HlyGrail what is it Update for the User

HlyGrail IPF Patent/Copyright bc1qh9q5gasa7vzwymfustn4fjexw8xuw77gckwut4

This is a HlyGrail Patent / Copyright to update and explain the HlyGrail Algorithm in todays market and reinforce the original ownership of the HlyGrail Algorithm, copyright, and licensing requirements.

The owner of the HlyGrail Algorithm is Roger Johnsrud and can be reached via https://hlygrail.com.

The bitcoin ADDRESS of this document above was created as a new bitcoin ADDRESS before ever being on the blockchain. The SHA256 hash of this document and file size is included in an Intellectual Property File (IPF). The IPF document ALSO includes this ADDRESS and was then encrypted by AES-256 using the ADDRESS as the Key. A transaction was then created on the bitcoin blockchain with the above ADDRESS as the receiving coin for the transaction and includes the SHA256 hash of the ENCRYPTED IPF in the OP_RETURN with a prefix HlyGrail added in front of the Hash.

With one OP_RETURN the IPF can be proved to be the original by the Hash in the OP_RETURN and can be AES-256 decrypted by the ADDRESS used as the Key of the same transaction. Opening the decrypted IPF will reveal the SHA256 hash of THIS document which will prove the ownership of this document and that the document existed before the blockchain transaction.

This document can now be stored plain text as the HlyGrail IPF Patent / copyright provable by the hash of this file matching the hash in the IPF which can be verified from the blockchain.

History:

The project started in 2013 to encrypt data on the bitcoin blockchain without bloat. The HlyGrail Algorithm was conceptualized with initial coding attempts in 2015 but the tools for transactions like this one were limited and had to be developed. It was virtually impossible for others to see the OP_RETURN since most wallets did not include it, or the wallets did not even exist.

The first transaction was f670285c527a03f8a7bc5dd363ff9ba655fd6d7f7a4a38ea0eb551954a0a2831 which stated in the OP_RETURN “Get Ready for HlyGrail!” There are development transactions to it.

The first IPF was 54ab5bb230ebf9520b637d65a6e06f9b2f5702b8ebe36b44a67311e906beb2ec and had HlyGrail for the prefix but the decryption Rijndael-256 mode NCFB was difficult to get the hash.

The first HlyGrail0 that worked was ADDRESS 18hsnexG7KGUBtKgYnCS2Zyg59V5CsBPAt Transaction dc997ab339f36fbb5b15ea22bea2e58ef98d273bb0bfcf6dcbbbe270d1b267ef and introduced the HlyGrail Algorithm and licensing requirements.

The second HlyGrial1 was an update to clarify using HlyGrail as a pretend to the Hash and how files could be stored for retrieval. Also, it suggested using the Transaction to find the ADDRESS and OP_RETURN. The ADDRESS is 187ckcgWJWfa82mD4nrNLxUChtX5QJCg12 and the transaction is c1877e1f0aa7477ad6d5b1ca42620e062a3de3f518daeb31655817ab75b16882.

The bible was encrypted in ADDRESS 1LABBGXCk8MttukwuxBQEvua4pzKAsjA9S followed by the website for the HlyGrail 1NWWm8pCjJwqHcHwYLYJmRad6teia9r1Xd http://hlygrail.com in the OP_RETURN. Then the IPF created for the Bible 3G8vH5MwfqzuMrGC21qNX9cFUVEdqGoCKo.

Introduction

This algorithm was initially created to enable smart contracts with the bitcoin blockchain. The concept was to allow any media or program to be stored in any electronic storage medium that can be verified by the bitcoin blockchain and or executed without bloating the blockchain. At the time of development the only available storage for bitcoin was the OP_RETURN with a 40 byte limit. (There were 80 bytes available but there was serious discussion to limit the OP_RETURN TO 40 bytes to prevent bloat.)

The first project was to encrypt the Bible on the blockchain in a way that no one could modify even a single character. Because of the limited space in the OP_RETURN, it was not practical to create thousands of transactions and chain them together one output into another input to create the Bible.

It should be noted that Ordinals with Taproot has the same problem of bloating the blockchain and far less practical than the HlyGrail algorithm. In fact, Inscriptions in Bitcoin skirt the HlyGrail licensing (if done for profit) by just storing the data in the transaction as an inscription using more bytes instead of an external verifiable document.

