HyperChain: A Lightweight Second Layer Blockchain Platform

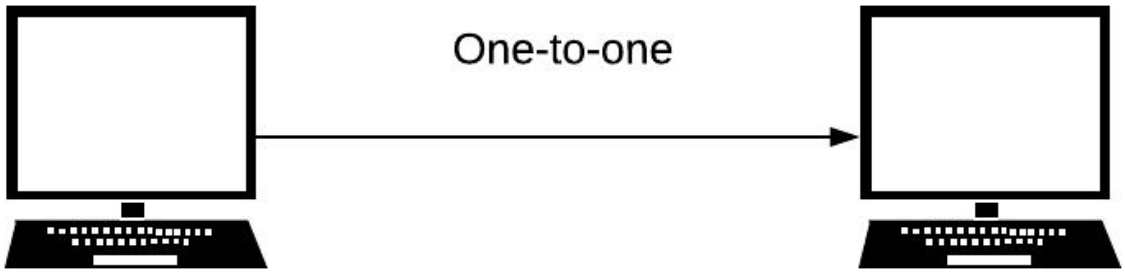
The HyperStake Developers
www.hyperstake.io

Abstract. HyperChain provides an easy, low cost, and light weight method for provably secure publishing of data to the HyperStake blockchain. HyperChain provides a non-restrictive template for publishing data and communicating with listening devices over the blockchain. The nature of the non-restrictive template allows for many use cases and gives 3rd party developers a convenient platform to develop projects that require immutable data, secure communication, one-to-many communication, and more.

1. One-to-One Relationships

Bitcoin, a decentralized peer to peer digital currency, established that transacting can occur in a decentralized secure way between two parties, or even multiple parties (in the case of a multi-signature transaction transactions) [1]. Bitcoin is very good at transferring value from one person to another, but is not intended to be used to do more than simple value transfer.

Several digital currencies have begun focusing on micro-transactions [2][3], which can send very small amounts of value with little or no fee. Although micro-transacting is important, current blockchain projects are focusing primarily on one-to-one relationships where the sending party already has knowledge of the receiving party of the transaction. This system works well for value transfer, the sending party is able to have complete control over whom receives their value and/or data, thus representing a one-to-one relationship in the transaction, one sending party paired with one receiving party.

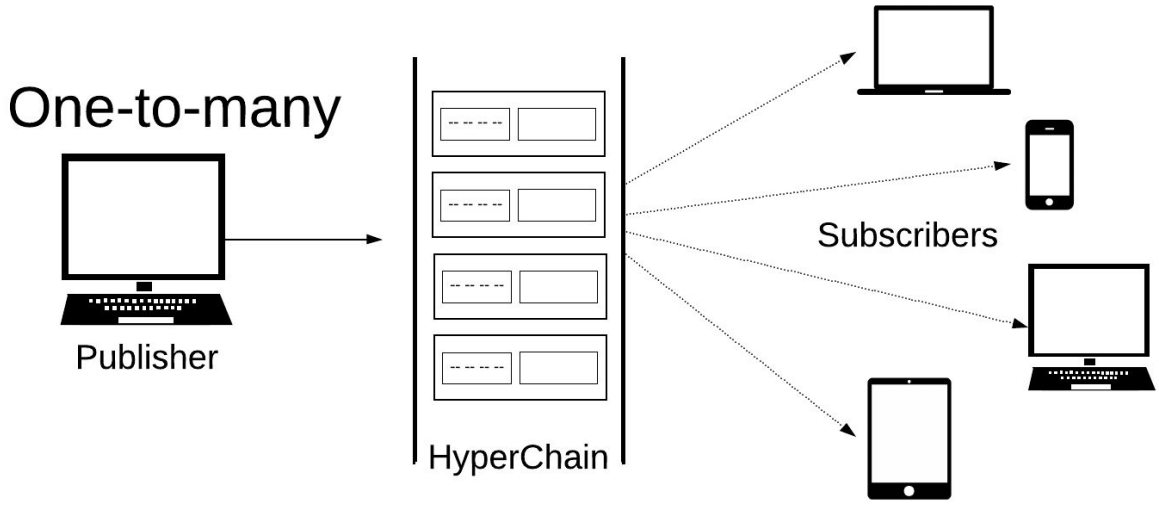


As the blockchain industry continues to expand beyond digital currency use and begins to focus on the benefits that immutable data records provide, one-to-one transaction style begins to restrict the usability and potential use cases.

