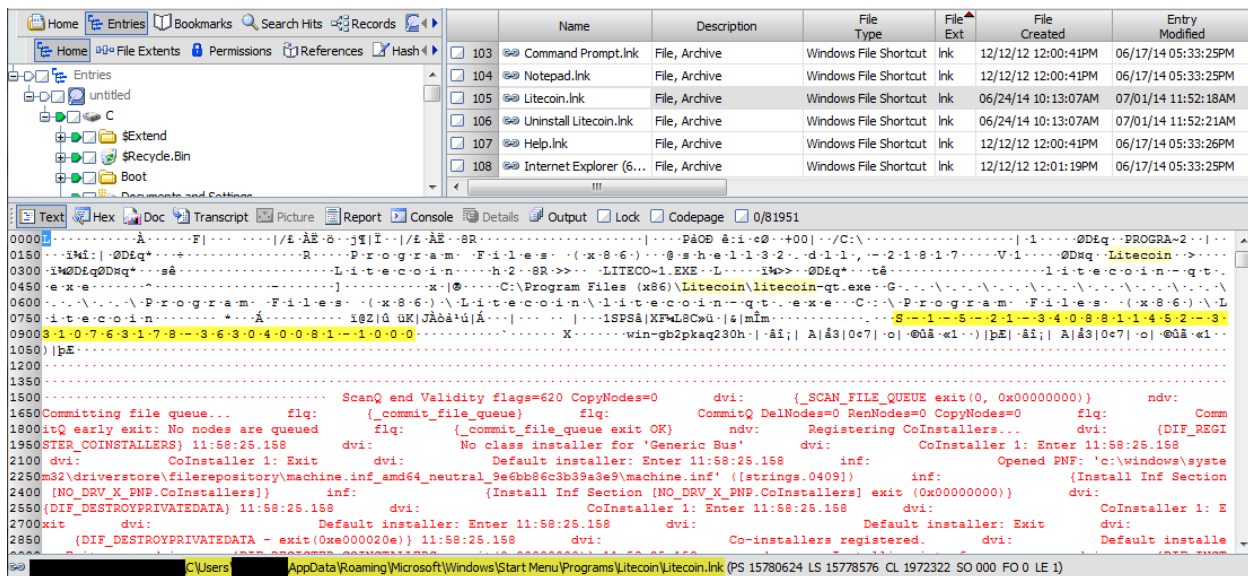
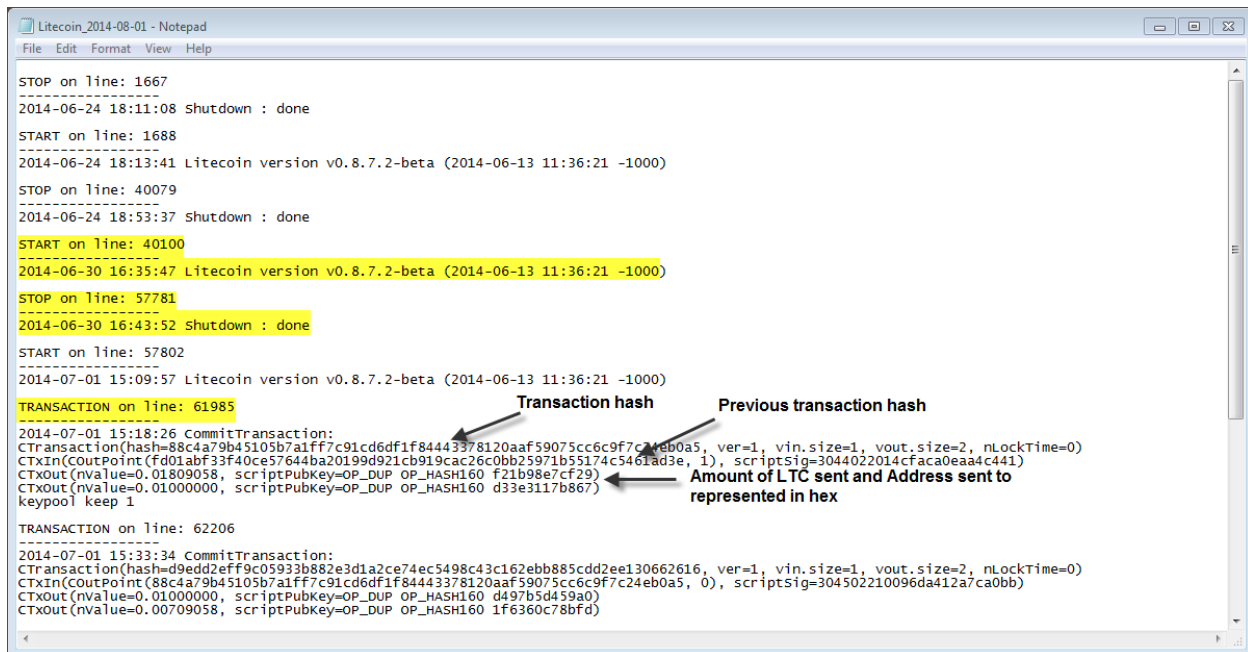