NFTs are not practical in bitcoin and still are not with Ordinals because of blockchain bloat. Other blockchains like Ethereum infringe the HlyGrail Algorithm by using external storage such as IPFS effectively storing the data in the CID called from the blockchain and using a register like the OP_RETURN. In addition, all NFTs created for profit using the same technology of accessing external executable data via a blockchain and designated register infringe on the HlyGrail Algorithm. No matter what, the current method of storing NFTs all bloat their respective blockchains.

HlyGrail Method

The process introduced the concept of the Intellectual Property File (IPF). This is technically a proof of existence document tightly bound to a blockchain transaction by storing the hash of the IPF in the OP_RETURN of a bitcoin ADDRESS or other register in other smart contracts on other blockchains.

The **ADDRESS is stored in the IPF** and can be used as a key to encrypt and decrypt files including the IPF. The IPF could be any file as simple as a jpg (The ADDRESS watermarked inside an image then hashed. Like todays NFTs but provable ownership if copied.) to an encrypted computer program that verifies its own hash and / or embedded ADDRESS from the blockchain before executing.

The advantage of the bitcoin blockchain is it is not owned by anyone, the code is not under the control of an individual or group, no one owns major portions of the coins, AND the ability to know the unique ADDRESS (transaction information) prior to the transaction, which can be imbedded in the IPF. This means the ADDRESS can be known and embedded in the media before the hash is created, and before the actual transaction. **This is the HlyGrail, embedding the ADDRESS in the IPF before hashing.**

The IPF can verify the existence of unlimited media via the hashes of the specific media referred in the IPF, and the files can be kept unsecured virtually anywhere. The media can be duplicated anyplace.

The IPF and files it contains can be unencrypted or encrypted prior to creating the hash / hashes. Initially HlyGrail (486c79477261696c) was the prefix for the OP_RETURN to alert the hash was to an IPF file. It was not specific to the type of file or encryption method. The IPF file would need to be obtained and the owner would need to instruct the encryption to be able to verify or use the file.

The plan was to use different capitalization for HlyGrail to notify how the IPF was used and encrypted.

As example:

- HlyGrail required external knowledge to use the IPF correctly, either found on a website or via other documentation.
- HlyGraiL with a capital L might mean AES-256 encrypted.
- HLyGrail with a capital L might mean the file is executable.
- HLyGraiL with both capital L might mean the file must be decrypted and then executed.
- Hlygrail with lower case g might mean it is a JSON file.
- HlyGraIl with capital I might mean image.
- No matter what there are 256 combinations, and the format must be known and agreed on.

This is how the new version of IPFS uses coding type added to the front of the hash of the file before storing the CID. This was discussed in HlyGrail1. I would like to add HlyGrail and derivatives to IPFS.

The files were initially encoded by rijndael-256 mode NCFB, some modified to AES, some with and some without IV. Because different IV can change the encryption, you can only decrypt the IPF for consistency.

Update

After extensive usage, blockchain changes, and the shift to using the blockchain to store everything, I recommend the following practices for the HlyGrail algorithm.

- Use the bitcoin blockchain for storing the hash of the IPF. Bitcoin is extensive. The HlyGrail algorithm stores very little on the blockchain and all the tools like search are already built in. It takes virtually no energy to store and use the HlyGrail algorithm no matter how big the application. The IPF can be the media, or store to verify unlimited media including programs.
- Always use the receiving ADDRESS in the transaction because you can know it before the transaction, and no one, not even the blockchain knows it before it is published to the blockchain. Also, ownership can transfer by creating a new transaction output in the future.
- Prefix the Hash with HlyGrail (copyright) or capitalization for algorithm identification.
- Use the OP_RETURN of the transaction to insert the IPF Hash.
- Always use SHA-256 to determine the hash of a file. (In the future if more security is needed it can be changed to SHA-512 or other algorithm.
- If security is needed, AES-256 the IPF with ADDRESS as the Key. This must be the FINAL encryption before the hash, or the hash will change each encryption because of random IV. Decrypting with AES and the ADDRESS always works even without knowing the IV.
- Always include the ADDRESS in the IPF especially before creating the hash. (Unless the IPF is simple media like a jpg. You can watermark the ADDRESS in the image before hashing. If someone copies your IPF image you can prove it stolen compared to copying an NFT image.)
- Provide the Transaction ID to others to find the transaction, ADDRESS, and Hash.
- For online usage I recommend storing the IPF and documents on IPFS so smart contracts and other programs outside the blockchain can locate the data. You can store the files anyplace, but you need to identify the file locations in the IPF so programs can find additional data.
- Recommend using JSON for most IPFs. Include the ADDRESS, Titles, Descriptions, Owner, media title, and hash you are protecting, and more such as location of the file in the IPF.

