[CONFIDENTIAL]

Bitcoin 6.0: Military Grade e-Payment System

Dr. Yuvraj Kumar, Satoshi Nakamoto
dryk7007@gmail.com , satoshin@gmx.com
www.dryuvrajkumar.com, www.bitcoin.org

Released on: "Wednesday, June 24, 2015 ~ 11:52 AM GMT"

[*"Reworked, Enhanced, and revised version of the original Bitcoin Core Proof-of-work*"]

I. **Abstract**:

A purely anti-node non-peer-to-peer version of electronic money would allow online payments to be sent directly from one person to another without going through a financial institution and node promoters such as digital and gold-miners. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using an anti-node non-peer-to-peer network which is not only distributed but encrypted to the core of the gene. The network timestamps transactions by hashing them into an ongoing chain of hash-based genesis-concept combined with military grade chain-reinforced-encryption using AES, SHA, RSA and custom algos, forming a record that cannot be changed without redoing the genesis-concept. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcasted on a best effort basis, and nodes can leave and re-join the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone. However, in the proposed model, the central authority server itself acts as the node and represents the network transport layer by itself which enables it to not to be dependent on individual nodes, and the crypto-formulation continuous at server level. By this process, the dependency on nodes are ruled out and the server (the executioner, in this case) gets all the bits combined in one platform from the beginning to the end using enhanced salting layer, providing the facility of de-centralized defacto standard e-payments within nano seconds or less i.e., account to account transfer using one central block processing schema.

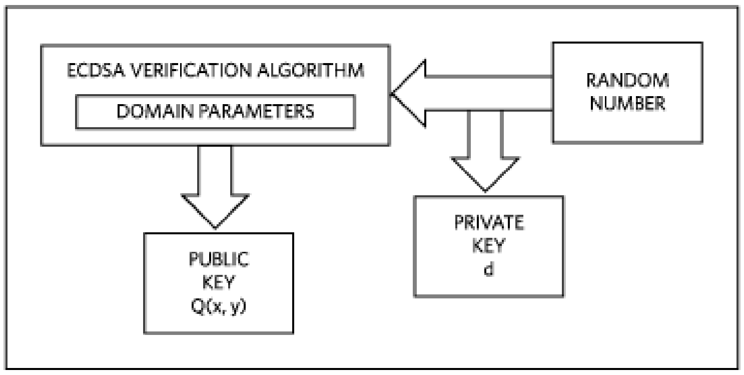
II. **Introduction:**

e-Business in the online world has come to rely almost exclusively on banks both regulated and unregulated serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust-based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non- reversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

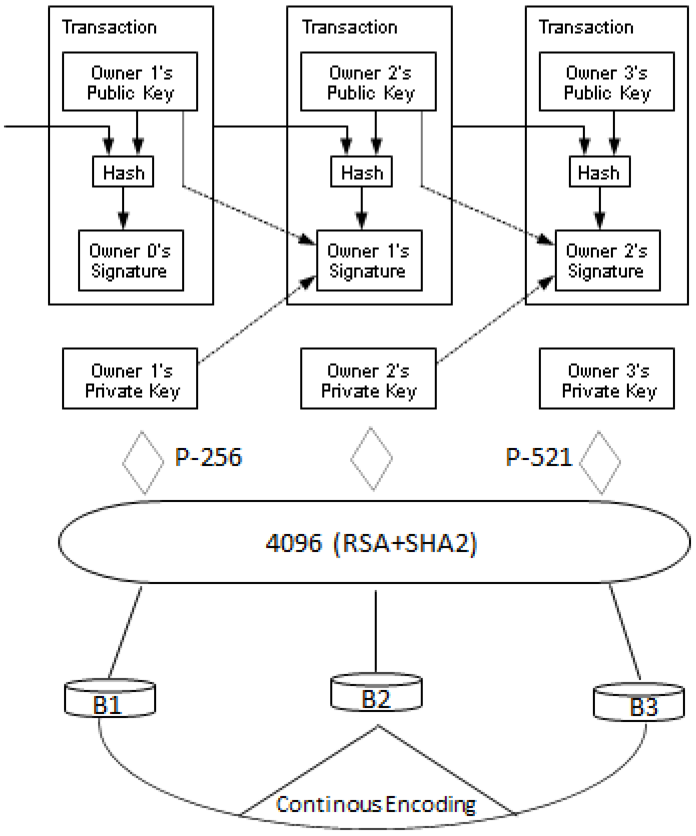
What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and multi-encrypted transaction using same base node on the originating server towards to the end node of the account holder, which means that the transactions executes and ends at the server itself while ensuring data is converted into blocks, encrypted, decrypted and processed in real time. This saves the time and effort taken for mining process, node dependability and block aggregation which acts as a contributor to ledger being delayed, blockcypher modification and non-intended route morphing. In this paper, we propose an enhanced solution to the existing double-spending problem using a peer-to-peer, semi-node distributed real-time military grade block encryption and decryption mechanism using 4096 bit RSA + SHA2 crypto-dynamics conjoined with the AES-256 encryption for blockcypher redundancy assurance clubbed with timestamp server to generate computational proof of the chronological order of transactions. The system is secure as long as the server node and the transaction initiator along with the recipient collectively control the entire transaction under the same server umbrella which eliminates the propagation of participation of any third-party cooperating group of attacker nodes.

