What to know before you regulate crypto

Charlie Marketplace

Before 'regulating crypto', it is critical to understand each layer of the blockchain stack and the unique nuanced differences in information, privileges, and ultimately choices available to participants in each layer of the stack.

No specific regulation is proposed here. Instead, each layer of the stack is explained, separating out information asymmetries, related privileges, and ultimately the choices to be made into defined schedules. These schedules allow for grouping of participants at different layers of the stack which can then form the foundation of more nuanced regulation that incentivizes desirable choices over undesirable ones as defined by relevant regulators and their goals for public access, fairness, and use of this technology and its related protocols and products.

Layer 1s: Computation, Storage, Read Access

Layer 1 blockchains are generally comprised of: [**computation environment**] [**storage**] [**read access**] which may or may not be combined. For example, the Ethereum blockchain stores a history of 'state transitions' (historical updates to the ledger), which result from transactions bundled into blocks. Transactions can be

simple:

1. Lower my ETH balance

2. raise Bob's ETH balance

or complex:

```
1. Call the ETH_PRICE function from this specific Chainlink Oracle contract
IF 1 ETH >= $1500
  2. Call the TRADE function from this specific Uniswap Exchange contract using the following parameters: [I] [SELL] [5] [ETH] for at least [7,500] [USDC] if I don't get at least that amount, revert the trade.
  ELSE IF 1 ETH < $1500.
  Do nothing.
```

In either instance computation occurs (here, at a minimum, is the number bigger or smaller than 1500) and if the trade happens storage of the transaction. For transactions to be included in blocks, a **'node**' must propose its inclusion into a block and that block must reach consensus with other nodes, after some time period (known as time to finality), the block is 'final' and the transaction(s) is forever included in the history of the ledger.

Computer software, often called 'clients', perform the variety of tasks related to computation, storage, and access of these blocks of transactions. The (human) operators of these clients maintain uptime and receive income for performing their tasks correctly. Because operators can maintain multiple copies of the same client in parallel (and in fact may be forced to do so depending on the design of the blockchain), each copy is typically called a 'node'. Often, blockchain specific phrases like 'miner' (e.g., Bitcoin proof of work consensus nodes) or 'validator' (e.g., Ethereum proof of stake consensus nodes) are used.

Node Privilege

A variety of privileges arise from the power nodes have relative to the users of the blockchain (i.e., submitters of transactions). Some include:

- Knowing about transactions that are available to be placed into future blocks; often called 'transactions in the mempool' (memory pool).

- Choosing which transactions to bundle into a block they add to the chain.

- Choosing the order of transactions within a block they add to the chain.

- Inserting their own transactions strategically into the blocks they add to the chain.

Due literally to the laws of physics, different nodes will have different information about the mempool in any specific instant. Transactions are submitted to *a node* and that node then propagates that available transaction to other nodes in a peer-to-peer fashion. That information is subject to the speed of internet access and even with advanced fiber optic cables and satellite internet the speed of light.

When a node gets a turn to make a block, it does so with only the information it has available. If blocks are made at a high speed (e.g., every few seconds) it is possible available transactions take multiple blocks to even be considered for inclusion because the node making the block doesn't know about it!

This adds another privilege and a caveat:

- Users can strategically submit transactions to certain nodes

- Nodes can strategically *not* propagate transactions to other nodes.

This is often referred to as a "private mempool". Inversely, a node can *censor* transactions from certain users and refuse to incorporate them into blocks they build or even refuse (or delay) to propagate them to other nodes that *would* include them.

The Trilemma

Relatedly, if it is desirable for all available transactions to actually be available to every node before blocks get made, you must do 1 of 2 things: reduce the number of nodes to reduce communication distance or expand the time between blocks to give transactions more time to propagate.

Both of these options affect what is called the 'trilemma': Decentralization, Security, Scale.

Decentralization is the idea that more nodes is generally good because it reduces the benefits of the privilege nodes have, reduces censorship risks, and increases competition among nodes (spreading block proposer powers more thinly).

Security is the idea that doing bad things (faking transactions, splintering the blockchain, spamming the blockchain etc.) should be expensive and/or punished. Relatedly, that there should be redundancies in the block creation process, e.g., multiple different software 'clients' should be available so that a problem with one client doesn't result in blocks no longer being added to the blockchain.

Scale is the idea that more capacity for transactions in both number and speed to finality are good because it improves the user's experience and the market for blockchain.

Proposed Schedule

Schedule 1: A blockchain with few nodes, one or few clients, high censorship risk, and low transparency in available transactions.
Schedule 2: A blockchain with many nodes, one or few clients, low or medium censorship risk, and low transparency in available transactions.

Schedule 3: A blockchain with many nodes, multiple clients, low censorship risk, and high transparency in available transactions.

Selling the right to make blocks

It is possible nodes outsource their block creation to a market of ‘block proposers’. This is called Proposer-Builder Separation. The key difference between Schedule 2 and Schedule 3 is *access to the mempool by those not running nodes*.

This is important not only in Proposer-Builder Separation (where non-node block proposers need to see the mempool to make blocks) but also for users. Users that can see available transactions are better able to defend against the information asymmetry that nodes inherently have. A simple example: how much should I be paying for this transaction?

To avoid over-paying for your transaction to be included, it is critical to know *what are other people willing to pay* and then putting yourself in line at a price you find acceptable. If 3 blocks worth of transactions are paying >$5 each; and you are okay with being included in any of the next 25 blocks, then maybe you can save some money and pay less than $5.

Access to Blocks (i.e., RPCs)

As stated previously, to submit a transaction to a blockchain requires (A) you run a node and propagate it yourself to other nodes; or (B) you submit the transaction to a node and they propagate it for you to other nodes.

Submitting a transaction to a blockchain typically requires a remote-procedure-call (RPC). You write your transaction in a pre-specified format (often JavaScript Object Notation, or 'JSON') and send it over the internet's hypertext transfer protocol: more commonly known as 'http' the thing in front of the websites you visit!