Some Usage Examples (Unlimited possible)

Bitcoin IPF -> NFT

Individual File – The media is the IPF. (See the HlyGrail License below.)

- Determine a new receive ADDRESS for your IPF (NFT).
- Include a watermark of the ADDRESS in an image or metadata of other media. This and / or encrypting the file with the ADDRESS is key to the HlyGrail algorithm.
- Store the NFT (media) anyplace you want. I recommend IPFS but you need to keep it live.
- Create the SHA256 Hash for the NFT and pretend HlyGrail or use the IPFS CID.
- Create a bitcoin transaction and spend to the new receive ADDRESS and store the Hash or CID of the media in the OP_RETURN of the transaction.
- DO NOT CHANGE your NFT after committing to the blockchain or the hash will not match.

Your NFT can now be verified by the transaction and easily read or decrypted from off chain storage.

Sell it by spending it to the new Owners ADDRESS and include the file. If in IPFS the CID is the file. If IPFS ever drops your NFT just put it back up and it will have the same CID. The power of the HlyGrail IPF is it can be a billion pixels and not bloat the blockchain. Anyone can have or serve a copy, but only the owner in the IPF or the private keys can sign the message verifying ownership.

Create an Attribute IPF (NFT) if you want features and other information on the blockchain.

IPF Attributes (NFT) (Get a HlyGrail License for commercial use. You need a license to create IPFs for others, storing, or providing services like an auction site or embedding ADRESSES.)

- Create the media IFP as above for the individual NFT.
- You can create additional support NFTs such as a 2D image, 3d model, story text, and audio. Each would have its own separate IPF ADDRESS in the media (NFT).
- Store the IFPs or NFTs (media) on IPFS or your server. If you load to IPFS, keep your file alive on the network. Maintain the original IPFs in case you need to load it/them back to IPFS.
- Determine a new unused Bitcoin receive ADDRESS for the **Attributes IPF**.
- Create an IPF file with JSON with any name or use the IPF ADDRESS as the filename.
    - Store the new Bitcoin Address in the IPF. (This and / or encryption is the HlyGrail.)
    - Store the original owner Name and contact information.
    - Store the NFT / NFTs title, description, and any other information such as file type, stats, or abilities AND file locations.
    - Store the NFT / NFTs Hash / Hashes or CID /CIDs of the media.
    - Since it is JSON, you can store anything as key / value pairs, and the reading program will decide what is needed. Expose the attribute Keys so others can use your NFT.
- Save the IPF in a file with or without encryption and create a Hash or on IPFS for a new CID.
- Create a new transaction with the Attributes IPF ADDRESS and store the IPFS CID or HlyGrail Hash in the bitcoin blockchain OP_RETURN of the IPF ADDRESS.
- As the owner, keep the IPF files and attributes IPF file, ADDRESS, and transaction along with your private Keys to all bitcoin addresses in the event you want to sell, decrypt, or verify any files. Each media is a separate IPF / NFT to be used and controlled separately as needed.

Sell or Upgrade the Attribute NFT

Create a new IFP JSON file with a new unused receive ADDRESS.

- Store the new ADDRESS in the IPF
- Store the new Owner and contact information.
- Store and new features as key value pairs
- Store the original NFT / NFTs Hash or CID (This provides the media quickly.)
- You can add additional multiple media file Hashes or CIDs.
- Store the old Attribute IPF Transaction (Makes it easier to access.)

Transmit the new IPF with the new owner or data.