III. Payments:

Satoshi, in his original proof-of-work defined an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership. Now, with the introduction of new and latest reverse engineering systems and tools, the possibility of hash-reversing and altering the private and public keys created by the Elliptic Curve Digital Signature Algorithm (ECDSA) and others, the blockchain is no more the secure block-chain as it was intended to be for the utilization by the block-chain systems. Blockchain is broken, it is being broken and the in-route alteration is now a reality, hence other than the private-public key transactional sequence, the multi-layer gyro-curve of the overall transaction has to be implemented starting from the group 1 of the private key (d) and ending up in the group 2 of the public key (Q). The below mentioned image shows the existing block verification system:



In this entire duration, the payee can't verify that one of the owners did not double-spend the coin and that the data is altered or routed making the coin disappear due to the possibility of node entry / exit. A common solution is to introduce a trusted central server authority, or gyro-gyro-mint, that checks every transaction for de-encryption, gyro-volatility and double spending along-with the block cypher's authenticity. After each transaction, the coin must be returned to the gyro-gyro-mint to issue a new coin, and only coins issued directly from the gyro-gyro-mint are trusted not to be double-spent without losing the validation test. The problem with this solution is that the fate of the entire money system depends on the company running the gyro-gyro-mint, with every transaction having to go through them, just like a bank, in these terms of banking used for the block submission and retrieval, we talk about the de-centralized approach of public banking. The below provided image shows the revised, reworked and revised version done by Yuvraj Kumar; of the existing proof-of-work of Satoshi:



Satoshi, as mentioned in his previous work; needed a way for the payee to know that the previous owners did not sign any earlier transactions. For our purposes, the earliest transaction is the one that counts, so we don't care about later attempts to double-spend. The only way to confirm the absence of a transaction is to be aware of all "encrypted" transactions. In the gyro-mint based model, the gyro-mint was aware of all encrypted transactions and decided which arrived first with the block-queue processing activated. To accomplish this without a trusted party, transactions must be publicly announced [1], and we need a system for participants to agree on a single history of the order in which they were received. The payee needs proof that at the time of each transaction, the majority of nodes agreed it was the first received. However, irrespective of the fact that which block arrived first, the data with its timestamp modified and manipulated in-order to route the block to the un-intended node, the purpose fails. Hence the end-to-end encryption on the central authority which ensures node’s authenticity while ensuring the data integrity is of paramount importance.