2. One-to-Many Relationships

One-to-one relationships work well for specific use cases, but are not efficient when using a blockchain to publish immutable data that is intended for a group or even for an unknown end user (i.e. the general public). For example, most websites published today on the internet do not have precise knowledge of exactly which users will be connecting to their web-server, if knowledge of the end user were required it would be extremely difficult to gain widespread use of the website.

One-to-many relationships are much more efficient for many use cases, for example one-to-many has effectively been used to send television and radio broadcasts from one party to multiple parties, there is a single signal that is broadcast from a television studio and many recipients receiving the exact same broadcast on their television. The sending party in a one-to-many relationship saves enormous amounts of complexity and data, while also allowing for a model where the receiving party chooses to listen to a certain channel rather than the sending party being required to have knowledge of the recipient. A one-to-many model is much more efficient for public use of blockchains and immutable data.



3. Introducing HyperChain

HyperChain provides the ability to securely communicate in a one-to-one or one-to-many relationship between devices connected to the HyperStake network. HyperChain provides a

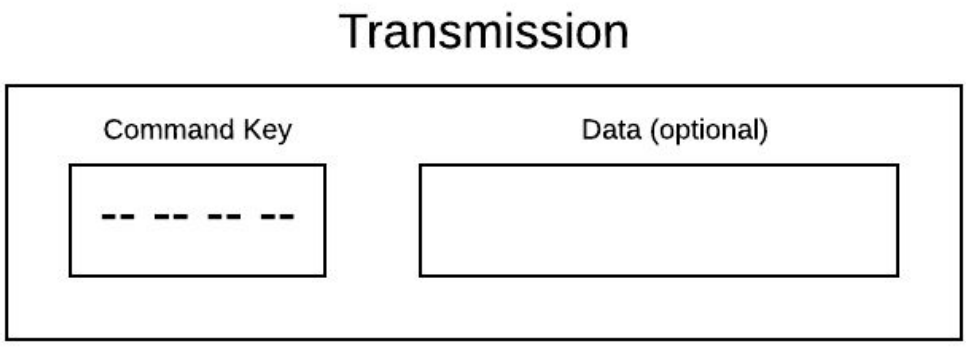
platform for users to create channels that broadcast transmissions and data into HyperStake's blockchain. Recipients are able to monitor HyperChain for broadcasts that belong to channels that they have subscribed to and react to those commands and/or data accordingly.

By leveraging the security provided by ECDSA Secp256k1 public/private key cryptography, a node (device connected to the HyperStake network) is very easily able to confirm that any communications broadcasts to a HyperChain channel are from a device that is *whitelisted* (approved to broadcast from a channel).

HyperChain uses the concept of *subscribing* to transmissions. Transmissions are commands that are sent to a HyperChain address (channel) on the HyperStake network. By subscribing to a channel, a node simply watches for transactions sent to a specific address, sees if the command was sent (and cryptographically signed) by a whitelisted node (essentially a public key), and finally checks if the command is one that is recognized and approved. If the command sent to the channel is recognized by the device monitoring the channel, then the device will respond in any way that it chooses, such as by running an external script or operation.

