White Paper Rough Draft
2017-12-6

(In the interests of a timely release and feedback from more people, this "green" paper is a rough draft of the technology for MediaCoin. In order to not violate patent applications in progress, some details and descriptions are omitted.)

Encrypted Content Delivery

The primary weakness in the Public-Private key infrastructure that permits "Man in the Middle" Attacks has fundamentally been client authentication. Weaknesses have been discovered in the implementations and negotiations with the Public Key Infrastructure, but the secure design required both a certified server and client. While zero knowledge Succinct Non-Interactive Argument of Knowledge (zkSNARK) can be used to verify certain kinds of computations, we believe the major advancement offered by the cryptocurrency wallet plugins is client authentication. For the first time in the history of the World Wide Web, users have begun to adopt public keys accessible by web sites. Moreover, these public keys are tied to addresses that can be verified and used as a means of value exchange, through cryptocurrency, blockchain and even more relevant, tokens within a blockchain.

Tokens are an ideal mixture of trust in a trust-less system and internal accounting specific to a particular data silo. The implementation of user wallets prompts for payment, verifies the transaction with the client, and can be submitted blockchain that will prevent double-spending. Within a data silo, micro-transactions can be performed and tracked with regular bulk submission upon externality referencing events. Properly triggered, the double spend is still prevented and transactional costs can be aggregated and minimized in the same manner as ACH within the banking networks.

Our MediaCoin Token will leverage several advances in infrastructure, user adoption of public key cryptography in wallets, a network token representing value, military grade encryption decryption in secure browser sandboxes, GPU acceleration of memory intensive encryption, protocol enhancements to data tunneling (TOR), peer discovery (DHT) and file delivery (IPFS). While true privacy is not the goal of our implementation, we will facilitate enhanced privacy by full packet encapsulation in client. Tunneling is a mechanism necessary to content delivery and network access, but our design implementation won't directly increase latency to enhance privacy.

Tokens on 2+ Networks

Cryptocurrency, browsers, operating systems and posix compliant text editors (vi), are attempting to support multiple platforms where feasible is an important usability concern. Our intention is to support Ethereum Classic (ETC) for its immutability and community, and Ethereum (ETH) for their early adoption, liquidity, and token market capitalization. Chinese regulations permitting, we are willing to support the NEO network in the future. The extra development required to support ETH is minor, while a significant rewrite would be required to support NEO. When other blockchains are interoperable, we may support them in various degrees if it provides a significant advantage over Exchange based 'exotic pairs.'

Valuation on separate currencies requires some additional infrastructure to be implemented for seamless interaction. Firstly, we will attempt to equalize the price between ETC and ETH markets, through sales and purchases. New sales will be expressed in USD, but utilizing the currency that offers more price stability and transaction speed. Price volatility is especially dangerous to business, since currency swaps are often handled to provide liquidity and cover purchase orders, not as speculative

investment. Regulations typically require 'proven reserves' as a percentage of capitalization to ensure liquidity. Our reserves will be managed to minimize price fluctuations within a weekly range, provided settlement occurs within established timeframes matching other industries. We condemn market manipulation akin to insider trading, and seek long term price stability instead of volatile speculation. Exchange listings will be selected where margin based trading can be eliminated, rendered unprofitable or minimized.

Peer Session Handlers

Phase 1 implementation by MediaCoin is a proof of concept system where a central server and central repository exists for delivering streaming content. Music and video files exist on the server in an encrypted format, with normal browser based URLs for accessing that content. Publishing is performed by submitting unencrypted content to the central servers. (Offsite backups and redundant storage will temporarily be performed through traditional contracts, potentially with CDN and reverse proxy.) The main distinction in Phase 2 application topology is a layer 4 session handler. This is similar to the "resume" and multiple connection "download accelerators" for FTP. The protocol itself will have these upgrades, with support offered by the client. A DHT node in torrent (and IPFS) offers similar functionality. Reverse compatibility will exist by creating a series of HTTP and web-socket based connection handlers, similar to masternodes in DASH or the central gateways in IPFS. The SessionNode handlers will be aware of the client and their connection. The migration to this system will performed by a series of new DNS records and will be performed transparently to the user.

Phase 1: Allows for a quick and easy http download, similar to the internet we know today
Phase 2: The exchange of data between members our peer network allows for a more decentralized and more efficient download speeds and storage
Phase 3: The session control/TOR, as well as other measures, allow for increased privacy for our downloads

Network Tunneling

A higher level protocol session handler will be possible in Phase 3, communicating with the SessionNodes with a series of preferences. Installation of client software will allow any member of the network to function as a basic SessionNode, with peer weighting assigning greater priority to nodes with better bandwidth and more session compatibility. Emscripten with websockets will allow basic web browsers to connect to a rich suite of protocols, handling configurable parameters based on client preferences. Emscripten will only be used to initiate new connections in javascript, and not relay any connections for other users or nodes. UDP and Triple UDP based connections will be permitted to the SessionNodes, which can be forwarded or encapsulated into TCP and possibly ICMP traffic. It is important to review 'firewalker and 'netcat' utilities for further analysis of data transfer on ICMP.