Figure 19. The Litecoin Ink files contain the SID belong to the user account which the application was installed on.

Litecoin has a debug.log file, in the directory previously mentioned, that functions in a similar manner to the multibit.log file. It contains information relating to user usage of the application and transaction data (Fig. 20). As it is a log file and can be overwritten with heavy usage, only recent usage and transaction data will be contained. All timestamps appear to be in UTC. Transactions sent from the

wallet begin with “CommitTransaction”. There are received transaction hashes present in debug.log, however, they do not contain full transaction information like multibit.log.



```

Litecoin_2014-08-01 - Notepad
File Edit Format View Help

STOP on line: 1667
-----
2014-06-24 18:11:08 Shutdown : done

START on line: 1688
-----
2014-06-24 18:13:41 Litecoin version v0.8.7.2-beta (2014-06-13 11:36:21 -1000)

STOP on line: 40079
-----
2014-06-24 18:53:37 Shutdown : done

START on line: 40100
-----
2014-06-30 16:35:47 Litecoin version v0.8.7.2-beta (2014-06-13 11:36:21 -1000)

STOP on line: 57781
-----
2014-06-30 16:43:52 Shutdown : done

START on line: 57802
-----
2014-07-01 15:09:57 Litecoin version v0.8.7.2-beta (2014-06-13 11:36:21 -1000)

TRANSACTION on line: 61985
-----
2014-07-01 15:18:26 CommitTransaction:
CTransaction(hash=88c4a79b45105b7a1ff7c91cd6df1f84443378120aaf59075cc6c9f7c24eb0a5, ver=1, vin.size=1, vout.size=2, nLockTime=0)
CTXIn(CoutPoint(f01abf33f40ce57644ba20199d921cb919cac26c0bb25971b55174c5461ad3e, 1), scriptSig=3044022014cfaca0eaa4c441)
CTXout(nvalue=0.01809058, scriptPubKey=OP_DUP OP_HASH160 f21b98e7cf29)
CTXout(nvalue=0.01000000, scriptPubKey=OP_DUP OP_HASH160 d33e3117b867)
keypool keep 1

TRANSACTION on line: 62206
-----
2014-07-01 15:33:34 CommitTransaction:
CTransaction(hash=d9ed2eff9c05933b882e3d1a2ce74ec5498c43c162ebb885cdd2ee130662616, ver=1, vin.size=1, vout.size=2, nLockTime=0)
CTXIn(CoutPoint(88c4a79b45105b7a1ff7c91cd6df1f84443378120aaf59075cc6c9f7c24eb0a5, 0), scriptSig=304502210096da412a7ca0bb)
CTXout(nvalue=0.00709058, scriptPubKey=OP_DUP OP_HASH160 1f6360c78bfd)

```

Figure 20. Parsing the debug.log file can be useful to an examiner attempting to determine recent user activity and transactions.