For the vast majority of users, they install a 'wallet' software, e.g., MetaMask, often as an extension to an internet browser like Google Chrome. This wallet software will have a 'default RPC [receiver]'. A specific node or network of nodes that receive transactions from the wallet. Often the RPC is associated with the company that made the wallet, e.g., Infura which is owned alongside MetaMask by the software company ConsenSys. This wallet software will allow for custom RPC [receivers] as well. The user can go to the settings menu and put in relevant details that identify a specific node(s) to submit transactions to (including *which* blockchain!).

This is where the previously discussed privileges of nodes (and the defense users have) come into play. Users can choose which nodes to submit transactions to, e.g., purposefully sending them to nodes that don't propagate them (private mempools). Or if a node is censoring them (refusing to propagate their transactions and/or include them in blocks) they can submit their transactions to a different node. Having a diversity of nodes (more Decentralization) especially nodes that run different clients (Security) improves competition, reduces censorship, and increases resiliency of the blockchain network.

Proposed Schedule (Nodes)

Schedule 1: A node receives only RPCs from specific users, i.e., with bias or censorship; and selectively propagates it.

Schedule 2: A node accepts almost all valid RPC sent to it, with rare, prespecified exception, and propagates it in a standardized, consistent way, with rare, prespecified exception.

Schedule 3: A node accepts any valid RPC sent to it, without bias or censorship, and propagates it neutrally to other nodes in a standardized, consistent way.

Proposed Schedule (Block-Builders)

Schedule 1: A node curates its own blocks, inserting its own transactions and/or ordering transactions as it sees fit to maximize its revenue; or otherwise purchases proposed blocks from proposers that maximize revenue which may include censorship and/or transactions not propagated neutrally to the 'public mempool'.

Schedule 2: A node curates its own blocks, inserting transactions and/or ordering transactions as it sees fit to maximize revenue under some to be determined constraint(s) that sufficiently reduces privilege from information asymmetry including but not limited to handling transactions not propagated neutrally to the public mempool; or otherwise purchases proposed blocks from proposers that follow similar to be determined constraints.

Schedule 3: A node that curates its own blocks using a prespecified publicly known neutral formula for inclusion and ordering with little to no inserted transactions that maximize revenue or otherwise profits from information asymmetry or other privileges; or otherwise purchases proposed blocks from proposers that follow similar to be determined constraints and prespecified publicly known neutral formula given the neutrally propagated transactions available to it at the time.

Not all block builders are equal

This is not to imply anything about one Schedule being worse or better than others. A professional organization dealing exclusively in Schedule 1 nodes may warrant regulation that an individual running an open-source block building software that uses Schedule 1 strategies would not warrant. There is real value in keeping choice, profitability- and even private transactions as options in this technology.

Regarding private transactions: imagine if your wallet key was hacked and you wanted to transfer your money safely to a new wallet. If the hacker is running a node and receives your transfer request in the public mempool, they could copy your attempt to save the money and finish the hack. Private transactions *have already been used* to save people from these kinds of incidents by ensuring the recovery transaction is hidden from the hacker!

The point, again, of this paper is to ensure an understanding of how each layer of this technology faces different information symmetries, privileges, and choices. So that any regulation that does come can be nuanced enough to accurately reflect the preferences and incentives regulators want participants to act under (e.g., don't censor individuals arbitrarily) with minimized collateral damage and maximum freedom for participants to choose under what (legal) constraints they can practically comply with and operate under.

Building Transactions (i.e., Front-end websites and wallets)

Of course, users don't routinely format their blockchain transactions in JSON format and send it over HTTP themselves. The wallet creates the JSON using a mix of user inputs and website inputs.

Blockchain products (e.g., the Chainlink Oracle or Uniswap Exchange mentioned in Layer 1s) can have a full spectrum of complexity. Where sending some tokens to you can be as simple as lower my balance and increase your balance; some products can act as decentralized identity; others as financial instruments (e.g., futures and options with strike prices). Users mostly interact with these products using apps and websites.

For example, if I want to trade the Ether token (ETH) for the Circle USD token (USDC) I can go to Uniswap’s website, https://app.uniswap.org/#/swap, and input my desires:

1. Trade [ETH] for [USDC];
2. Amount: [5] ETH

The website will do some calculations and tell me my expected amount of USDC along with a suggested minimum amount to accept.

3. Minimum amount: [min] USDC to accept

Because similar transactions before me in the same block can affect the price I end up paying (this is why nodes have the privilege of ordering transactions), if I don't get the minimum, it can revert and give me back my 5 ETH.

I still pay for the computation though! If you could get out of paying for computation you could spam the blockchain for free.

The website does **not** take my ETH and give me USDC.

It formats my JSON request (including a to address of the Exchange Protocol Contract, to be discussed) and gives that JSON to my wallet software. I can then review the request and confirm that my wallet should send the JSON request (over http) to a node to propagate the transaction until it's included in a block.

I don't need the website at all. I could format the JSON myself using available documentation, reading the code, or using a different unrelated website or tool to format it for me. The website is just helping me communicate correctly.

It is closer to accurate to think of the website as ***spellcheck*** than to think of it as a broker or custodian of my money.

But- where people build habits, hackers see opportunity. If the vast majority of users don't bother to review their JSON formatted request before sending it with their wallet (or worse, the request is just a bunch of code looking stuff few can read accurately) then hackers don't have to hack the wallet.

They don't have to hack the blockchain protocol contracts (to be explained).

They can do the same scams that plague all kinds of websites (and email providers): trick users into going to different (e.g., phishing) websites OR the more complicated "DNS Spoofing" attack where you convince web browsers that your phishing site is the website.

Web2 Companies comingle front-end and backend protocol

Banks, E-Commerce websites, Credit Score agencies, and numerous other professional organizations have cybersecurity employees, consultants, and contractors monitoring their websites and helping their users avoid common scams/hacks. It's why so many banks will force you to input a code you receive via text after successful logins with your password: 2 factor authentication is annoying for users (costing 10,000s of users 1,000s of minutes each day) but it protects the least technical users from many types of scams.