- Store the new IPF JSON safely or on IPFS for a new CID or other storage.
- Create a new transaction and spend from the previous IPF ADDRESS to new IPF ADDRESS and store the new IPFS CID in the bitcoin blockchain OP_RETURN of the new IPF ADDRESS.
- Give the IPF file, ADDRESS, and transaction to the owner of the transferred NFT.

Servers can serve the NFTs and / or the most current IPF to requestors along with the IPF ADDRESS if encrypted. Systems like the metaverse can include the NFT once verified.

Bitcoin Patent or Intellectual Property File (IPF)

Example usages on https://mycryptopatent.com such as Universal ID, secure medical data, and more.

Same as Attribute IPF but way more powerful. You use the IPF usually with JSON to contain:

- ADDRESS
- Owner
- Title
- Description of Patent or other extensive documented media.
- Number of Reference Documents
- Include Array of Multiple Reference media:
    - Title
    - Description
    - File Name
    - File Type
    - File Hash
    - File Size
    - Encrypted AES256 Y/N with ADDRESS
    - Hash Type SHA-256 or other

Media – Audio, Video, Images, Documents of any type - Protection from Modification

Create an IPF that includes: (JSON is best.)

- ADDRESS (Include the ADDRESS in a watermark or other metadata of the media)
- Information such as Title, Description Owner etc.
- Hash of the Media (SHA-256 usually acceptable. State if different.)

In a new Transaction with the ADDRESS, store the Hash in the OP_RETURN.
Share your media as desired and alert the public it is protected by the IPF Transaction.
If someone improperly uses your media, you can prove ownership with the IPF.

Computer Program

This is a smart contract application as we tend toward Internet Computing. This is a way to utilize all unused computers for parallel programming and a way to verify a program cannot be modified by virus.

Create an IPF that IS the program. (In ANY programming language.)
The program contains:

- ADDRESS of the blockchain transaction
- You run the program by feeding in the Transaction as a parameter.
- The program calculates its own SHA256 Hash
- The program looks up the transaction and gets the ADDRESS and OP_RETURN Hash
- The program verifies the Hash and / or the ADDRESS of the program before proceeding.
- Additional verified data can be stored on IPFS or elsewhere plain or encrypted by including their Hash such as a CID in this IPF program. If the data IPF is encrypted, the data could be verified by the hash, loaded then decrypted with the ADDRESS before using the data.

Create a transaction with the ADDRESS and load the Hash of the IPF program into the OP_RETURN.

This can be further secured by encrypting the program with the ADDRESS and a preprocessor would need to be run feeding in the transaction and decrypting the program before it could run.

Data files can be stored by several methods depending on the application.

- They can be stored on the machine in traditional databases for local programs. Only the local machine needs the files and stores results. This would work like a windows executable file.
- They can be stored traditionally with encrypted API calls with legal endpoints stored in the computer program for multiuser requirements. This would work like an earthquake application on a computer but polling encrypted servers for the data. (Sorry Chainlink, IPF was first.)
- Data could be stored in servers like multi player video games with two-way communication between the client program and the server via encrypted API / REST calls.
- Initial data could be stored in IPFS and verified by the IPF before running the program. (The data file should include the ADDRESS of the program. The file location, and data file hash would need to be included in the program before the program was Hashed and stored.)

Variables could be stored with Taproot like Ordinals but or in an IPF.

- Initial variables and the new IPF ADDRESS stored in IPFS as a new CID then stored like the IPF example. The file location (IPFS as example) would be included in the computer program.
- The computer program would read the CID in the program code, look up the initial IPF bitcoin transaction, follow the spend chain (see below), verify the ADDRESS and hash then load the variables from the CID or file, compute the program, and derive new state variables.
- The program would create a new IPF JSON file with the new variables and new receive ADDRESS and store the file to IPFS with the new ADDRESS for a new CID.
- The program (holding the bitcoin keys) would then create a new IPF spending from the current variable ADDRESS to the new variable IPF ADDRESS. The program on startup would follow the variable transactions to the end and input the latest variables before processing.

Variable and how they are stored need further study. The above IPFS is one method but would be slow. In general, the variables need to be available on a fast service or possibly in Taproot. As stated, they could be traditional secured databases using API or REST calls identified in the initial IPF Program.