Litecoin registry keys only had values relating to system information. Litecoin keys were found in SYSTEM and SOFTWARE hives. The following keys were associated with Litecoin installation:

- HKLM\SYSTEM\Classes\litecoin
- HKLM\SYSTEM\Wow6432Node\Litecoin
- HKU\[User SID]\SOFTWARE\Microsoft\Windows\Uninstall
- HKU\[USER SID]\SOFTWARE\Litecoin

When Litecoin was uninstalled, all the previously mentioned artifacts were still present with the exception of the registry keys and lnk files. Unlike MultiBit, uninstalling Litecoin did not lead to the creation of an object that could date that activity.

String analysis of the VM VMEM files led to similar results seen in MultiBit as there were multiple references to addresses that were sent to during that user session (Fig. 21).

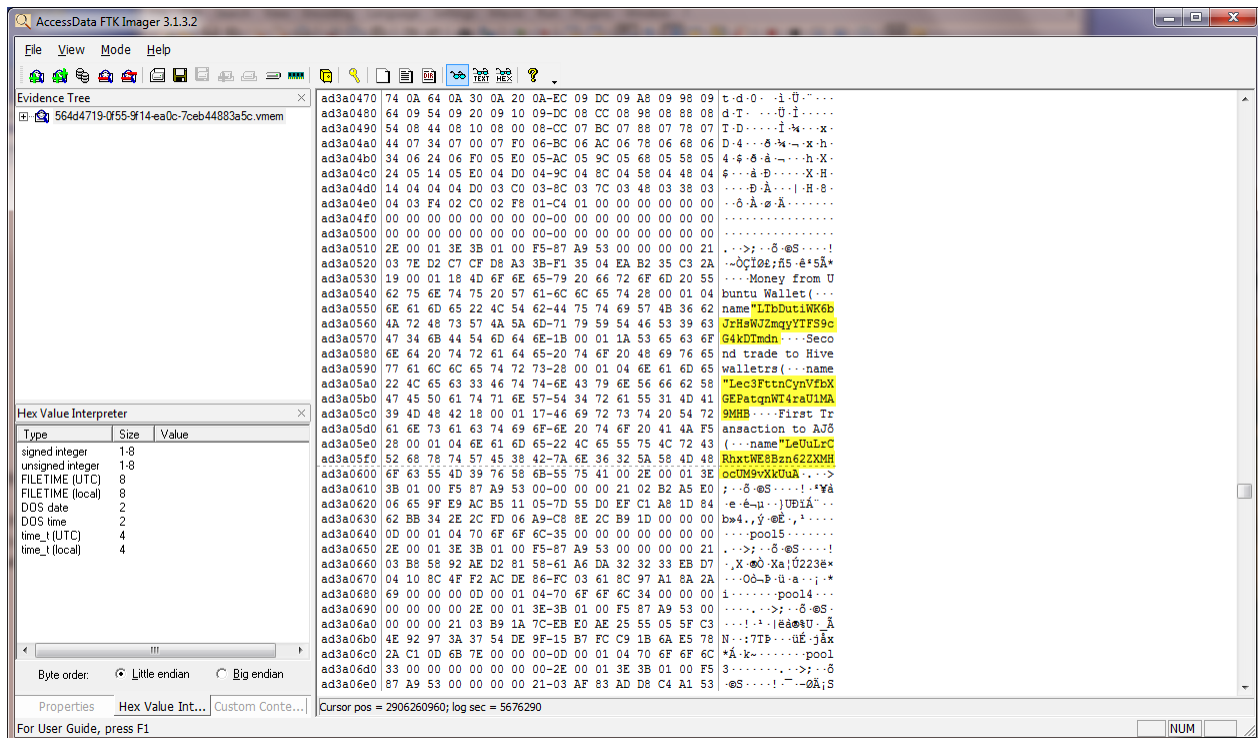


Figure 21. String searching in memory is another source to observe recent transactions between the last shutdown of the machine.