Some very important difference between those companies and blockchain products exists though:

1. Companies maintain usernames, passwords, and sensitive data on their users;

   This makes their own systems highly valuable for hackers separate from users making mistakes. Blockchain products almost never maintain this kind of sensitive data, even if they have a website to help with JSON formatting, and that website may for existing regulatory purposes track IP Addresses, the product code itself generally accepts users as they are with no risky data collection and neutrally process whatever JSON the nodes read and send to their 'smart contracts' (the code behind their products).

2. Companies *monopolize* how their users access their assets;

   I cannot access my Bank of America checking account using my Wells Fargo credentials. They don't talk to each other! My Bank of America money exists entirely under their control (i.e., custody). Blockchain product websites don't control access to user funds. The data (and money) exist on a public, readable, neutral blockchain. They are just helping users send requests to the nodes that then update the chain. I can make a Uniswap transaction request on *many* websites including *no website* if I want to make the JSON myself.

3. Companies *monopolize* user history of public protocols;

My gmail (google email) uses the Simple Mail Transfer Protocol (SMTP) to send my email messages to other people's email inbox addresses. I could use SMTP by myself, but because [**storage**] is not decentralized nobody would have access to my emails except me (if I keep it) and the recipient(s) (if they keep it)! I use gmail because it keeps track of my past emails for me! But this comes at the price of centralization. Yahoo doesn't have access to my past gmail emails. Relatedly, many services (including gmail) are *not neutral*. In an effort to reduce spam (because it costs $0 to spam, see: Security above) they often censor SMTP messages from servers they don't already know.

Crypto separates front-ends and protocol; a fundamental difference

We demand banks be careful with data, because *they collect it*. We demand their websites be safe from cybersecurity incidents because they are *the only way to access our money* that *they hold*. We demand gmail have a disposal policy for email messages we want to purge because *they are our messages*.

We demand they protect their websites because *it is the only way to access the message history* that *they hold*. Blockchain transaction histories are public. It is *our* choice of wallet that sends JSON requests that *we format* to RPCs that *we select*. Those wallets are *not* the only way to access our money- we maintain control of our blockchain accounts (addresses) via the "private key" (a cryptographic password you cannot reset or safely share).

While we may use a website to help format our requests correctly, those websites don't collect the primary history of our requests (they only pass them to our wallet for final confirmation); nor do they *hold* (custody) our money; nor do they monopolize our *access* to protocol.

Again, they (can be) closer to spellcheck than to a bank website.

Proposed Schedule I

Schedule 1: A website or application (including a wallet application), i.e., 'app', that monopolizes how users access their assets including but not limited to taking custody of assets and/or forbidding access to assets outside of itself (e.g., if a product's code only works when a specific website or application and/or an affiliated node adds special information, e.g., a key or password, to the JSON request); and/or retains without the ability to transfer and purge its storage of "private keys" that access assets for which users have property rights.

Schedule 2: An 'app' that, due to important "off-chain" (not on public blockchain storage) information required to accurately/intelligently interact with an associated protocol and/or due to the inherent complexity of an associated protocol, cannot reasonably have the required JSON request for interacting with an associated protocol be formatted by an individual or unaffiliated 3rd party and/or interpreted accurately in a form that is human readable and reviewable by a 'reasonable person' (in the traditional legal sense).

Schedule 3: An 'app' that interfaces with an associated protocol in a way that is easily formatted, readable, and reviewed by a reasonable person and/or in a way a 3rd party could (with some reasonable effort) accurately mimic, including but not limited to any important information required to accurately/intelligently interact with an associated protocol being available "on-chain" and/or there being little to no important information required to accurately/intelligently interact with an associated protocol.

Proposed Schedule II

Schedule 1: Central Exchanges and Custody Apps that act effectively like crypto banks; they hold your assets and/or you can't get your assets without them.

Schedule 2: Websites associated with complex products where important information for accurately/intelligently interacting with said products is not easily accessible elsewhere, for example, if key information is "off-chain" (not on public blockchain storage), or the information is (for whatever reason including technical ones) is not stored/formatted in a way a reasonable person can use for creating their own transaction request; especially in such a way that 3rd parties could not effectively interact with the product, making the website a de facto monopoly on interacting with the product(s).

Schedule 3: A website that may or may not be associated with a (likely, but not necessarily) simple product with little to no important information required to use the product correctly; if important information is required to use the product correctly/intelligently that information is readily available from a variety of sources and/or available directly on the website. The core functionality of the website (formatting JSON requests for users) should be copy-able (with some reasonable effort) even if the website has some unique benefits that they may or may not charge to access or are not easily copied.

Because this section details such a paradigm shift in how we think of ownership and access of digitized assets, some clarifying examples are provided:

Proposed Schedule III – Specific Examples

Robinhood is an app that holds users' funds, including crypto assets, allowing users to buy/sell them. These assets must be accessed via their app using credentials they control. They may or may not in the future release products that do not fall under Schedule 1 (e.g., a "wallet" app that allows transfer and purge of private keys); but it should be clear the main product is Schedule I.

OpenSea is the major NFT (to be discussed) marketplace for Ethereum (and other blockchains). It operates the "Seaport" exchange smart contract (the product code).

To save users money, they use a central database to store items called "transaction signatures". They use this to allow people to list their assets for sale (for other users to buy) without paying the transaction fees to put that listing "on-chain" (public blockchain storage).

Only when an item is sold does the corresponding signature get included in the JSON request for transferring the NFT as a sale (allowing OpenSea to get its 2.5% fee as revenue for coordinating the transaction).

While users retain ownership and access to their NFTs and can transfer them without reliance on OpenSea's website; important information for interacting with the marketplace accurately/intelligently (assets listed for purchase and bids on assets) rely on OpenSea's monopolized database (it is *not* on public blockchain storage).

3rd party 'aggregators' of NFT listings across marketplaces including direct competitors of OpenSea, rely on OpenSea to make important information available to accurately/intelligently price, buy, and sell NFTs. At any time, OpenSea can censor or refuse access to this information.

Uniswap is the major fungible token (to be discussed) exchange for Ethereum (and other blockchains). Everything that happens in Uniswap is permissionless. Anyone can create new token exchange smart contracts between 2 assets (these contracts are called "liquidity pools", different than the mempools mentioned before).