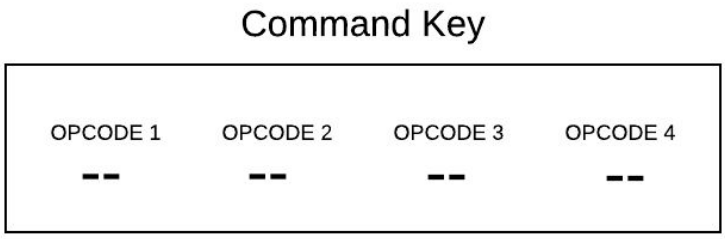
4. Transmissions

A transmission is published to a HyperChain channel by a publisher. A transmission is comprised of a *command key* and an optional *data payload*. The transmission is added into a HyperStake transaction and sent to a HyperChain channel. Transmissions can be used to send commands to a channel, or to send data. For example, Alice sets up a HyperChain channel and hooks up all of her homes smart door locks to monitor the channel for unlock or lock messages published to the channel. Alice only needs to use the command key component of the transmission to send an unlock/lock command to her home's devices and the additional data field does not need to be added. Bob has his home's smart locks set up to look at the command key and then lock or unlock based on a time delay added in the data payload section of the transmission.



5. Command Keys

Nodes react to channel transmissions based on the *command key* contained in the transmission. Command keys can tell the subscribing node to perform a certain action that is paired to the specific key, or alternatively they can tell the subscriber how to interpret the following data payload.



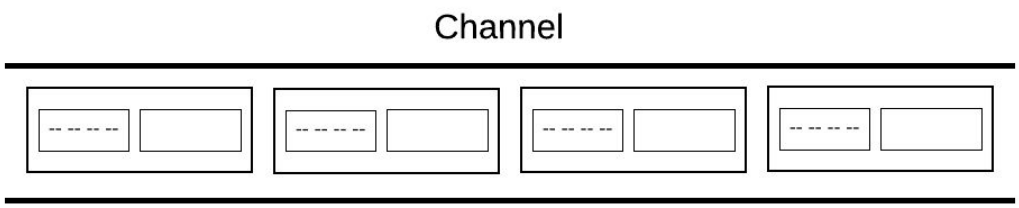
Command keys are unique to each channel and allow for complete flexibility for the channel's development team or publisher. A command key is a very small piece of data (4 bytes/32 bits) that leverages 29 of Bitcoin's unused scripting language operation codes (OP_CODE) within the redeem script of a transaction output. HyperChain provides room for over 24,000 (29_4) lightweight customizable commands, as well as the ability to pass additional arguments and data with the associated command.

6. Transmission Filtering

Transmissions can be sent by anyone and to any channel. The subscribing node decides which commands they watch for, and who is able to send valid transmissions to the channel. If the transmission is sent by a device that is not approved, then the command is ignored.

7. Channels

Transmissions are sent to channels. Within the HyperStake network, a channel is seen as a HyperStake address that starts with the letter X. Channels may contain many different transmission from many different publishers and the subscribing node has the ability to ignore certain transmission by using a filter that discriminates by publisher or command.



8. Subscriptions

HyperChain allows a network of devices (nodes) to be run, monitored, or controlled without the need to connect to each individual node using ports. Given that a node is actively running HyperStake, they will be able to monitor HyperChain for communications that have been securely sent to a channel that the node is subscribed to. The ability to subscribe to channels on HyperChain not only streamlines communicating with a single node or device, but also gives the ability to have an entire network of devices monitoring a channel to perform operations based off of a single transmission at the same time (one-to-many), which means the creator of the transmission would only need to create and send the command once and still can effectively have sent that command to a range of listening devices.

9. Transmission Fees

In order to prevent spam, sending a transmission requires an associated fee. The fee will be based on the current amount of transactions that are being processed by the HyperStake network, and will have a fee market which determines that the highest fee per byte will be the most likely transmission to be placed into the next block. As with typical HyperStake transactions, as many transmissions will be added per block as is possible. Although publishers will want to add fees to their transmissions, it will also be possible to establish a zero fee system. The HyperStake network does not actually require transaction fees, stakers are able to include whichever transactions into the block that they decide. If many stakers enjoyed the use of particular HyperChain channels, they could begin including transmissions to those channels into a HyperStake block fee free, this would likely have the caveat of higher average times to be included into a block.