Litecoin has similar network traffic as Bitcoin. Unfortunately, there is currently no built in protocol as of Wireshark (Version 1.10.0). Wireshark categorizes Litecoin traffic under the TCP protocol. Litecoin traffic operates on port 9333 and contains many of the same messages as Bitcoin including: inv, getdata, and tx. As there is no built in protocol an examiner must manually parse the litecoin traffic (Fig. 22). The inv and getdata messages are relatively straightforward to parse, normal packets will be 115-bytes in length and contain the hash of either a transaction or block in big endian.

inv		getdata		tx	
Byte offset	Item	Byte offset	Item	Byte offset	Item
0-53	Packet Structure	0-53	Packet Structure	0-53	Packet Structure
54-	Data	54-	Data	54-	Data
54-57	Packet Magic	54-57	Packet Magic	54-57	Packet Magic
58-69	Command Name (69:6e:76)	58-69	Command Name (67:65:74:64:61:74:61)	58-69	Command Name (74:78)
70-73	Payload Length	70-73	Payload Length	70-73	Payload Length
74-77	Payload checksum	74-77	Payload checksum	74-77	Payload checksum
78	Count	78	Count	78-81	Transaction Version
79-82	Type (01 - TX, 02 - Block, 03 - Unknown)	79-82	Type (01 - TX, 02 - Block, 03 - Unknown)	82	Input Count
83-114	Data hash	83-114	Data hash	83-262*	Transaction Input *Variable length - 180 bytes in this example
				263	Output Count
				264-297	Transaction Output *Amount of tx output fields dependent on output count
				297-301	Block lock time or Block ID

Figure 22. The packet details of the three main messages in the Bitcoin protocol are detailed. The fields within Transaction Input and Output are not shown in this figure. Note that these packet details have the same structure as Litecoin network traffic or any other cryptocurrency using the Bitcoin protocol.

The tx message is more difficult to parse due to variable fields (Fig. 23). The Transaction Input field contains 36-byte Previous output (32-byte transaction hash and 4-byte index), 1-byte Script Length, variable length Signature Script, and 4-byte Sequence fields. Typically, there is only one Transaction Input field, however it is more common to see multiple Transaction Output fields. The number of Transaction Output fields will depend on the value of Output Count. Transaction Output contains an 8-byte Value (represents the 1×10^8 amount sent), 1-byte Script Length, 25-byte Script fields.

