Current Limitations Of Crypto Gaming, Abridged

Note: This is a slightly abridged version; the full version will be released at a later date, but this still hits all the main points!

Intro

Prevailing sentiment is that current crypto games are subpar compared to their centralized alternatives because of issues surrounding developer talent and, though less so, regulations. “Just wait until the big gaming studios come into crypto”, “No major game developer will incorporate crypto without regulatory clarity”, and “No one is really playing these play-to-earn games for fun” are all phrases that have been echoed in recent months. There *is* some truth to these; powerhouse game studios can usually out-engineer small teams of developers, a lack of regulatory clarity can deter traditional developers from incorporating crypto into their games, and a not-insignificant portion of play-to-earn gamers would enjoy Halo over DeFi Kingdoms. That being said, the reality is that lower developer firepower and unclear regulations are only part of the reason why current crypto games can’t hold a candle, gameplay-wise, to traditional games.

Surface-level, what might a utopian version of play-to-earn look like? Imagine Rocket League, but with the addition of “cars as tradable NFTs” and tournaments where you can pay tokens to enter and earn tokens if you win. Maybe you can even bet on teams as tournaments are on-going and livestream the games. Imagine Borderlands, but with a marketplace for selling items and allowing high-powered players to take lower-leveled players on boss raids for loot. Imagine Fortnite, or any other game you enjoy, with similar mechanics, or an MMO with a bustling crypto-based economy. To top it off, imagine all of this is decentralized (players custody their own items) and provably fair (impossible to cheat). I’m sure many gamers would be chomping at the bit to participate in this hypothetical future of gaming.

This future of decentralized gaming (or, “the metaverse”) is currently constrained more by technology than by regulatory issues or a relative lack of developer talent.

Why Decentralization?

Centralized game creators own everything. Players’ data, items, and freedom to participate can be determined by the game creators. The game’s IP and profits are typically the sole property of the creators. Creators can change any of this at any time, which allows them to alter game mechanics, ban players without a reasonable cause, take away items that players have rightfully earned, or shut the game down completely. There may also be unintended side effects as a result of centralization such as high traffic shutting down servers, easy methods of cheating, or bugs that harm players’ experiences.

These lead us to some reasons why we would want decentralization in games:

- **Censorship resistance:** No central party can determine who can play.

- **24/7 Uptime:** Users and developers are assured that the game will always be there. Scalability is not limited by cloud service costs; games scale with demand.
- **Open Infrastructure:** Many typical costs for infrastructure (databases, servers, etc.) can be cut down by outsourcing them to existing, open technologies, which saves time and money.
- **Ownership:** Users can take full custody of their own items.
- **Interoperability:** Games can incorporate items from other games, and users can use their assets in any ecosystem that supports them.
- **Anti-cheat:** Rules are written directly in code and can be verified.

How Do We Get Decentralized Games?

Let's think about this from first principles. At a high level, a game takes a series of inputs from players, who are utilizing some sort of interface, and produces some outputs. You shoot at an opponent, and the opponent takes damage. The goal is to be able to *implement* and *enforce* these "transition functions" - mappings from inputs to outputs - in a decentralized fashion.

Centralized games typically build most of this from the ground up, though some aspects like game engines, servers, and platforms may be provided through integrations. Creating scalable, enjoyable games in a decentralized fashion is arguably harder, but, thanks to the openness and high level of composability offered by decentralized solutions, even more parts of the tech stack can be outsourced. Some technologies that are available to build such experiences include:

- **Cheap/programmable asset ledgers:** Solana, Polygon, Avalanche, etc.
- **Cross-chain communication**: Layer Zero, IBC, Synapse, Multichain, etc.
- **Oracles:** Chainlink, Pyth, Band, etc.
- **Decentralized storage:** IPFS, Arweave, Ceramic, Filecoin, etc.
- **Decentralized/scalable compute**: Akash, Fluence, Aleph.im, ZK-rollups[1], etc.
- **Interfaces/rendering**: Render Network, The Graph[2], Livepeer, etc.
- **Connections:** Wallets, Pocket Network, Helium, etc.