IV. End to End Loop Gyro-Encryption:

The solution we propose begins with a timestamp server as the initiator but doesn’t end with the timestamp alone. A timestamp server works by taking a hash of a block of items to be timestamped and widely publishing the hash, such as in a newspaper or Usenet post [2-5]. The timestamp proves that the data must have existed at the time, obviously, in order to get into the hash. Each timestamp includes the previous timestamp in its hash, forming a chain, with each additional timestamp reinforcing the ones before it. The hash generated has to pass to through a string of algorithms as a standard procedure for getting ready for the transport. The block package is enclosed using combination of “RSA+AES256+SHA2+SHA256+ECDSA+ppkgvsea9917” algorithm encryption. The ppkgvsea9917 algorithm is by far the most trusted multi-layer encryption algo existing hidden inside the root of the world’s most power symmetric computing system buried in the great alps. The ppkgvsea9917[9] and ppkdvsea9918[9], gv# and dv## represents gyro-volatility and dynamic-volatility respectively [9].

V. Concept:

To implement a distributed block mechanism under the central authority, all the blocks before initiation should be encrypted with the gv or dv algo framework and the request should be handed over to the timestamp server on a peer-to-peer basis, enstaging the proof- of-work system similar to Adam Back's Hashcash [6], rather than newspaper or Usenet posts. The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

For our military grade encrypted blockcypher timestamp network, we implement the proof-of-work by incrementing a nonce in the block until a value is found that gives the block's hash the required zero bits. The usage of CPU effort expansion is not required to satisfy the proof-of-work, re-doing of the block changing without redoing the work is not required. As later blocks are chained after it, the work to change the block would include redoing all the blocks after it where the initial block started.

The concept also solves the problem of determining representation not only in majority decision making but also the fear of rejection of blocks during and post transaction of a block, but also the questions raised on the data integrity of the cypher in transit. If the majority were based on one-IP-address-one-vote, it could be subverted by anyone able to allocate many IPs. This enhanced Proof-of-work is essentially one-CPU-one-authority-multi-vote. The majority decision is represented by the shortest chain, which has the greatest proof-of-work effort invested in it with the chain of standards embedded into each other, proves authenticity of the intended transaction. In the previous work of Satoshi, he emphasised that If a majority of CPU power is controlled by honest nodes, the honest chain will grow the fastest and outpace any competing chains, and that’s a problem. Dependency on the CPU power leads to no guarantee that the nodes are the real contributors to the overall transaction transport chain, rather it’s the application of the right standards combined with the right face value. To modify a past block, an attacker would have to redo the proof-of-work of the block and all blocks after it and then catch up with and surpass the work of the honest nodes. To reinstate the block defacto real time-encryption-decryption standard, the central authority has been provided the right to determine whether the initiator and the receiver are on the same loop trunk, this leaves the attack vector required for the attacker to hush into the honest nodes, eliminating the need of the node requirement, in short, the miners role is not required at all. Attackers catching up diminishes exponentially as subsequent blocks are added and decrypted under the central "root" authority.

While the application of the central authority is required in order the transaction to be fully complete, the parties involved have to be on the same transport layer package which ends up getting complete at the server side encrypted-block bucket. This eradicates the need for increasing hardware speed and varying interest in running nodes over time, the proof-of-work difficulty is no more required to be determined by a moving average targeting an average number of blocks per hour. This also stops the generation of unnecessary blocks by the attackers if at all they are trying to generate, they can't, since package of a block is a combination of packages, trying to decrypt one layer of the package introduces more layers of package which keeps on increasing till the drill down is down, to the power of *n*.

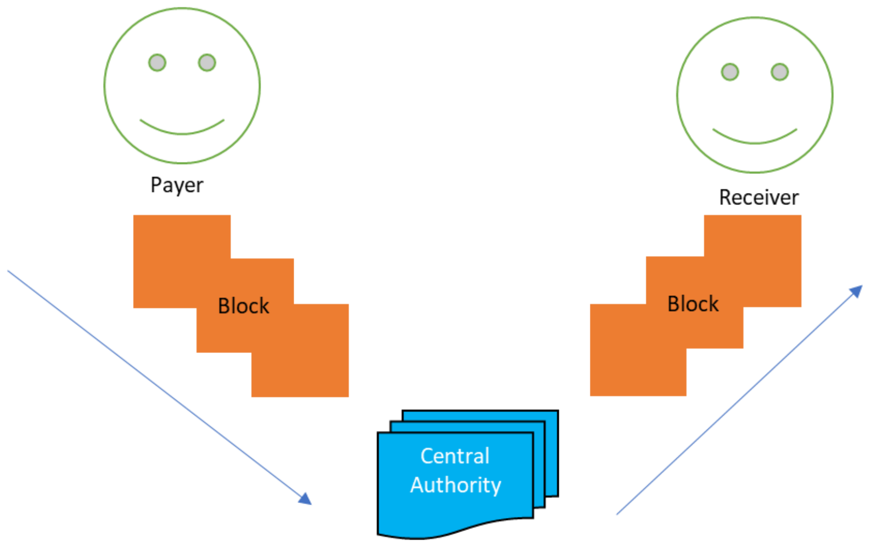
VI. Transport and Processing:

Transport and processing process are as follows:

1. On initiation, block is identified and packaged
2. Set of blocks are packaged into the package of gyro-encryption pool
3. Transaction is broadcasted to the central authority in real time
4. Both the parties involved (payer + receiver) are connected the central authority
5. While the connections are alive, the central authority runs the gyro-pool for decryption of the blocks using the sea$^{\$}$ (secure encryption algorithm)
6. While the decryption is happening, the information is floated to both the parties in real time on the status of the transaction with an approximate time left (similar to the ETA) indicator
7. The authority issues confirmation on the decryption and issues the transaction to the receiving party using same encryption model again.

There is no need of the application of nodes as they always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one, and this is a problem which was not identified earlier while the previous paper was written by Satoshi on the proof-of-work.

As per Satoshi's PoW, the new transaction broadcasts do not necessarily need to reach all nodes. As long as they reach many nodes, they will get into a block before long. Block broadcasts are also tolerant of dropped messages. If a node does not receive a block, it will request it when it receives the next block and realizes it missed one. The new proposed model ensures that no third-party player participates in the transport process and that the transaction stays between two entities governed by the central authority and not on the miners or crypto-cpu-nodes.



VII. Reward:

By convention, the first transaction in a block is a special transaction that starts a new coin owned by the creator of the block. This adds an incentive for nodes to support the network, and provides a way to initially distribute coins into circulation, since there is no central authority to issue them. The steady addition of a constant of amount of new coins is analogous to gold miners expending resources to add gold to circulation. In Satoshi’s case, it is CPU time and electricity that is expended. But in the new proposed PoW, the factor is the inclusion of central authority which guarantees the involvement and granting of the transaction in the form of a completed transaction. The circulation of coins is not governed by the central authority, but by the two parties involved in the block exchange trigger.

The funding of incentive is not required to be provided since the inclusion of the gold-miners is ruled out as compared to the earlier PoW submitted by Satoshi. If the output value of a transaction is less than its input value, the difference was the transaction fee that is added to the incentive value of the block containing the transaction in the earlier work, now this is not required in the new model. Once a predetermined number of coins have entered circulation as guaranteed by the central authority, the incentive need not transition entirely to transaction fees and be completely inflation and third-party free and only the parties involved in the transaction benefit from the transaction, not the miners.

The act of encouraging nodes to push and stay honest is also ruled-out. Also the possibility If a greedy attacker is able to assemble more CPU power than all the honest nodes, and he choosing between using it to defraud people by stealing back his payments, or using it to generate new coins is also ruled out since the role of gold-miners is eliminated which means that the blocks don’t travel anymore between the distributed node network.

VIII. Storage Revival:

The requirement of a storage space has been ruled out as the transaction is fully autonomous and doesn’t leave any trail of the transport due to the enhanced salting applied before the start of the packet journey. In the work submitted by Asiha Mangen [10] it has been learnt that the packets being transported in the network cannot be traced and that packet sniffers and tracers are not equipped to identify the log signature. While in transport, the transactions are hashed in a Merkle Tree [7][2][5], with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored and the processing is seamlessly processed by the authority.

A block header with no transactions would be about 80 bytes. If we suppose blocks are generated every 10 minutes, 80 bytes * 6 * 24 * 365 = 4.2MB per year. With computer systems typically selling with 4GB of RAM as of 2014, and Moore's Law predicting current growth of 3.6GB per year, storage should not be a problem even if the block headers must be kept in memory.

