HlyGrail or Holy Grail of Bitcoin

This document is an update to the original first coded filing of hlygrail0 that was filed under transaction: dc997ab339f36fbb5b15ea22bea2e58ef98d273bb0bfcf6dcbbbe270d1b267ef.

The hlygrail0.pdf said:
Use the following ADDRESS to find the transaction that contains OP_RETURN as output:
ADDRESS: 18hsnexG7KGUBtKgYnCS2Zyg59V5CsBPAt – Key for decoding hlygrail0.txt
OP_RETURN: SHA-256 hash verifies the encoded Intellectual Property file hlygrail0.txt

Address 18hsnexG7KGUBtKgYnCS2Zyg59V5CsBPAt and the OP_RETURN were outputs of the address 1ME8NEZEPhQdrCaLHojFnDDuiXU4joFTAE.

As stated in the original document any input or output in the transaction that has an OP_RETURN could be the decoding address.

So let me clarify. The transaction containing the OP_RETURN is the best for finding the intellectual property file and the application will need to specify what decoding address matches the hash.

The hash for verifying the encrypted file is always in the OP_RETURN of the transaction, at least as long as there is only one OP_RETURN in the transaction. This could change in the future if the blockchain allows more than one OP_RETURN in a transaction.

In many cases it may be better to have the output address or change address in the transaction be the encrypting address since it adds more proof that the encoder knew the encoding address prior to the transaction and checksum verification hash for the intellectual file in the OP_RETURN.

Sometimes it is difficult to determine which address will be the paying or input address for the transaction because HD wallets may select the address for you without you knowing ahead of time. Also a wallet may select many addresses for the input and usually choses the one or more with the least amount of funds in your wallet.

If you KNOW you are going to create the transaction and PAY with a known address it is better to pay with one address and use all the funds in the transaction so the transaction will have only one input. That input can now be the decoding address that will go with the hash or checksum stored in the OP_RETURN.

If on the other hand you KNOW, and usually you can predict, the address you are paying to, you can choose that address for the input address for the encoding key for your intellectual property file with the verifying checksum or hash in the OP_RETURN. It would still be possible for the address to receive other inputs at a later date so once again the address, either in or out, is not the best way to find the OP_RETURN. In this case the initial first transaction for the address is the key.

In summary, you should use the “transaction number” to specify the OP_RETURN containing the checksum and possibly pointer to multiple files and identify which address either input or output is the decoding key for your encrypted intellectual property file.

Examples and additional patent.

If anyone uses this process either for bitcoin or other blockchain technology for profit, they must have a license with the author of this document. Otherwise it is free for personal or nonprofit usage.

Suppose you have a website that contains the majority of the blocks of the Bitcoin blockchain.

The website has an index from 00000000 to FFFFFFFF which is over 2 Billion potential blocks.

Index 0000000 points to the bitcoin transaction for block0 in the blockchain.

This transaction can be looked up on the blockchain and get the OP_RETURN containing the checksum or hash for the encoded block0 which is encoded using the ONLY output ADDRESS for the transaction.

Block0 now does not need to be held in the full node blockchain computer. In fact, the encoded Block0 can be placed on ANY website or location or even MANY at the same time because all blocks are crypto logically identical.

Only one transaction needs to be held in the blockchain physically and the block can be stored outside the blockchain greatly freeing the amount of memory needed to be held by a full node.

As a side note: You would probably store most blocks, especially those that have had inactive spent transactions, outside of the full node computer and only need to keep the transactions to encrypted blocks, the most recent say 1000 blocks and unspent transactions in the full node.

Smart processing could move blocks in or out of the full node depending on how much access is needed. If you are only validating transactions as a node you keep most decrypted blocks in verified storage. The software could move blocks in and out of local memory depending on activity.

This process would allow storing thousands of times more data in the blockchain and cut transactions to either financial only or OP_RETURNS pointing to off chain data without blockchain bloat.

Now we need to get everyone to require a minimum input value, say $1 - $10, for transactions just to keep junk OP_RETURNS off the blockchain. The input value can go back to change but the miner would be willing to process the OP_RETURN because there is always a miner's fee.

Transactions to enable massive parallel programming

This is another feature of using the HlyGrail algorithm and that is to place computer programs off the block chain but encoded and run from the block chain. As example the OP_RETURN verifies an encoded intellectual property PROGRAM is valid and the address is the key to decrypt the code.

As example, Mary has a computer not doing anything so she loads a program that runs a function that returns a value after an hour of processing. She loads two programs on her computer.

1. Computer A is a program and goes to the blockchain to verify computer B program that is encrypted using the bitcoin address and verified by the OP_RETURN.
2. Computer A decrypts B and runs computer B program.
3. Computer B reads Computer A hash code and returns an encrypted code to A that allows A to

continue running the program if the hash matches what is stored in B. (The hash could be an algorithm that changes but only A and B would have all the local computer variables to make A process. In simple terms, B would just be the encrypted hash for A.)

4. A would bill the requester for running the algorithm and would process the algorithm with the variables send from the requester and return the function answer to the requester.
5. A would now wait for the next call and Mary's computer is making money instead of just heating the house.

This process would allow the same function to be running in parallel on MANY computers at one time with only slight modifications.

This is part of this patent / copyright. The author intends to use this process to create a program that calculates proof of work for a new Bitcoin or other altcoin block. The algorithm will be coded to use a number, or series of numbers to hash with the header sent by the requestor and return the result if a match is found. The verifier does not need to run mining software or have a full node.

Contact and further information.

Visit the website for more information or follow the address spends for this document and maybe you will find more hash codes to files of interest.

Look for my new article on how using OP_RETURNS to store everything will open the blockchain to enormous transactions which means bitcoin "circulates" which is what causes value for money.

Donations are appreciated to bitcoin: 1Gq9zUJVX8npd3WAoKSj7GbXbmCZUJMETx

This document is protected by HlyGrail 187ckcgWJWfa82mD4nrNLxUChtX5QJCg12

Owner: Roger Johnsrud, Genus Enterprises LLLP Date 1/30/2016

Website http://hlygrail.com

HlyGrail 486c79477261696c is the prefix for the OP_RETURN.
The suffix is the SHA-256 hash checksum for the intellectual property file hlygrail1.txt

Decode the intellectual property file hlygrail1.txt with rijndael-256 mode NCFB using the ADDRESS: 187ckcgWJWfa82mD4nrNLxUChtX5QJCg12

The decoded intellectual property file hlygrail1.txt contains the SHA-256 checksum of THIS plain text file. (hlygrail1.pdf)

HlyGrail Patent / Copyright Number 18hsnexG7KGUBtKgYnCS2Zyg59V5CsBPAt is the key for the transaction dc997ab339f36fbb5b15ea22bea2e58ef98d273bb0bfcf6dcbbbe270d1b267ef and the hash in the output OP_RETURN is the checksum for the hlygrail0.txt intellectual property file.