Anyone can find and trade tokens via liquidity pools that Uniswap, the organization, may or may not have created. Any token that meets the prespecified neutral "ERC20" standard for fungible tokens can be added to Uniswap liquidity pools and/or traded through them (assuming liquidity exists in a pool for the token; every sale needs a buyer).

All relevant information, e.g., the price to trade tokens in a liquidity pool, the amount of tokens in the liquidity pool, the expected amount of tokens received from a trade with the pool is available at no cost by asking the smart contract for its price and storage prior to trading.

Numerous aggregators, 3rd parties, and individuals readily organize their own JSON requests to send to their desired nodes asking to interact with these contracts. Uniswap, the organization, does not and specifically wrote the product code (smart contracts) to make it impossible for them to censor these interactions. Accordingly, their product code is duplicated across numerous blockchains by numerous competing organizations and websites for interacting with these contracts are ubiquitous in crypto.

This is the quintessential example of a Schedule 3 'app'. The website is easily copied, has no monopoly on important information for interacting accurately/intelligently with the associated protocol, never takes custody of user assets, nor collects any data on its users.

Some useful but not critically important information it makes available to those interested in providing liquidity to liquidity pools such as histograms of liquidity, fees accrued over the last 24 hours, etc. can all be duplicated using information entirely available "on-chain" (public blockchain storage). It charges no fee to access this information.

Note: in the product code, the organization does retain 1 special privilege. It can turn on the "fee switch" and take a portion of fees accrued to liquidity providers as revenue for having originally created the product. This privilege is not relevant for its website/wallet schedule, but will be discussed in the next section on protocols.

Protocols

The distinction between protocols and websites/applications in the previous section can be difficult to parse. The easiest dividing line is on-chain code. These are the instructions that nodes servicing blockchains store and implement when they include valid JSON formatted RPC transactions in the blocks they add to the blockchain.

Protocols (and really any code) can do the same thing in entirely different ways, and this is important. The *how* determines differences in risks that users face and it should be the primary driving force for what regulations are implemented and ultimately which development choices are incentivized. Some of the key questions to consider when thinking of scheduling (and potentially regulating) crypto and its protocols include:

1. Does the protocol replicate a regulated financial product?

In the US specifically, 'retail investors' (i.e., regular people who are not certified financial planners, wealthy, and/or accredited via an exam) are forbidden from accessing certain financial products. They cannot invest in the earliest rounds of major startups (outside of certain friends and family round exceptions); They cannot participate in certain types of commodities trading, e.g., oil futures without a broker; they cannot take high leverage positions on stocks; they cannot bet against stocks via shorting outside of certain brokerage accounts or heavy rules on the amounts. Offering these risky services to retail investors will land companies in large lawsuits with regulators like the SEC and CFTC.

There are protocols openly offering large amounts of leverage, tokenized commodities, unregulated fundraising, and shorting with little to no oversight over users (remember, many don't collect any data at all on users!). Each protocol is unique and the available law on these products are built in context of information symmetries, custody arrangements, and broker/dealer privileges that may or may not exist in the crypto version!

2. Does the protocol rely on important off-chain information?

Let's take AAVE for example. AAVE is a "pawn shop" protocol: users can deposit assets and earn interest from others borrowing those assets. All loans must be collateralized, so every borrower is also a depositor of something. I deposit ETH, someone deposits USDC. I borrow USDC against my ETH and pay interest (in USDC) to the depositors.

To borrow 1 asset against a different asset requires information: what are they worth? If I deposit $2000 worth of ETH and borrow $1500 worth of USDC; and weeks later the market price of ETH falls to $1000. I have less deposited than I borrowed. How would I repay the loan! To prevent the system from accruing "bad debt" (positions that are underwater on their loans) it's critical to *liquidate* positions (and pay back the loans) before the debt goes bad!

For that to happen, we must have the market prices of ETH and USDC readily available at high frequency (e.g., at least once an hour) and also have a threshold such that if the market prices diverge, we can close the loan in an emergency. AAVE uses Chainlink Oracles for this exact purpose! If Chainlink reports the market price of ETH is below some threshold, my deposit becomes eligible for liquidation. Note: AAVE does not close it for me. AAVE is a decentralized protocol.

Instead, when my deposit becomes eligible for liquidation *anyone* can pay off (some fraction) of my USDC loan and receive (some fraction) of my ETH deposit. I benefit because I don't lose *everything*: part of my loan is paid off and I still have some ETH and owe less USDC than I originally I borrowed; the depositor of USDC (and the AAVE protocol) benefit because they avoid bad debt; the repayor benefits because they receive slightly more ETH (in Chainlink Oracle value terms) than they paid (in USDC terms). It's a transparent decentralized on-chain system.

Anyone can ask Chainlink for the relevant price (Chainlink puts this important off-chain information *on-chain*); anyone can ask the AAVE contracts for the deposits and loans outstanding, including which loans are eligible for liquidation.

3. Does the protocol serve as critical infrastructure for the crypto ecosystem?

Chainlink is not the only Oracle service. Many exist at varying market shares. But the importance of Chainlink’s price feeds to the crypto ecosystem cannot be understated. It brings off-chain information on-chain.

Similarly, *bridges* are protocols on multiple chains that coordinate transfer of value between chains.

For example, if I have USDC on Ethereum and want SOL on Solana, I have several routes to get there. A popular route is to use a Central Exchange: move my USDC to Coinbase, trade it for SOL, and then withdraw the SOL to my Solana wallet address.

Another route that doesn't use a Central Exchange would be to go to Portalbridge.com/#/transfer - a website that assists users in accessing the Wormhole protocol for cross-chain exchange of tokens. I could trade USDC on Ethereum for SOL on Solana by going through the tokens that Wormhole makes available on both chains (often a synthetic token they control, like solETH backed by Ethereum’s ETH).

The tokens amassed in the bridge contract on their original chains are extremely valuable to hackers and $100Ms of hacks on bridges have occurred taking funds from users investing in bridges.

