BitBlock

A Decentralized Data Delivery Network (3DN) for User-Created Videos

Abstract. A decentralized implementation of a video-sharing application would allow for both viewers and creators to better monetize their efforts and attention. Using a novel digital token, advertisers could pay smart contracts to display advertisements on the application, and, in turn, the smart contracts would pay viewers and creators for using the application. This provides a financial incentive for both viewers and creators to migrate from currently popular video-sharing platforms.

1. Introduction

Popular video-sharing platforms, such as TikTok and YouTube, extract large portions of value from creators and viewers. Most of the advertising revenue is kept by the centralized company for servers, employees, investments, and profit. A decentralized competitor could alter these categories in ways that allow for a significantly larger portion of revenue to go to users.

Centralized data storage servers can be replaced with the InterPlanetary File System (IPFS) in which peers can easily exchange content. Filecoin, a decentralized storage market, can augment this content sharing by ensuring that videos can persist indefinitely. Centralized computational servers can be replaced with local computation and smart contract blockchains, such as Ethereum, in which computation is decentralized and automatic.

Costs associated with employees are unnecessary, and most investments are unnecessary (as scaling to user growth can happen automatically). Finally, and perhaps most importantly, the concept of profit extraction does not exist in this model. All of these (previously necessary) costs can instead be funneled into users' wallets, which would attract new viewers and creators and incentivize the creation of better content.

To implement these decentralized and financial concepts, we propose an application that takes advantage of decentralized data storage (through IPFS and Filecoin) and decentralized computation and metadata storage (through Layer 2 Ethereum solutions) to allow users to upload their own videos and stream others' videos.

2. Data Storage

The InterPlanetary File System (IPFS) is a protocol and peer-to-peer network for storing and sharing data in a distributed file system. IPFS uses content-addressing to uniquely identify files across the entire file system. Filecoin, a related project, is a decentralized blockchain and storage market that allows clients to negotiate deals with storage miners to provably store their data for a period of time.

When a user posts a new video through the application, it should automatically be stored on the Filecoin network for the default period of time. Using services such as Textile's Buckets, this initial upload should also pin the data on IPFS in order to be easily retrieved by viewers. Because IPFS and Filecoin use content-addressing, each video will be identified by a single IPFS Content Identifier (CID) which we will refer to as a Video Identifier (VID). Advertisements are treated similarly to regular content: they are stored through IPFS/Filecoin and retrieved through their VID.

We can use the chosen Layer 2 Ethereum solution to serve as a master database of VIDs and their metadata. This is further described in Section 3.2.

3. Computation

Ethereum is a blockchain, a decentralized ledger of transactions, that allows for the automated execution of code (i.e. smart contracts). Using the base layer (also known as Layer 1) of Ethereum is expensive and slow. To serve the needs of a scalable video-sharing application, we should use a Layer 2 solution such as the Polygon Network. The backend computation of this video-sharing application will exist in the form of smart contracts on Polygon; this is typically referred to as a decentralized application (dApp).

3.1 Video Ownership

Creators must be able to prove ownership of their videos. Thus, when a creator uploads a new video, a Non-Fungible Token (NFT) is minted and sent to their wallet. This NFT will contain the VID and metadata about the post (timestamp, etc.).

The creator rewards for viewership (described in Section 4) will be sent directly to the public address which own the NFT. Thus, the main smart contract will need a reference of who owns each and every NFT. In order for this to be possible, transferring of the NFT will necessarily go through the main smart contract first. For example, if User A wishes to transfer the NFT to User B, they must send a special transaction to the main smart contract which results in logging the new owner as User B, and then sending the NFT to User B.

3.2 Metadata

Certain interactions with the local application/website are sent on-chain to the BitBlock smart contract. These interactions include: viewing, commenting on, liking, disliking, and reporting a specific video.

3.3 Recommendations

In the first iteration of BitBlock, the recommendation algorithm will be a simple random batch of videos. However, the probability of each VID will depend on its Likes-View ratio and Dislikes-View ratio. Upon launch, the client will request a certain number of VIDs from the main smart contract, which will return random VIDs. In each batch of size X, there should be some number of promoted videos, based on the user's preferences. Once the client is close to exhausting this batch of videos, it will automatically request another random batch.

For example, let's say batches are of size 10, and User A's preference is 1/10 of their videos are advertisements. Upon launch, the application should receive 9 regular VIDs and 1 promoted VID (in an unpredictable order). After having watched, say, 5 of these 10 videos, the application should request another batch of 9 regular VIDs and 1 promoted VID in order to avoid potential downtime of having to wait for the next batch.

4. Token and Payments

Using a novel Ethereum ERC20 token, financial incentives for all parties can be added to the ecosystem.