The exiting part of using a computer program stored outside the blockchain, besides not bloating the blockchain, is the possibility of concurrent programming. Once the file is saved, thousands of people could download the program verified by the blockchain and run the copy securely. The application could be built to take inputs and provide outputs. As example, a bitcoin block calculation could be run with different nonce. A master would set up the current block transactions. Feed to the network the data and thousands could run with different nonce. CERN could calculate the next value using thousands of independent computers running verified parallel algorithms and paying for answers computed. Instead of letting the computer heat the house when not used, it could be paid for doing the next science project. Even if my computer was old and took 10 times longer to get an answer than others, it would still contribute to the process. The beauty of computing with HlyGrail is that the programs take up very little blockchain space but provide rapid access to verify off chain code to be run independently by thousands of users. Apps could chain function to function using parallel processing.

Blockchain Size for a full node.

The entire blockchain up to the current time could be stored off chain but verified by a single blockchain transaction. This could be built into the blockchain software. Every x number of blocks a new hash could be computed to that block, the blockchain stored off chain, and verified in one transaction linked forward from the last verification transaction. This would make it a lot easier for new full nodes to get up and running. New node operators could download the most current blockchain securely then verify much faster before adding new data.

Look up other methods on the MyCryptoPatent website, such as universal ID, protected medical data, marketing system patents, t-shirt design, media, compensation systems and more. Virtually any computer program can be controlled and verified by this process.

Standards

File standards have always been an issue in discussions because no one can realize the full power of the HlyGrail algorithm. As example, no one could demand that every software application use one programming language. Because of the diverse use of the algorithm, there can be no fixed standard for file types. However, some attempt was discussed above and possibly the IPFS version approach to CID might cover many possibilities. As stated, there are 256 different combinations with HlyGrail.

I use the Bitcoin Patent Intellectual Property File (IPF) in JSON as described above for many files. This format ties the document to the blockchain ADDRESS, the owner, title, and general description with an unlimited number of reference documents provable by their hash in the IPF but without exposing the secret details other than title and basic description of the file. You could verify the Library of Congress with each document title and hash in the IPF using a single transaction hash.

As example, if you want to patent the design of an airplane, the IPF could contain the hash for audio, video, schematics, funding, facilities, processing, contracts, and any other data plain text or encrypted in a single document and registered on the blockchain. You control what documents you want others to see verifiable via the hash in the IPF without exposing other provable information to competitors.

A general standard for an IPF as an attribute NFT would be to use JSON. You can put any attributes you want in the code with Key Value pairs including multiple images or other media hash with links to the data. Anyone knowing the format could extract or ignore any Key.

You could have a 2D Icon and a 3D character each watermarked with an IPF ADDRESS stored with independent CID on IPFS. The JSON IPF would then be created with the ADDRESS and contain:

```
{{“address”: “ADDRESS”,
  “owner”: “OWNER”,
  “title”: “Icon and 3d Character”,
  “description”: “My fabulous Metaverse Character with green eyes.”,
  “numberFiles”: “2”,
  {“media”: [
    {“title”: “Icon”,
     “description”: “2D image”,
     “fileLocation”: “IPFS”,
     “fileHash”: “CID”,
     “filetype”: “jpg”},
    {“title”: “3D Character”,
     “description”: “3D model”,
     “fileLocation”: “https://mywebsiteapi.com?3dcharacter”,
     “fileHash”: “HlyGrailxxxxxxxxxxxxxxxxxxxxxx”,
     “filetype”: “obj”}
  ]},
  {“assets”: [
    {“title”: “Health”,
     “description”: “Character Health”,
     “health”: “100”,
    {“title”: “Weight”,
     “description”: “Weight in pounds under Earth Gravity”,
     “weight”: “450”
 ]}
}
```

The ERC721 and ERC1151 Metadata JSON are attempts to integrate IPF features but still miss the power of the HlyGrail algorithm. To move MOST if not all storage and programming off chain you need to add the IPF ADDRESS to the media prior to hashing and storing the media, then storing the hash in the new OP_RETURN of the IPF on the blockchain. Ethereum and other single address blockchains would need to use a register with a unique value that is used as the ADDRESS in the IPF. This could not be guaranteed unique and unknown before use rising many security issues for ownership.