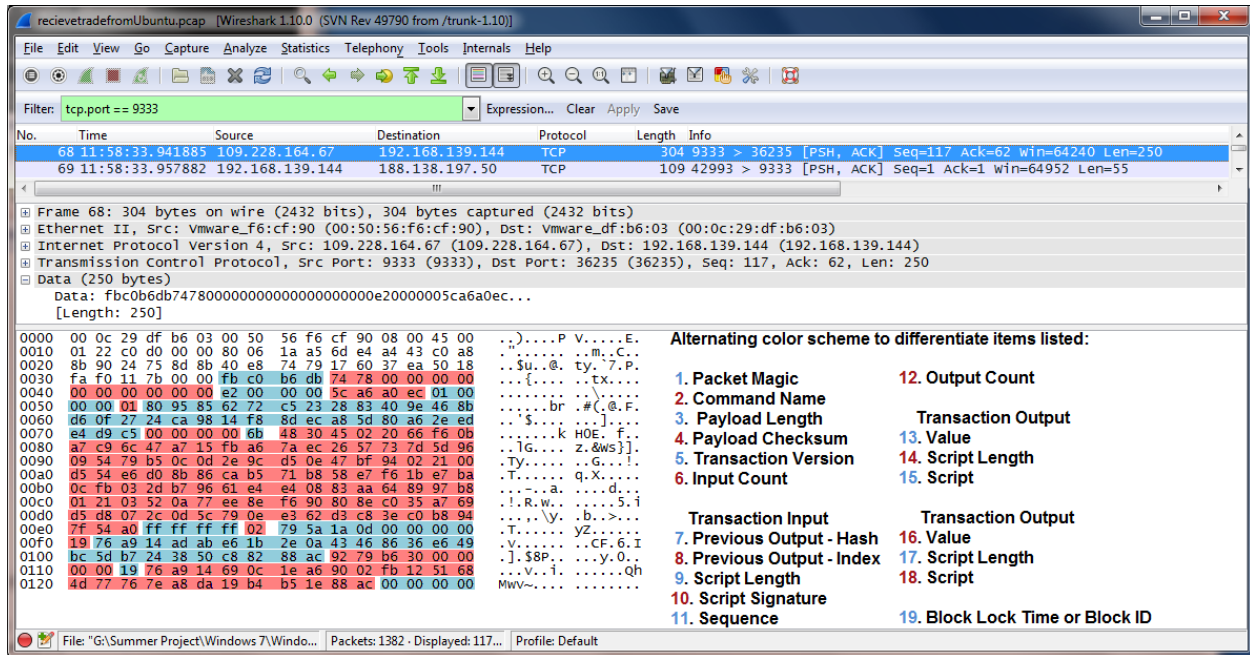


Figure 23. The breakdown of the contents of a Litecoin tx message. This process can be repeated for any cryptocurrency using the Bitcoin protocol. Please note that the length of the “Script Signature” (10) is variable and the “Output Count” (12) will determine how many “Transaction Output” fields there are. Additionally, the “Transaction Output Script” value must be processed to obtain the address in a similar fashion described in the Bitcoin section (note the Network ID for Litecoin is typically “L”).

Installation of the Litecoin wallet leaves behind many pertinent artifacts on the user’s computer. An examiner can determine which account the program is installed on, recent transactions, and general usage of the application. Even after uninstalling the application, these artifacts are still observable. Unlike MultiBit, there was no unique file created when uninstalling the application.

Darkcoin - Darkcoin 9.11.6

Darkcoin shares many similarities to Litecoin. Perhaps because the two wallets examined both utilize the Qt application framework. Installation did not create lnk files to analyze. The wallet.dat and debug.log files have similar artifacts and can be found in C:\Users\[User]\AppData\Roaming\Darkcoin and /home/[User]\.darkcoin directories on Windows and Ubuntu, respectively. The wallet.dat file contains references to addresses belong to the wallet and those that were traded to (Fig. 24).