4. How much value is in the protocol and are risks disclosed?

Code that runs autonomously is powerful. You can trust smart contracts to do exactly what they are coded to do and ignore invalid requests (e.g., give me Bob's money!). But with power comes responsibility. If a contract responsible for $100,000,000+ of value across 10,000s of users can be tricked into giving its balance to a hacker - we want to know before that happens so users can balance the risks and rewards available! Even protocols that look similar (e.g., "pawn shop" protocol) can get to the same result (lending and borrowing) in different ways. The *how* can determine what "attack vectors" are possible which can inform how we scope risk and introduce safeguards to prevent hacks.

For example, let's say AAVE allows deposits of a low value token: a hypothetical garbage token called GRBG. Let’s say, today that token is worth $0.01 each. You would need a lot of it to borrow $10M of USDC from AAVE depositors. We know AAVE uses Chainlink oracles to compare value across different tokens. That makes Chainlink an attack vector for AAVE. Trick Chainlink and by definition you trick AAVE.

Are users aware of this? Are they aware that if someone borrows their USDC they could end up holding GRBG instead if there’s bad debt? Has the AAVE contract code been audited by a reputable smart contract auditing company (they exist!)? What did the audit report? Is it available for users to read? Is it written such that a reasonable person can understand it? Does the AAVE documentation disclose their critical reliance on this 3rd party?

If Chainlink relies on a single Uniswap liquidity pool to report the price of GRBG - that means anyone can change the price of GRBG by buying it! Let's say I spend $1,000,000 buying GRBG. I don't get 100,000,000 GRBG because as I buy one, the price of the next one goes up.

Let's say I get 10,000,000 GRBG (an *average* price of $0.10 each) but the 10,000,000th GRBG cost me $0.25 (recall first one cost me $0.01). Assuming nobody sells GRBG (which would reduce the price), Chainlink will report the current price 1 GRBG = $0.25. I turned $1M into $2.5M "worth" of GRBG. I can't sell this for $2.5M though! Because as I sell the price falls.

Well, I could go to AAVE, deposit the $2.5M ‘worth’ of GRBG and borrow $2M of USDC, and then never pay my debt ever. I just turned $1M into $2M, leaving AAVE's USDC depositors with $1M in bad debt. This type of exploit is called an "oracle attack" and has happened to several pawn shop protocols!

Of course, AAVE is the preeminent lending/borrowing platform in crypto today. It has a variety of controls to prevent this risk including: borrowing caps (only allowing me to borrow $100,000 worth of USDC since the token is low value and the oracle is limited); price smoothing (not trusting large changes in Chainlink's price in a short amount of time); among other controls.

Protocols similar to AAVE using less robust oracles, allowing lower quality deposits (AAVE is strict with the assets it allows to be deposited), etc. have been exploited for $10,000,000s+. Would a reasonable person have understood the quality of deposits allowed? Were the oracles used disclosed?

Generally, contracts responsible for $100,000,000 deserve more scrutiny than contracts responsible for $10,000. Any regulation should be prespecified, fair, and proportional to the relevant risks.

Some information is already nearly universally practiced in crypto today: disclosure of audits and data on the current amounts at risk. More standardization of audits and ensuring their readability would be a great first step regulation that doesn't need to forbid innovation in this sector nor impose specifics on *how* protocols function.

5. Can the protocol be *changed* without user input or awareness?

So far, you may have the impression that code launched onto a blockchain is immutable. This is common but not always true. Blockchains with smart contract capabilities often have a *Turing Complete* programming language to coincide with them.

This means that if it can be coded on a non-blockchain computer, it is possible to code on a blockchain computer, e.g., the Ethereum Virtual Machine (EVM) computation environment has the Turing Complete solidity programming language.

This means you can write a contract that says "be the same as [this contract] over here" and then in the future change what [this contract] is. These are called Upgradeable Proxy Contracts. Your protocol uses a contract(s) that itself relies on other contract(s) and its reliance can be changed.

This is a specific design choice. It's entirely possible to have a contract that can't change. A contract that doesn't have any roles ("owner", "manager", "admin", etc.) with the prespecified ability to call any functions that change anything in [storage] that the code relies on. As mentioned before - Uniswap has only 1 thing that can change: The fee switch. It is currently off. There is a role, the admin, that has the power to turn it on. But otherwise, it’s coded to make it impossible for the admin to move user's money.

When an organization or website or ‘app’ or community of users wants to do something different and the associated protocol cannot be changed, this is called a ‘migration’. They will launch a new contract(s) with new features and user funds will stay in the original (e.g., Uni v1) until they are alerted of the migration and decide whether to move to the new one (e.g., Uni v2, or as of March 2021, Uni v3).

AAVE would be considered upgradeable. There is a defined process (called AAVE Improvement Proposals) for doing things like changing borrow caps or adding new tokens as allowed to be deposited and/or borrowed against. This will be discussed more in the DAOs section.

Who can change protocols and *how* they can change them are important. When Facebook decided to change its business to invest heavily in virtual reality & metaverse, it reported it to the board of directors and their stockholders, but the decision was ultimately the CEO's to make. Just like Facebook doesn't have a record of every stockholder nor their contact information they can't just email everyone to ask for permission. There is a system in place for delegates of stockholders to stay informed and provide feedback prior to certain decisions being made.

Protocols can use tools like Discord, Twitter, Telegram and websites to report updates and ask for feedback; they can use tools like Snapshot for 0 cost off-chain voting or launch contracts that allow on-chain voting to give control (or semblance of) to their users and/or token holders. This will be discussed more in the Fungible Tokens section.

Ultimately, the power to change how things work is the power to break things and is inherently risky. Are these powers disclosed? Do users know how to make their voice heard? Would a reasonable person be able to use the protocol and/or withdraw funds if the most popular website for formatting JSON requests to an associated protocol stopped functioning? Has the new contract(s) been audited? Is there new reliance on 3rd parties or known attack vectors? Are they disclosed?

Proposed Schedule

Schedule 1: Protocols comprised of smart contract(s): replicating complex and/or potentially regulated financial products that require important information not available via public blockchain storage for a reasonable person to use accurately/intelligently and/or relying on (and/or transmitting) off-chain information via/like Oracle services; and/or hold a to-be-determined significant amount of value within them including, but not limited to, the ability to move that value and/or significantly alter the functioning of said contracts without the notice and/or consent of users.