IX. Transaction verification:

It was possible to verify payments without running a full network node in the previous work of Satoshi. A user only needed to keep a copy of the block headers of the longest proof-of-work chain, which he can get by querying network nodes until he's convinced, he has the longest chain, and obtain the Merkle branch linking the transaction to the block it's timestamped in. He can't check the transaction for himself, but by linking it to a place in the chain, he can see that a network node has accepted it, and blocks added after it further confirm the network has accepted it, this was a problem too. Since the blocks have to travel and expose themselves with the elliptic curves being active, the capture, processing and release of the data to nodes make the transaction unreliable, due to the fact that the miner is always a third party and will stay as a service provider, which is not required for a transaction to take place between two parties.

The verification is securely done through the private tunnel activated for a transaction and that the intrusion is not possible, since the application of packaging infrastructure in place, the possibility of packet trace and block manipulation is next to impossible if the gv and dv algos are rightly implemented. In the proposed model, the e-payment network uses its own military grade network away from prying eyes and ensures that both the suppliers and the buyers are safe and confident about their transaction which they performed.

As such, the verification was reliable as long as honest nodes control the network, but is more vulnerable if the network is overpowered by an attacker, hence the need of eliminating the miner to include and power the node network has been eliminated. While network nodes can verify transactions for themselves, the simplified method can be fooled by an attacker's fabricated transactions for as long as the attacker can continue to overpower the network. One strategy to protect against this would be to accept alerts from network nodes when they detect an invalid block, prompting the user's software to download the full block and alerted transactions to confirm the inconsistency and this proves impossible when network bandwidth and storage is concerned since GBs of data is required to be transmitted and downloaded frequently and that’s a huge burden on the operator. Businesses that receive frequent payments will probably still want to run their own nodes for more independent security and quicker verification. Businesses that receive frequent payments are also not required to run their own nodes for more independent security and quicker verification since the trunk-based transaction is in place already which means asset valuation and protection both are ensured without any chances of leakage during pre and post transaction sequence.

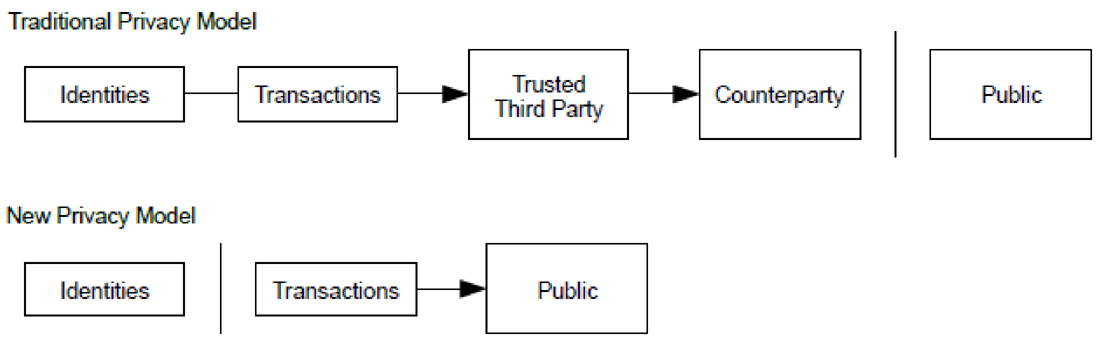
X. Encryption and Decryption for Pay-out:

Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous transaction or multiple inputs combining smaller amounts, and at most two outputs: one for the payment, and one returning the change, if any, back to the sender.

It should be noted that a transaction would not be dependent on several transactions, and those transactions further depending on many more, would be a problem here. There is never the need to extract a complete standalone copy of a transaction's history, but to safeguard the interests of the parties involved, it is imperative that the tunnel truncation is done between two parties while maintaining relationship with the initiator and the authority.