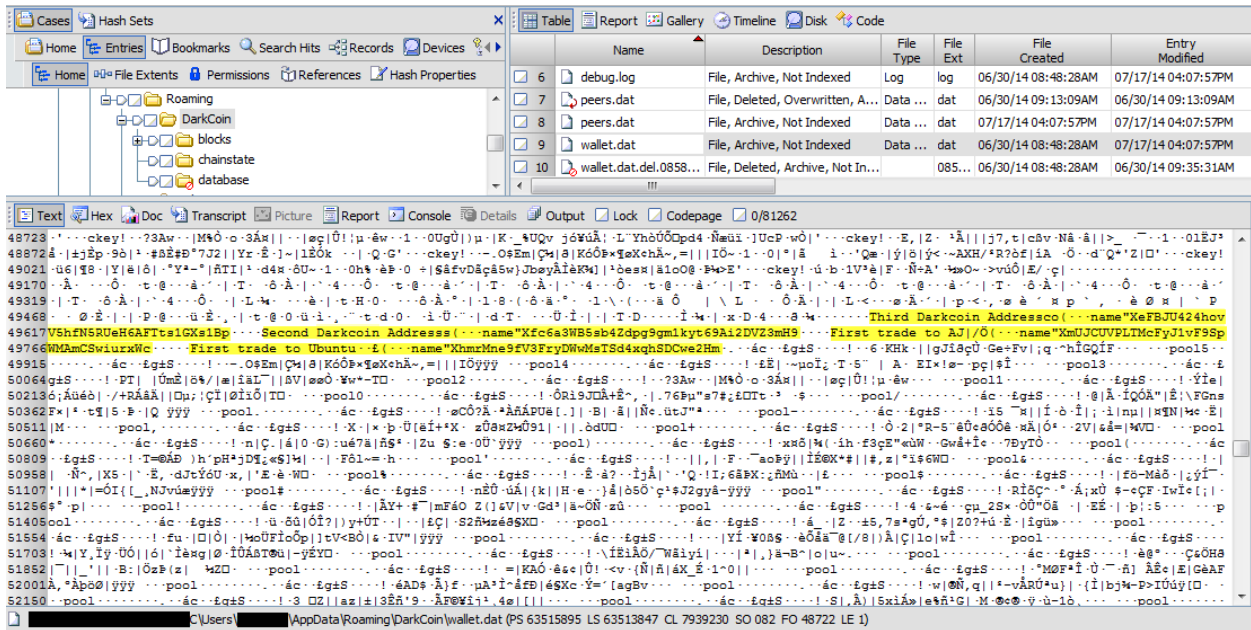


Figure 24. The wallet.dat file contains references to both addresses belonging to the wallet and addresses that were sent to in transactions.

The debug.log file has the same format and type of information as the Litecoin variant.

CommitTransaction indicates transactions sent from the Wallet. Application start and stop times are noted by "Darkcoin version" and "Shutdown : done" lines. (Fig. 25).