Schedule 2: Protocols comprised of smart contract(s) relying on little to no important information not available via public blockchain storage such that interacting with said contract(s) can be done by a reasonable person; and/or with any potential changes to the contracts not significantly altering the functioning of said contract or their risk profiles without the notice and/or consent of users, including but not limited to upgradeable proxy contracts.

Schedule 3: Protocols comprised of immutable smart contracts with no reliance on important information not available via public blockchain storage such that interacting with said contract(s) can be done by a reasonable person with no significant changes to the functioning of said contract possible.

Fungible Tokens

Fungible tokens, often referred to by the standard template they are built on, e.g., ERC20 (Ethereum Request for Comment #20), are tokens that are indistinguishable from one another in the same grouping. There is no ETH # 5 or ETH # 3250. There are just 1,000,000s of ETH and they are all equivalent.

In addition, ETH can be broken into smaller units as needed up to 18 decimals (where the smallest denomination is called 'wei'). Other tokens can have different amounts of decimals, the core functionality is that they are not unique. Another core functionality is that they are permissionlessly transferable (I can reduce my balance and increase your balance without asking anyone for permission- as long as I send a valid JSON request to a node and the transaction is added to a block).

These tokens can have a variety of relevant traits that overlap with some of the concerns, disclosures, audits, risks detailed in previous sections and not duplicated here.

They can act as a unit of account, representing ownership (right to property) of something that may or may not be recorded on public blockchain storage.

- They can act as a medium of exchange, by being transferable, you can transfer them to someone else (or more technically, decrease your balance in the token contract's storage and increase someone else's balance in the token contract's storage) in exchange they transfer you a different token. The someone else can be a contract, e.g., liquidity pool contracts.
- They can have direct use as a payment for services or other related utility that does not result in you receiving something in return "on-chain".
- They can have custom functionality that is triggered upon transfer or other function added or modified from the template they are built on.

Examples can be illuminating here.

Token Examples

The **ETH** token is used to pay for computation and (change of) storage on the Ethereum blockchain. The market price of ETH (e.g., via a Chainlink oracle) is irrelevant for this use case. The price for computation is determined by a free market of users submitting their transactions to nodes in mempools with a willingness to pay per unit of computation.

Each unit of computation is called 'gas'; generally speaking, the more ETH you pay per unit of gas the sooner you can expect your transaction to be included in a block because nodes are self interested in profiting. For example: adding 2 numbers requires 3 units of computation ("gas").

If I submit a transaction interacting with a calculator contract with the parameters: {3, '+', 5} I include in my submission how much ETH I am willing to per per gas (here, I'll need 3 gas units). If I bid 0.0001 ETH per gas, I would pay 0.0003 ETH to get the value [8] returned. If the market is pricing gas at higher than 0.0001 each, I would need to wait until the market price falls before I could expect a node to include my transaction in a block (because other available transactions in the mempool are more profitable!).

The **LINK** token is used to pay for Chainlink Oracle services. It is a modified ERC20; more specifically ERC677. This allows it to include data inside its transfer() function allowing recipients to know exactly which Oracle service is being requested directly inside the payment for said service. This is very efficient compared to separating payments and the receipts like we normally experience.

The **xSUSHI** token is used to accrue revenue from the Sushiswap protocol. Holders of the SUSHI token "stake" (deposit into a staking contract) their SUSHI in exchange for xSUSHI (this is reversible). Sushiswap liquidity pools charge 0.3% for trades on their decentralized exchange.

Most, 0.25%, goes to those providing liquidity between the pairs of tokens. If this reminds you of Uniswap above it is because Sushiswap is one of many copies of the Uniswap code base!, while 0.05% goes to those holding xSUSHI.

Effectively, the 0.05% accrues in dozens (if not 100s) of tokens throughout the day, and then they are all traded for SUSHI tokens and given (proportionally) to xSUSHI holders. This is similar to dividends via stock buy-backs, but designed to be significantly more decentralized and automated than traditional corporate dividends because the entire value flow occurs on-chain!

There are 1,000s of tokens across 10s of blockchains - in addition - the social layer comprised of the community of users, early developers, supporting/associated companies, etc. have a spectrum of control over how the tokens function and accrue value.

Proposed Schedule

Schedule 1: Tokens with or without a defined supply that act as complex and/or potentially regulated financial products that require important information not available via public blockchain storage for a reasonable person to accurately/intelligently use or hold, including but not limited to those representing rights to external and/or "off-chain" property requiring the correct performance of an owner, manager, admin, or other intermediary for delivery of said property or value flow.

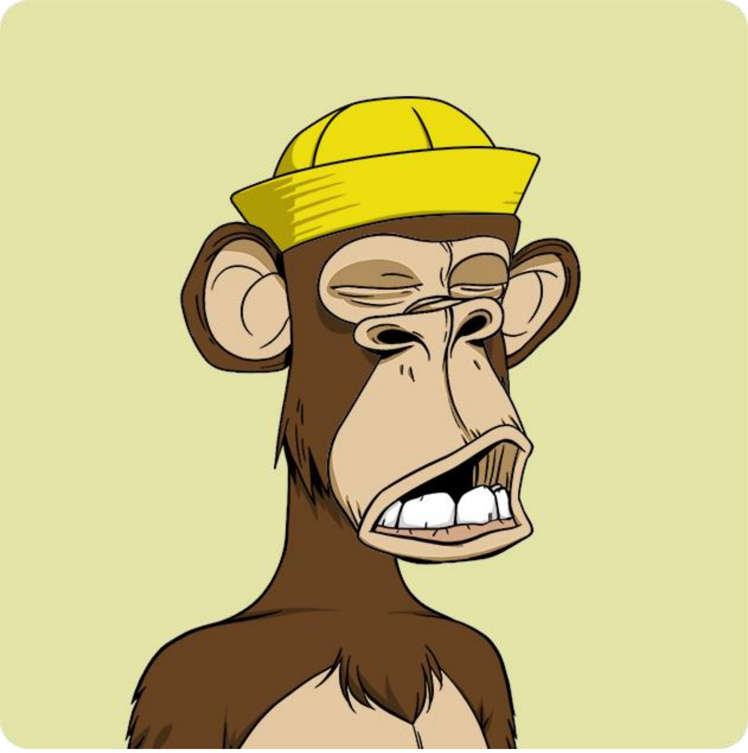
Schedule 2: Tokens with or without a defined supply available and verifiable via public blockchain storage; with use outside the crypto ecosystem and/or representing rights to an external and/or "off-chain" property requiring the correct performance of an owner, manager, admin, or other intermediary for delivery of said property or value flow.