The HlyGrail algorithm stores the ADDRESS in the media before recording the IPF NFT to the blockchain. You can run any smart contract using any language on or off the blockchain verifying the ADDRESS then loading and using the IPF NFT. Transfer ownership or update the NFT by transactions from the current IPF to a new IPF or just spending to a new ADDRESS with no changes.

Bitcoin works best in the Purist Mode where the transaction and ownership are for ASSETS without bloating the blockchain like Ordinals and other crypto blockchains. Use Taproot capabilities to manipulate the HlyGrail algorithm programs and assets off chain or variables. Bitcoin can now be used as intended to transfer the ownership of the assets in transactions and not hold expensive unnecessary data on the blockchain. A file verified using the IPF proves the data is unique and immutable without storing data on chain. To access the secure data, you do not need to run a full node and the files can be stored for easy access anyplace, without needing access and download using blockchain smart contracts.

Summary

The HlyGrail algorithm uses a new unused receive ADDRESS that is either watermarked into an image or included in metadata of other media, and / or included in an Intellectual Property File (IPF). The IPF can contain reference to multiple media and include their SHA256 hash (or other hash). The media or IPF is either plain or AES-256 (or another encryption) encrypted. The final file or IPF is then SHA256 hashed (or another hash algorithm). The Hash / CID is then stored in a new blockchain transaction using the new ADDRESS as the receiving address with the Hash or CID placed in the OP_RETURN or other permanent register of the blockchain transaction. (If you do not include the ADDRESS in an image or protected document as currently done with NFTs, anyone can steal it.)

The HlyGrail license covers ANY blockchain that uses this or similar algorithm by using a unique ADDRESS or other unique part of a transaction that is known before the transaction which is embedded in the data (image/media, program etc.), and storing a Hash of the data in a permanent register (OP_RETURN) of the transaction. The hash identifies a unique data object and even with collisions would be easy to determine which file was intended. The intention is to use the ADDRESS from a transaction to tie off-chain media plain or encrypted back to a blockchain transaction for verification of ownership and authenticity while keeping the media usable off the blockchain.

Final Thoughts

I never pushed this algorithm for several years waiting for the bitcoin blockchain to mature. I have watched inefficient and expensive NFT solutions with concern as they bloat their blockchains. Now that Taproot has been installed and processes like Ordinals are bringing bloating to the bitcoin blockchain, it is time to let the world know about the HlyGrail Algorithm and the full power. I have planned this since 2012 and watched idiots say the bitcoin blockchain is just a money game and cannot do smart contracts. This algorithm provides a real value and UNLIMITED use cases for only one OP_RETURN. I have been able to build any smart contract or IPF Patent for years for the bitcoin blockchain more efficiently, less expensive and with any level of security.

I can hardly wait to see the explosion of the IPF used as NFTs. Put a virtually invisible watermark of your IPF ADDRESS on your NFT. Use a slightly different color pixel than your main background to add the ADDRESS. Computers can find it instantly, but the eye cannot see it. Then store it with the "Individual File" method above on IPFS and record the IPFS CID in the bitcoin OP_RETURN. You are done. Make something real and sell it! Make a whole 3D virtual world instead of a pixelated drawing and only use a few dozen bytes of the blockchain. Real art, not a money game.

I would recommend minors not to process Ordinals and other transactions that bloat the blockchain without a huge fee if ever. Programmers should use Taproot to write smart contracts that manipulate the IPF and storage. (With a license of course.) Reasonable minor fees for IPFs would be expected.

I have a software system for creating the IPF. I am looking for investors to license the use of it while I improve and make this tool available as a patent tool, auction site, and IPF locator for the world.

License and Commercial Use

The license in the original HlyGrail is stated below with a new donation address and updates for clarity. Let me interpret the license for those who do not understand the purpose.

The intention is to allow ANYONE or business to personally create an IPF and secure it to the bitcoin blockchain. The intention is to use the ADDRESS from a transaction to tie off-chain media back to a blockchain transaction for verification of ownership and authenticity. There is NO license required if the creation of the IPF and OP_RETURN on the blockchain is accomplished personally without any unlicensed outside tool or service designed for IPF creation.

The creator owns the rights to their IPF, and underlying media is protected by the IPF by Common Law. They can do what they want with it. They can encrypt, decrypt, store, send, receive, sell, verify or other process requiring the HlyGrail Algorithm without a license. However, they cannot use or build an auction site to sell IPFs including their own or assist others without a license.