10. Theoretical Examples

Mining Rig Manager: Imagine that Joe runs a cryptocurrency mining operation that has 10 different mining machines. Each of these mining machines are mining the same cryptocurrency called *CoinABC*. Joe notices that there has been a blockchain fork on the *CoinABC* network and would like to ensure that each of his miners are on the correct fork. Joe broadcasts a transmission to a HyperChain channel that his 10 miners subscribe to. The transmission tells his miners to run a script that will reorganize to the correct fork according to the checkpoint he added to the data payload of the transmission. Using only one simple command, Joe was quickly and securely able to tell each of his 10 nodes what to do.

Exchange Rate Database: A channel is setup nicknamed *BTC-Prices* that has a purpose of storing Bitcoin/USD exchange rates. A publishing server is set up that finds the BTC/USD rate

every 10 seconds from Exchange-1, Exchange-2, and Exchange-3. The server processes the price information into its transmission format, and sends a transmission for each different exchange prices with a unique command key that designates the exchange that the transmission is referring to. A portfolio monitoring application on Bob's computer is connected to HyperChain and is subscribed to *BTC-Prices*. Bob only uses Exchange-2 and sets his subscription to ignore any channel transmissions that are not for Exchange-2. When Bob checks his portfolio values, it quickly grabs the information from HyperChain instead of having to send requests to Exchange-2.

Game Engine: A game is created that uses the HyperChain for its network connection, multi-player discovery, and pseudorandomness. A game is setup that creates a new channel for users that are looking to discover other users to play with. Bob subscribes to the multiplayer discovery channel and finds a game to join with Alice. Bob and Alice begin playing the game by starting a new HyperChain channel dedicated to their communications. The game is a turn-based board game and uses HyperStake's blockchain to determine randomness for rolling dice that determine where the user lands on the board. Bob's turn starts when the next two HyperStake blocks have been added to the network, and the dice are rolled using the block hashes as seed entropy for the pseudorandom number generator. Bob lands on a square that makes him select from options A-D. Bob selects option D and ends his turn. Bob's selection and end of turn are broadcasted to the HyperChain on the channel both him and Alice are subscribed to for the game. Alice sees Bob's move since she is subscribed to the same channel. The game repeats these events until some other events are triggered that end the game.

Token Platform - A specification can be created that will utilize HyperChain's channels and transmissions in order to encode a token. For example a channel could be setup for a particular token HYP2. To use HYP2, users would need to run the associated software developed by the HYP2 developers. This software would not have to concern itself with creating its own secure blockchain, but would issue tokens using its channel on the HyperChain. The HYP2 developers may decide that they would like to raise funding for their HYP2 wallet software and give a HyperStake address to the public which they can send HYP to in order to fund development. When HYP2 is ready for release it sends HYP2 tokens to each address that sent HYP to the development fund and sends 10 HYP2 for each HYP sent. HYP2 transactions would be able to use the existing safety of the secp256k1 key pairs already used on for HyperStake transactions, but would transact in HYP2 tokens instead of HYP. Since HYP2 transactions use HyperChain, they would still need to use HYP as a transaction fee in order to be added to the blockchain.

The Internet of Things (IoT) - The decentralized blockchain protocol of HyperChain enables public distribution of private data without identification. Running the HyperChain IOT node will enable the publishing of individualized data points through subscriptions to transmissions on

channels. Parties interested in a subscription to the data will reward the publishing nodes in exchange for their data collecting and contribution. Biometric data collected through modern wearables powered by PPG sensors, gyroscopes,and accelerometers, will play a role in the first blockchain of human bio-data that can be used for medical studies. Privacy is maintained through the blockchain where only the public address and other selected data is published willingly and with a reward. Through a reward system, users opt for allowing the collection of their data. Giving an IoT node the control over data it collects further decentralizes data, where conglomerates and centralized organizations are not owning your data and profiting from it, (Google, Facebook) rather, the party creating the data can contribute to a larger pool of data and receive the award. The value in the data lies in the volume and ability to filter for specific purposes.

References

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”
[2] “Lightning Network: Scalable, Instant Bitcoin/Blockchain Transactions”, https://lightning.network/
[3] Colin LeMahieu, “Nano: A Feeless Distributed Cryptocurrency Network”