Schedule 3: Tokens with a defined supply (including rules for supply changes if applicable) available and verifiable via public blockchain storage with direct use within the crypto ecosystem and not representing rights to external and/or "off-chain" property requiring the correct performance of an owner, manager, admin, or other intermediary for delivery; including but not limited to those representing rights to internal or "on-chain" property with important information available and verifiable via public blockchain storage for a reasonable person to accurately/intelligently hold said token and/or use said token to receive the expected property or value flow.

Non-Fungible Tokens

Non-Fungible Tokens, or "NFTs", often referred to by the standard template they are built on, e.g., ERC721 or ERC1155, are tokens *not* interchangeable within a grouping. Each is numbered, such that you don't own 'Bored Ape Yacht Club', you own id # 5,300 of Bored Ape Yacht Club. Each token is identified by its id number. In addition, NFTs can (but are not required to) include metadata such as images that can increase the differentiation between id numbers. Bored Ape Yacht Club ('BAYC') #5300 has the following metadata:

{"image":"ipfs://QmNdBa5TqGbEahNNWUE6TLtcxSUPHzLaX2PwPa9HD1pMV5","attributes":[{"trait_type":"Hat","value":"Fisherman's Hat"},{"trait_type":"Mouth","value":"Dumbfounded"},{"trait_type":"Fur","value":"Dark Brown"},{"trait_type":"Eyes","value":"Closed"},{"trait_type":"Background","value":"Yellow"}]}



This is JSON format and it relates to the image associated with BAYC #5300: an ape with dark brown fur, eyes closed, and a yellow fisherman's hat (among other characteristics).

What's so special about this picture and why are people paying $10,000s of dollars for pictures like this? Can't you just download the image for free?

The answer connects back to [**read access**] and [**storage**] that blockchains provide. The NFT token contract itself has an id: its smart contract address. The official Bored Ape Yacht Club NFT token contract address on the Ethereum Blockchain is:

0xbc4ca0eda7647a8ab7c2061c2e118a18a936f13d.

100s if not 1,000s of computers running the Ethereum software in parallel (and 10,000s of Ethereum users!) all agree that this address and this address alone is the true BAYC address.

Any NFT within this contract is legit. Any another NFT claiming to be Bored Ape Yacht Club, even with the same exact metadata and/or image, is a fraud because it has a different address.

There are more crypto addresses than there are atoms in the universe, so replicating an exact address is mathematically impossible even with trillions of computers brute force guessing and checking for trillions of years (and no- quantum won't help!).

This is not too different than how we number the many people with the name John Smith in the US via social security numbers. A blockchain address identifies an NFT just like a social security number identifies us. A driver's license with the same name isn't enough to drain another John's bank account.

This is actually quite powerful. Blockchains have devised a mechanism to enforce digital scarcity - a prerequisite to truly digital property.

Instead of providing my first name, last name, email, phone number, birthday, and security questions to create accounts with every e-commerce brand I make 1 purchase from I can have a scarce and easy to read digital identification to access websites and make purchases with.

The Ethereum Name Service NFTs do exactly this- allowing people to assign human readable lookup names for their addresses. For example.

0x98D33aeBa642151d5fb53E7966d4A0121093e478 is registered to **johnsmith.eth** via Ethereum Name Service.

Similarly, because you can easily read the blockchain and ask who exactly owns which *real* (i.e., correct address) Bored Ape Yacht Club NFTs, you can provide utility (e.g., business opportunities, merchandise discounts, access to events) to only that exclusive group of individuals!

So why do people buy NFTs? Because they know which ones are real!

The information needed to get prestige and/or utility, and thus which NFTs are worth owning as digital property are available via public blockchain storage for free!

All the risks detailed in the Tokens section also apply to Non-Fungible Tokens. Throughout 2020-2022, NFTs were used to extract $100,000,000s from naive "investors" in crypto. Organizations promised profit to unsophisticated buyers who trusted individuals to correctly manage, promote, administer, and develop products related to the NFT (e.g., video games, clothing lines, TV shows, movies).

All the buyers had to do was 'mint' the NFT (be the first purchaser of an id number) and wait for the price to go up- the organization would do all the work for them… Unfortunately many organizations took the money and ran, colloquially called “*rugging*”.

Proposed Schedule

Schedule 1: Tokens with or without a defined supply and/or set of characteristics available and verifiable via public blockchain storage that act as complex and/or potentially regulated financial products that require important information not available via public blockchain storage for a reasonable person to accurately/intelligently use or hold, including but not limited to those representing rights to external and/or "off-chain" property or future profit requiring the correct performance of an owner, manager, admin, or other intermediary for delivery of said property or value flow.

Schedule 2: Tokens with or without a defined supply and/or set of characteristics available and verifiable via public blockchain storage; with use outside the crypto ecosystem and/or representing rights to an external and/or "off-chain" property or future profit requiring the correct performance of an owner, manager, admin, or other intermediary for delivery of said property or value flow.

Schedule 3: Tokens with a defined supply and characteristics (including rules for supply or characteristic changes if applicable) available and verifiable via public blockchain storage; including but not limited to those tokens for which purchases were premised on utility requiring the correct performance of an owner, manager, admin, or other intermediary for delivery of said utility; including but not limited to those provided "as-is" as digital collectibles with no premised utility requiring the correct performance of an owner, manager, admin or other intermediary for delivery of said utility.

Some NFTs are gateways to products or experiences and the purchaser should get what they paid for. Others are digital collectibles and changes in price don't make them financial products- like fine art.