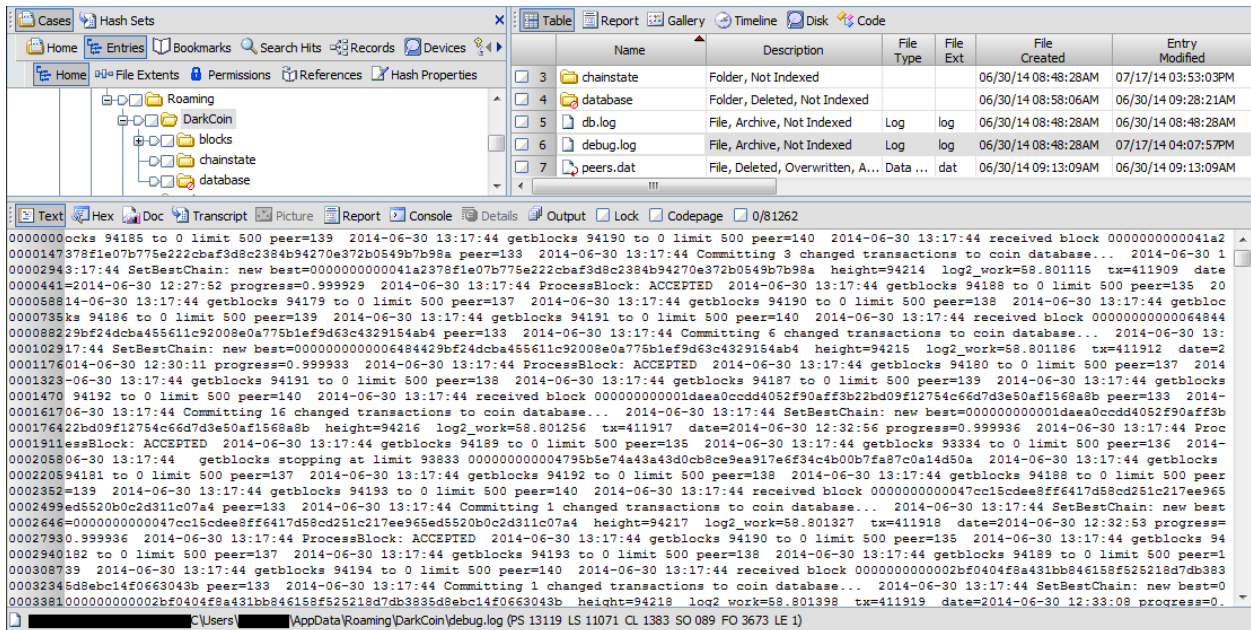


Figure 25. The debug.log file contains the same information found in the Litecoin debug log. This includes start and stop application time as well as transactions sent from the wallet. Timestamps are in UTC.

The HKU\[User SID]\Software\DarkCoin registry key appeared to be the only notable change to the registry during installation.

Oddly enough, the current version of Darkcoin does not contain an uninstaller and was not picked up on as an installed program in the control panel. For this experiment, Darkcoin was not uninstalled to determine what type of data gets left behind.

String searching memory with known user Darkcoin addresses contained references to addresses involved in transactions during the current user session (Fig. 26).

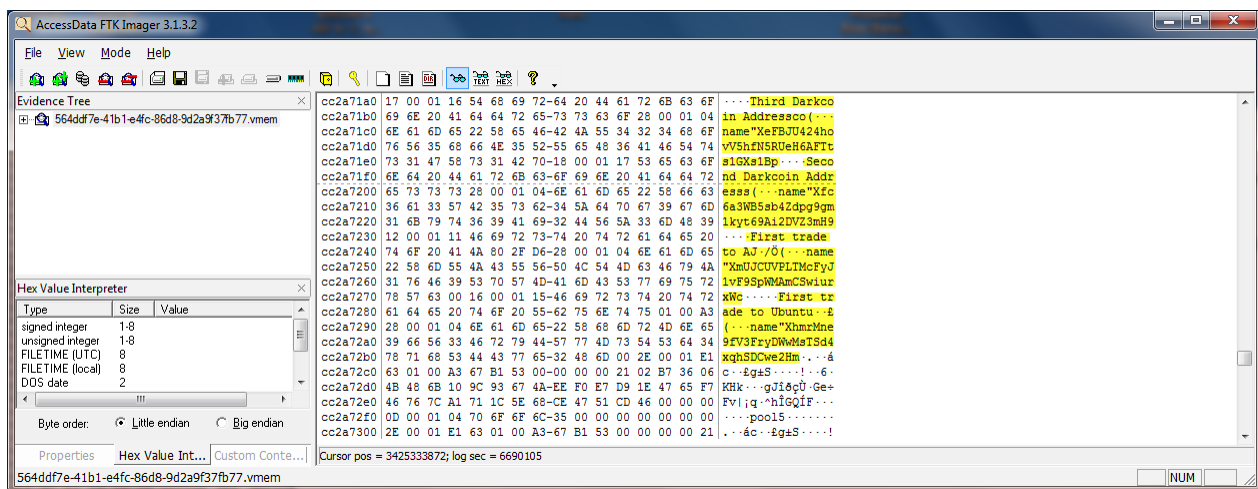


Figure 26. String searching addresses in memory contains the same type of data as stored in the wallet.dat file. However, this can also be used to determine some transactions that have occurred in this user session.

Darkcoin operates on port 9999 and like Litecoin runs off of the Bitcoin protocol. However, it does appear to contain some unique message types. Darkcoin's network traffic retains the typical inv, getdata, and tx structure indicating a transaction and can be manually analyzed in the same method described before (Fig. 27). The dseep message appears to be a unique message type to Darkcoin. However, according to a developer post, this message pertains to masternode ping messages and is not related to a particular transaction¹⁷. The network ID for Darkcoin is commonly an 'X'.

¹⁷ Duffield, Evan. [New version 9.4.0 and 10.8.0](https://darkcointalk.org/threads/new-version-9-4-0-and-10-8-0.690/). 5 May 2014. <<https://darkcointalk.org/threads/new-version-9-4-0-and-10-8-0.690/>>.