The underlying network topology will be masked, as user requests will no longer be associated with IP addresses directly. A higher order naming structure based on binary merkle trees, hashes for content, and search engine style events will be supported with backwards compatibility for traditional URL support. SessionNodes will be aware of multiple protocols, and will forward cryptographic credentials within the network. For example, it will be possible to run Tor client in front of MediaCoin, and establish a VPN connection inside the MediaCoin network, fully encapsulating the traffic three times with all of the security benefits from each architecture. The latency of such connections may suffer however MediaCoin tokens can be spent to prioritize traffic within our layer of the traffic.

Network Access

When nodes are able to opt-in to sharing their bandwidth, a network is created providing network access to other members. Content can be cached with delivery fees paid between nodes. Bandwidth reselling which can be purchased from ISP or cellular provider will be encouraged. Router software will be extended to include token based network access and revenue. The payment model for hotels, airports, hotspots and subscriptions for network access will adopt our pricing or be disrupted. The use of large antennas and other infrastructure will be rewarded by providing better access. We will encourage 4 separate pricing modes: upload, download, cache, tunnel and private. OpenWRT and Cisco iOS are intended platforms for initial deployment. Features such as UDP relay through TCP, alternate tunnels for DNS, extended peer discovery for extra protocols (IPFS, Torrent, eMule, Tor, etc) and more (patent pending undocumented) will be offered as additional paid features (price set by node, with common defaults).

As FTP download accelerators and BitTorrent demonstrate, having simultaneous connections increases speed. The ability to simultaneously download multiple chunks of data is clearly valuable, but often artificially limited. (Web browsers are limited to 2 simultaneous connections per server.) Multiple connections across different spectrum should be supported, with bandwidth pooling between bluetooth, wifi, cellular data and more. If 4 cellular data connections provide 3kB/sec each, our goal is to deliver as much of the aggregate 12kb/sec as possible with each contributing member compensated for data provided.

Mesh Network

While network access is often taken for granted, the Arab Spring required a mesh network to maintain communication. This exemplifies the power of Mesh Network. Anonymous users contributed hardware and technical explanations to create this network in Egypt. Ad-hoc peer relationships will create a similar network overlaid on top of the existing network access, similar to how blockchain creates a separate computing platform on top of existing network access. Deficiencies in the network will present economic opportunities for any node with data available in the geographic region. The access taken for granted in some industrial nations will be a valuable and new blessing in many areas of the world. The Green Paper provided will explain the overall value of the network and detailing some combinations of nodes (access), resources within the network (files), compensation (tokens), privacy, security, redundancy,

Quantum Risk

Nov 10, 2017 IBM Announced their 50 qubit (https://www.technologyreview.com/s/609451/ibm-raises-the-bar-with-a-50-qubit-quantum-computer/ quantum computer.) Sweden contributed $1 billion in funding towards their development of a 100 qubit system. Unlike D-Wave's 2000 qubit system (https://www.dwavesys.com/press-releases/d-wave announces d-wave-2000q-quantum-computer-and-first-system-order) the IBM quantum computer is capable of performing Shor's algorithm, which is capable of solving Discrete Logarithms. This is an NP-complete problem, solved in polynomial (P) time using Quantum Fourier Transforms (QFT). The main limitation of quantum computing is that problems must be solved "all at once" and can't be solved in pieces the way silicon transistors currently do large number math. The second limitation is decoherence, where the system falls apart before an answer has appeared. Thirdly, answers are probabilistic, with some noise and chance that a wrong or "no answer" will appear. The current simulator and cloud computing offered by IBM uses a mixture of

Python and their Qasm (Quantum assembler) language.
(https://quantumexperience.ng.bluemix.net/qx/experience)

The fundamental question on a system capable of solving Discrete logs and breaking All Modern Cryptography is mostly a question of "when" and not "if" it can happen. The underlying hardware used by IBM is nuclear magnetic resonance (NMR) machines, which are present in medical labs and university systems. Once the methods are figured, it is likely that many independent labs will be able to perform quantum calculations. If some of this research is able to leap forward based on the D-Wave designs (even though Niobium magnetic method and the NMR method are much different) then a quantum computer to break most cryptography is likely within 1 to 3 years. As larger and larger qubit arrangements become common, more and more cryptography will be solvable by such a system. If the techniques pioneered by DWave are not applicable, then a 3-6 year time frame is more realistic.

QFT is able to solve the private key if provided the public key in approximately the time it takes to load the question into the quantum computer. Estimates for time solving vary widely, from O(n), O(log n) to O(1) in the simple algorithm form. (https://en.wikipedia.org/wiki/Big_O_notation and examples http://bigocheatsheet.com/). Actual programming of a particular solution may change the number of qubits required for the same problem, which is more significant than the computing time required.