Some NFTs are receipts from crowdfunding where it is marketed and/or understood that purchasing the NFT will result in future profit requiring some manager/admin to fulfill their promise (e.g., create the video game where the character NFT will be playable). These likely intersect existing securities, fundraising, and/or similar regulation especially when the manager/admin (or DAO, to be discussed) retains key rights to the property (e.g., copyright on images), control over raised funds, and/or operation over key assets (e.g., websites, coordination tools).

DAOs: The Social Layer: Capacity, Culture, History

Decentralized Autonomous Organizations, or "DAOs", are groups of individuals reaching some form of consensus such that the 'group' acts separate (or in representation) of the individuals as it relates to crypto decisions and products in some less than centralized way. Whether these groups are unincorporated associations (members have unlimited liability for the group's debts and possibly crimes) or limited liability companies ("LLCs") using fancy new tech as part of their operating agreement or some other corporate form is to be determined. A common joke is that DAOs are groupchats with a treasury. Some might fall within the SEC rules on Investment Clubs. Others might be doing unregistered securities offerings. Others might limit memberships to known accredited investors. DAOs is as general a word as "Crypto Company".

In practice today, we generally see DAOs overlap in the following ways:

- A group of individuals with crypto assets communicate / form a community using tools like: Discord, Telegram, Twitter, etc.
- A crypto address exists, owned by the DAO 'entity' (often "multi-signature wallets", e.g., Gnosis Safes), holds (and/or manages) on-chain assets (e.g., NFTs).
- The individuals may or may not have property rights to the assets held (and/or managed) by the entity.
- The individuals come to consensus on how the DAO entity should operate (comms tools to use, authorized signatories on the "multi-sig") and use its assets (e.g., transfer, buy, sell, pay contractors, invest, give grants) often using "governance" tokens to represent voting power (more tokens = more votes = more power). The tokens can be NFTs or fungible ERC20 or other token standard (including not-transferable so not sellable tokens!). The point is voting power is on-chain and publicly viewable.
- The individuals may or may not be compensated for their participation in the DAO (including voting, community leadership, investing (or 'staking') in DAO products, providing liquidity to the DAO's token(s), etc.).

DAOs often serve as the social counterpart to blockchain Layer 1s. There's [capacity][culture][history] in the place of Layer 1's [computation][storage][read access].

DAOs, like companies, can have a wide range of membership counts, talent, efficiency, and goals. The ability to reach their goals (for example, MakerDAO's goal to evangelize the decentralized stablecoin Dai) is their **capacity**. As any group of individuals, DAOs develop a distinct **culture** that influences what they consider goals, (self) perception of the DAO's capacity, and intragroup conflict (and its resolution). For DAOs to persist members must flow in and out continuing **history** with new knowledge and market information relevant to measuring the DAO's capacity and documenting its culture.

The Ship of Theseus applies here: Is it the same DAO if every single member is replaced? Documentation on itself allows for a collective history that directly influences capacity and culture.

A most fundamental one: history of proposals and votes. Does the DAO have enough capacity to complete the tasks assigned to it? Well, what tasks were assigned and how long ago? A more practical one: How does the front-end website domain get paid for each year? Does someone have the password to the DAO Twitter?

Even if a DAO doesn't tokenize its votes or distribute profits. When people make the decision to invest their *time* in a DAO they have to assess capacity. Can this DAO reach its goals? To consider this they need history. The best place for this history to live is arguably... on-chain!

What better place to put critical information over time than the blockchain nodes? There are 100s of decentralized redundant copies of the ledger all over the world and its free to read on publicly access storage.

Well, some DAOs do this and some blockchains even have nodes vote 'on-chain' to preserve the history of good and bad nodes. But this means governance of the DAO competes directly with numerous blockchain transactions for inclusion in blocks. This can be expensive! Paying to vote! Poll taxes are so bad for participation that they're illegal to apply to real world voters in many countries.

One common workaround on Ethereum is Snapshot. Snapshot is signature-based DAO voting tool. It coordinates with long term storage solutions (e.g., Arweave and IPFS) to store proposals and vote histories. The voter in Snapshot uses their wallet to sign their vote with a cryptographic proof of their voting power (e.g., # of tokens at a timestamp) without paying for an on-chain transaction (which requires gas).

Other options include polls on twitter or discord. Another is to use a delegate model so that delegates represent the voting power of individuals backing them. Regardless, access to history is important. We'll leave out a discussion of culture and just say that it matters to people, it affects commitment to leaving history behind, and can affect perceptions of capacity too.

Proposed Schedule

Schedule 1: DAOs operating and/or managing a protocol and/or treasury, including those where individuals in the DAO have property rights to assets custodied by the DAO, that replicate and/or deliver individual property to complex and/or potentially regulated financial products; and/or those required to meet a to be determined documentation and/or reporting requirement on asset ownership, updates, disclosures, balances, goals, proposals, decision history/storage, vote results, delegation, governance, crypto address, authorized signers, etc.; including those that have tokenized property rights to assets custodied; which may or may not have tokenized governance and/or voting power,

Schedule 2: DAOs operating and/or managing a protocol and/or treasury, not including those where individuals in the DAO have property rights to assets custodied by the DAO; where governance or other associated token are used to incentivize individual investment in associated strategies (without DAO custody) and operate a de facto monpolized front-end as detailed in Website Schedule 2.

Schedule 3: DAOs operating and/or managing a protocol and/or treasury, not including those where individuals in the DAO have property rights to assets custodied by the DAO; which may or may not include governance or other associated token used to incentivize individual investment in associated strategies (without DAO custody); which may include managing a front-end as detailed in Website Schedule 3.

The lines are blurry

The lines between DAOs, Protocols, and Front-Ends is messy. Users may not understand that the website they use to access a DAO incentivized investment opportunity is not the same as the on-chain protocol.

The website could go down and while their assets may not be custodied (i.e., they're still retrievable with the right JSON) no website is available to help them spellcheck the JSON and submit the transaction they need to pull their money out.

When judging DAOs it's critical to contextualize the goal, the sophistication of the participants, the honesty of the marketing, the availability of history, accessibility of key information, and the power the DAO holds over an associated protocol.

Can the protocol be *changed* without user input or awareness? How much value does the protocol hold and at what risk is that value; is that risk disclosed in a way a reasonable person would understand and what efforts have been made to minimize that risk?