Services would be defined as software or individuals utilizing the HlyGrail algorithm for profit without a license. The main commercial restriction in the license is to prevent companies from claiming rights to the Algorithm, charging users for profit, and especially blocking others from doing the same.

This includes services that use the algorithm for creating, finding, storing, serving IPFs, referencing, verifying or proof of ownership, creating an auction website, marking assets (watermarks or other metadata) with the ADDRESS, manipulating an IPF and protected data, creates a program, or other use that uses the HlyGrail Algorithm for profit including AD revenue, must have a license.

As example, a metaverse that loads in a 3D model protected by the HlyGrail algorithm, then uses the IPF to verify ownership through message signing or other method can do so without a license, IF they do not charge for the process. If they charge to use the character in the metaverse, they need a license. Likewise, the metaverse is not allowed to use the 3D model without the consent of the owner without violating the owner's copyright and the IPF license.

For those companies who have, or intend to, violate the patent, get a license, or prepare to pay according to the general license. I hereby authorize ANYONE to take low level violators to Small Claims Court and file a suit as a protector of the HlyGrail Algorithm and keep 100% of any settlement. Low level violators (Small Claims Court cases) are entities trying to make a profit with the HlyGrail Algorithm without a license even if only for AD revenue or minor personal gain.

In the event of large entities such as companies where the damages are beyond Small Claims Court, ANYONE can report the violation and after all court costs and legal fees, 50% of the settlement including any damages, if any, will be provided for helping protect the Algorithm while keeping it free to use by the public. 50% to the author. This includes hardware implementations.

Bitcoin and Litecoin can both build into their core software the ability to use the algorithm. Other blockchains must obtain a license to build it in or use it. I have discussed this algorithm and the license which has been on the blockchain for years. Ignorance of the law and the patent / copyright is no excuse. Trying to get around the patent by using other blockchains or modifications is a violation.

License fees vary by use and case. You can find links to the current fees at https://hlygrail.com. Most could get a license free sending an email requesting to make a few IPFs for the family. Most fees will be low or will be provided when buying services from https://mycryptopatent.com. Even large entities building the application into a cell phone wallet IPF application will find licensing favorable, especially over a lawsuit if you do not get a license. Most applications, improving the growth and use of bitcoin will be favored. Providing auction sites for IPF NFTs or creations of IPFs? – Get a License.

Original License with slight modification for readability and new donate address.

Permission is hereby granted, free of charge, to any person obtaining the knowledge of this algorithm (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

1. Free personal / business use is allowed in all instances to create, use, and sell their IPF or part of it with no license required. Free use does not include building or using as example auction software to sell IPFs including their own without a license.

2. Commercial use requires a SPECIFIC licensing agreement. That agreement must be paid to and signed by the author allowing the use of the HlyGrail algorithm prior to use. Commercial use means using the software in a way to make a profit FROM the software not your IPF. As example, you can make a patent with the software and it is yours with no restrictions, but you cannot make software to create, serve, or auction IPFs without a license. Another example would be a musician can protect their music with the IPF with no license, but a music company cannot sell their IPF music without a license. This is not an exclusive list but an example to demonstrate the difference between personal / business use of the IPF, and making money with the IPF HlyGrail Algorithm.

If you do not have a specific licensing agreement, then you agree to a general license agreement that is, by your use of the algorithms or derivations of it for business (profit or not for profit) purposes, that the author of the HlyGrail algorithm equally owns 50% of all your assets related to the use of the HlyGrail payable on demand in exchange for the past or current use of the HlyGrail Algorithm.

 You can and are encouraged to use HlyGrail as a preface to your hash when making an IPF.

The above copyright notice and this permission notice shall be included in all copies of software algorithms but not necessary in simple image IPF NFTs.

The author (Roger Johnsrud) can be reached through https://hlygrail.com.

Donations are appreciated to bitcoin address: **1God28A8CahUcrLjtX4UQouDvdWVqQXSya**

Check the https://hlygrail.com website for other donation or transaction addresses.

HlyGrail Patent/Copyright Number bc1qh9q5gasa7vzwymfustn4fjexw8xuw77gckwut4