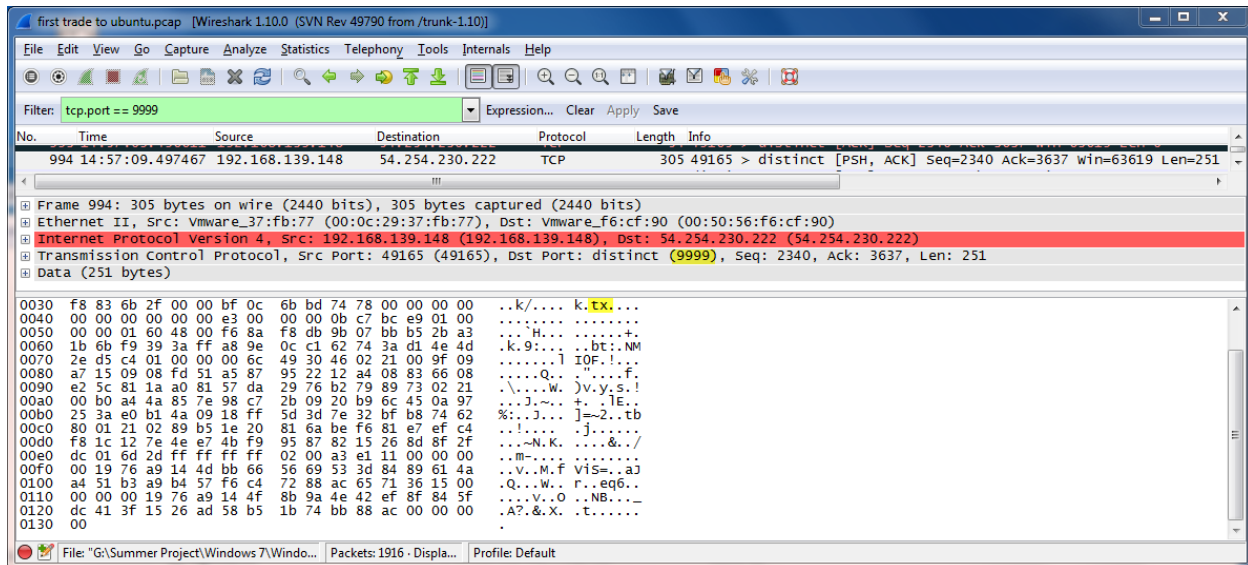


Figure 27. Darkcoin contains most of the same message types as Bitcoin and Litecoin. It can also be manually analyzed in the same fashion. Wireshark does not recognize Darkcoin and interprets it as TCP traffic running on port 9999.

Darkcoin's claim to fame is the anonymous Darksend feature which allows transactions to be 'completely anonymous'. However, as of Release Candidate 3, there are currently many restrictions on this functionality. Because of this, there are not enough users participating to actually mix the transactions correctly. So while this data does not include transactions using Darksend, it can give an examiner an idea of where to start looking for transactions.

Conclusion

Examiners must be aware of common artifacts and the extent of information that is obtainable from evidence. In this study, the best method for observing transactions were network captures in combination with a blockchain lookup utility. However, it is important to find remnants of those transactions on a user's computer. When examining a wallet application, it is vital to determine if there is a corresponding log file because they contain the most user activity and transaction information, outside of a network capture. The cryptocurrency wallet software examined shared remarkable similarity in the amount of information stored on the user's machine for both Windows 7 and Ubuntu 14.04. Network traffic was also similar across the cryptocurrencies studied. The process of parsing Bitcoin's network traffic, such as a tx message, was identical in Litecoin and Darkcoin. Consumers and

investors have demonstrated high interest in Bitcoin and virtual currencies as a whole and examiners should be prepared to analyze cryptocurrencies effectively as the likelihood of cases involving them continues to increase.