Hash functions are "unique identifiers" for information, like an abbreviation for a word. Unlike an abbreviation, they are not reserved and cannot be transformed back into the original data. Typical hash functions are MD5, SHA160, RIPE160, SHA2, SHA3. Symmetric key cryptography relies on a shared, secret key. Any knowledge of the secret key yields all data it protected. Typical symmetric "private" key encryption functions are 3DES, AES128 and AES256. Asymmetric "Public private" key systems use shared knowledge to hide a math problem, which contains a message or random data to be used as a secret key. Typical asymmetric systems are RSA, El Gamal, Diffie-Hellman (groups 1-5) and Elliptical Curves (DH group 14, 20, 21). Hash functions MD5 and SHA160 were weak enough to attack a decade ago. Specific data structures can be attacked now for RIPE160 and higher. (Mostly passwords 10 characters and under, based on today's GPU, with a known SALT.) Symmetric keys are "attackable" by guessing keys, but it highly impropable. All listed asymmetric keys are vulnerable to quantum computing with Shor's algorithm. Quantum safe public key system do exist, but they are not in use yet. (Lattices, McEliece, multivariate-quadratic systems, BLISS, etc.)

Astute readers may note that Hash and Symmetric key cryptography are not vulnerable to Shor's algorithm attacks. However, every implementation of hash and symmetric key cryptography uses asymmetric keys somewhere in the implementation as part of a complete system. Some implementations of software hashes, which are listed on a website, and Perfect Forward Security with partial messages are not vulnerable. (PFS is vulnerable if the beginning of the message is captured, but message fragments are not.)

Every Bitcoin and Ethereum transaction exposes "sec256k1" elliptical curve public keys. If an address has not performed a transaction previously, then the public keys in these systems are not exposed. Smart contracts in the Ethereum protocol do not have public or private keys, but the Owner of a contract will always have their keys exposed. "Dead" or "stuck" contracts cannot be recovered using quantum computing.

Quantum Migration

After quantum computing, the "best solution" offered among the Bitcoin community so far is creating

all transactions in pairs. The first transaction is a spend of bitcoin, and the second transaction moves to a new address to prevent a quantum computing attack against the public keys that were exposed. The "second best solution" is moving all bitcoin to a series of Central Authority Silos, which would store the balances in databases and have some sort of 2 or 3 factor authentication requirements, allowing registered users to safely store and spend their bitcoin. Neither practice would work with Ethereum, unless Smart Contracts were created without "owner" permissions and the authentication to the Smart Contracts do not involve modern public-private cryptography.

The most effective solution overall is an early migration. Quantum Safe Cryptography can be implemented now, and quantum safe blockchain can be developed within the next three to six months The author has spent some time on this, attempting to create a smart contract system with privacy and quantum safe libraries. In the budget projections, money is allocated to mathematicians and researchers in order to ensure accuracy, design attacks, and countermeasures. It is presumed since blockchain can be upgraded, new algorithms will fix any inadequacy revealed in the future. However, this proposition is not entirely valid unless the address and balance calculations are backward and forwards compatible.

In Bitcoin:

- sec256k1 "private key" which is a point on an elliptical curve
  https://en.bitcoin.it/wiki/Secp256k1
- sec256k1 "public key" which are 2 numbers, related to the elliptical curve
- a SHA256 hash of the public key
- a RIPEMD160 hash of the SHA256 hash

•

In Ethereum:

- sec256k1 "private key" which is a point on an elliptical curve
- sec256k1 "public key" which are 2 numbers, related to the elliptical curve
- a SHA3.keccak256 hash of the public key, truncated to 160 bits (non-reversible, even if the tech existed)
- Contract addresses are SHA3.keccak256 of RLP(nonce+sender_address), truncated to 160 bits. There is no public or private key for contracts

This data cannot be altered and still maintain the private key access to funds, which essentially locks the address scheme in place since the address is based on the public key, and the public key is revealed in transactions. In other words, after exposure the funds are accessible to anyone or to no one. Bitcoin could continue after upgrading to prevent quantum attack, but upgrades to the protocol will leave old accounts vulnerable until they perform transactions under the new architecture.

If we use a token instead of a central authority, we have access to cross chain transfers on exchanges as well as blockchain to blockchain. To facilitate a transition to a quantum safe blockchain, we are creating a custom code to be used where users can "burn" tokens and receive quantum safe coins on the new chain. The "burn" to transfer mechanism will be left unimplemented until the quantum safe mechanism is in place, tested and then the contract "library" will be upgraded with the function. In the meantime, contract to contract transfers are blocked by an implementation of Dexaran's ERC223 proposal (designed to prevent Ethereum loss). After migration upgrade, contract to contract will be permitted and tools will be made available making the process user friendly and will increase monetization.