4.1 Advertisers

Advertisers will be able to upload and promote content through the application. There is no clear distinction between users and advertisers; users can act as advertisers and promote any content they choose. After uploading a normal video, advertisers/users can send a transaction to the BitBlock smart contract to promote their video a certain number of times. This will cost a certain amount "per-view". After taking a small cut (to be used for further development), the funds are ready to be sent to viewers and creators.

For example, let's say Company X wants to show their advertisement to 1000 people, and it costs 0.01 BitBlock tokens per view. After uploading their video, they will sign a smart contract transaction which sends 10 BitBlock tokens to the BitBlock main smart contract along with "CID: xyz" and "numPromotedViews: 1000". Every time a viewer is shown the promoted video, they send a View action to the main smart contract, and the numPromotedViews for that video is decremented. This viewer is awarded some BitBlock tokens for watching the advertisement. When numPromotedViews reaches 0, the advertisement is no longer displayed to viewers.

4.2 Creators

Theoretically, for every view a video receives, some amount of the token is paid to the Ethereum wallet which holds the NFT.

In reality, the sum of all creators' revenue is directly related to the total network revenue from advertisers. Thus, at a specific time every day, the entire smart contract's advertiser revenue is split proportionally between creators and viewers. For example, imagine that 80% of advertiser revenue is chosen as the desired percentage to go creators. Imagine that advertiser revenue for today was 1,000 BitBlock tokens, the number of total views on the system was

100,000, and Creator A received 100 of those views. In this scenario, Creator A should receive 1 BitBlock token.

4.3 Viewers

Every time a viewer watches an advertisement, the main smart contract will pay a fixed amount of the token to their wallet.

5. Dissuading Bad Actors

The mixture of profit and anonymous accounts opens the door for greedy creators and viewers to act in dishonest ways in order to maximize their income. Features will need to be built into the network to deter this behavior.

5.1 Countermeasures

1. Users will only be able to request batches of random videos, which will automatically increment view counts to eliminate view farming.
    a. There will be a hard-coded time limit to prevent unreasonable number of batch requests for advertisement or view farming.
    b. Users will only be able to interact with (like, dislike, comment, report, etc.) videos from these batches.
2. Users will need to stake a minimum amount of BitBlock tokens in order to be able to interact with videos.
    a. This minimum stake value must be high enough to prevent creators from creating many accounts to spam-watch their own videos.
    b. The minimum stake value must be low enough to ensure the barrier-to-entry for a typical user is not too high.
    c. The staked tokens must be locked for a certain time period. This is to prevent a greedy creator from staking the tokens, interacting with their own videos, and transferring the tokens to another fake wallet to repeat the process.
3. Users will have a daily accrual of watched advertisements.
4. Users will have a weekly accrual of interactions.
5. Users will have a weekly accrual of uploads.

5.2 Bad Actor Scenario

One might naively assume that the first countermeasure, only allowing interactions with videos that were essentially randomly generated, would be enough to prevent creators from farming views and likes. However, imagine the following scenario: A dishonest creator could create many fake user wallets, all request a batch of videos, and then when some inevitably receive the creator's video(s), send a Like transaction. This would boost the videos like/dislike ratio and like/view ratio, thus boosting its probability of being viewed, thus boosting the creator's

advertising revenue. The only viable solution, unfortunately, lies in requiring staked funds to participate in the network.

As a result of the above measures, let's envision a greedy creator who wants to farm fake views. Let's say the minimum stake value is the equivalent of $5, the lock-up period is 30 days, and the the user accrues 5 interactions per week.

In order to increase profit, the greedy creator decides to create 1,000 fake wallets for $5,000 total stake. To be conservative, let's even assume that every fake wallet is able to View every video on the entire network (thereby ensuring viewing and Liking their own videos). After 30 days, the greedy creator could withdraw the staked funds and repeat the process with 1,000 newly created fake wallets. Thus, they would need to stake $5,000 for 5,000 likes per week.

5.3 Parameter Values

It would be difficult to automatically decide parameters dynamically because this hypothetical algorithm would need to rely on usage metrics, which we've already established can be faked to a certain degree.

In the early days of the network, when a somewhat small number of faked Likes is significant, the chosen parameters should be more aggressive. To avoid a high barrier to entry, the minimum stake and lock-up period should be kept small. Listed below are some proposed initial parameter values.

1. Batch request time limit: TBD, depends on batch size
2. Staking minimum: $5.00
3. Staking lock-up period: 30 days
4. Daily advertisement accrual: 5
5. Weekly interaction accrual: 2
6. Weekly upload accrual: 1

As the number of videos on the platform increases, the chances of randomly viewing one's own video decreases. Thus, the parameters should be relaxed over time, allowing more interactions and less minimum stake value/time.