XI. Privacy:

As per Satoshi's previous PoW; it was said that the traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. The necessity to announce all transactions publicly precludes this method, but privacy can still be maintained by breaking the flow of information in another place: by keeping public keys anonymous. This is similar to the level of information released by stock exchanges, where the time and size of individual trades, the "tape", is made public, but without telling who the parties were. The public can see that someone is sending an amount to someone else, but without information linking the transaction to anyone, this becomes a challenge when the money from the account is stolen. The user losing money is of more grave concern than the concern of the privacy itself and safeguarding the user at all times with his money is the key factor for the affective operation of the entire bitcoin 6.0 network.



As an additional firewall, a new key pair should be used for each transaction to keep them from being linked to a common owner. Some linking is still unavoidable with multi-input transactions, which necessarily reveal that their inputs were owned by the same owner. The risk is that if the owner of a key is revealed, linking could reveal other transactions that belonged to the same owner, but that risk has to be applied and allowed, since ledge information have to co-exist for the validation of the transaction at all times.

XII. Calculations

We consider the scenario of an attacker trying to generate an alternate chain faster than the honest chain and to stop this the role of a miner is eliminated. Even if this is accomplished, it does not throw the system open to arbitrary changes, such as creating value out of thin air or taking money that never belonged to the attacker. Nodes are not going to accept an invalid transaction as payment, and honest nodes will never accept a block containing them. An attacker can only try to change one of his own transactions to take back money he recently spent and to not to let the role of an attacker involve in this ongoing transaction, the two party's involvement should be intact during the entire course of transaction transport.

The race between the honest chain and an attacker chain can be characterized as a Binomial Random Walk. The success event is no more the honest chain being extended by one block, increasing its lead by +1, and the failure event is the attacker's chain being extended by one block, reducing the gap by -1, since the node doesn't exist.

Consideration for the recipient of a new transaction on how long he needs to wait before being sufficiently certain the sender can't change the transaction. We assume the sender is an attacker who wants to make the recipient believe he paid him for a while, then switch it to pay back to himself after some time has passed. The receiver will be alerted when that happens, but the sender hopes it will be too late.

The receiver generates a new key pair and gives the public key to the sender shortly before signing and is seamlessly packaged using the string combination with an additional layer of hash salting. This prevents the sender from preparing a chain of blocks ahead of time by working on it continuously until he is lucky enough to get far enough ahead, then executing the transaction at that moment. Once the transaction is sent, the dishonest sender starts working in secret on a parallel chain containing an alternate version of his transaction.

The recipient waits until the transaction has been added to a block in the central authority and z blocks have been linked after it. He doesn't know the exact amount of progress the attacker has made, but assuming the honest blocks took the average expected time per block, the attacker's potential progress will be a Poisson distribution.

XIII. Conclusion

We have proposed a military grade e-payment system for electronic transactions with modifications to the Satoshi's proof-of-work without relying on trust, nodes, miners and node traffic. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending hence the introduction of central authority is proposed. To solve this, we proposed a secure non-peer-to-peer tunnel network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if the nodes are not traced by the tracing mechanism. The bitcoin 6.0 network is anti-node and doesn't apply and supply miners the block information, rather it starts from one and ends in the central server for further processing. is robust in its unstructured simplicity. Nodes work all at once with little coordination. They do not need to be identified, since messages are not routed to any particular place and only need to be delivered on a starting to ending point basis. No need of Nodes is required and are not required to even leave and re-join the network at will, restricting them from accepting the proof-of-work chain as proof of what happened while they were gone which stops the attackers sniffing on the node channel. Miners are not required to vote with their CPU power, expressing their acceptance of valid blocks by working on extending them and rejecting invalid blocks by refusing to work on them. Any needed rules can be enforced with this closed-tunnel mechanism and incentivising is hence not required.

XIV. References

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In 20th Symposium on Information Theory in the Benelux, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In Journal of Cryptology, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In Sequences II: Methods in Communication, Security and Computer Science, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In Proceedings of the 4th ACM Conference on Computer and Communications Security, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957

[9] S. Ronnin, "World Power beneath the earth – security matters demystified," 1971

[10] Asiha Mangen, "Re-Encrypting the Encryption with Re-Encryption", 1994