[1] *Mainly referring to recursive proofs*
[2] *Referring to using subgraphs to scale queries*

Different dGames (decentralized games) and dApps will require a different combination of these. We already have some layers of the stack needed to make this vision happen, though there are some holes to be filled that will be addressed later.

Current Crypto Games

One approach is to encode the entire game, aside from the interface (usually run on centralized cloud servers, but sometimes hosted using a decentralized service such as IPFS) into smart contracts. Current crypto games, including Axie Infinity, DeFi Kingdoms, and Crabada, operate this way. Often, they'll use NFT metadata, randomness, and user inputs ("use this specific attack move") as inputs to their smart contracts to produce some output. Although most crypto games don't publish these contracts on block explorers or Github in order to keep their "secret

sauce" safe and to make things harder on botters, this is really all that's going on. This approach, when transition functions are not highly computationally-burdened or time-dependent, works great for more basic games such as turn-based strategy games. It also works great for DeFi applications and primitives, prediction markets, and the like as transition functions/rules are easier to program for financial assets that only require simple number manipulation.

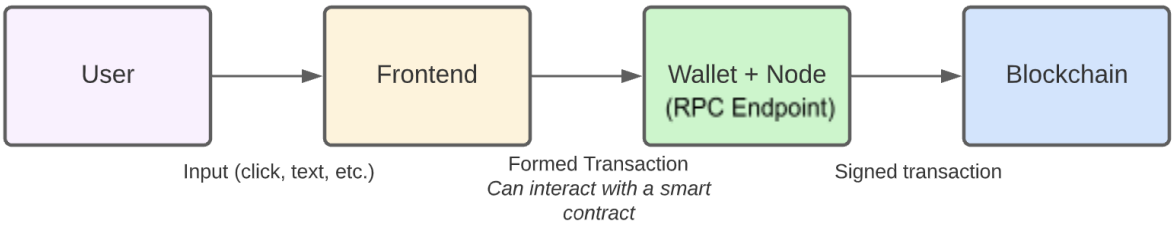

Figure 1: Simple dApp transaction flow

This flow can obviously increase in scope when you consider more complex transactions (for instance, cross-chain transactions or rollup commits), but this is the core flow currently utilized by most crypto games.

Game rules that determine outputs from number representations of NFT traits, randomness, and basic user input are easy to encode into smart contracts, as are AMMs or even more complex math such as the Black-Scholes equation for pricing options. The actual dApp or game interface is simply an abstraction over the underlying smart contracts that dictate the game's rules.

Some current, simpler crypto games have certainly seen success in attracting relatively high amounts of users and economic activity due to novelty and economic incentives.

- Axie Infinity has seen millions of daily active users, DeFi Kingdoms' JEWEL token's market cap reached over $1.3 billion in early January, and other (upcoming) crypto games are raising millions by offering tokens to investors.
- More and more developers are making the switch to crypto gaming, as are other gaming-related professionals like Justin Kan, co-founder of Twitch, who recently co-founded Solana-based NFT marketplace Fractal, which focuses on gaming.
- Some upcoming crypto games such as Aurory and Illuvium look like they'll blow other crypto games out of the water in terms of visuals and depth of experience.
- Gamified economies with real money on the line can be fun games in and of themselves.

However, after playing current crypto games for a while, you'll notice that these games are not very interactive. When you send your character on a quest in DeFi Kingdoms or to mine in Crabada, you don't actually control the NFT doing that task. The NFT just leaves your wallet and comes back later along with some reward. Axie's PvP mode is a bit more interactive, but it still uses simple turn-based transactions. Pokemon Red and Blue were doing this back in 1996; turn-based games are nothing new, though the self-custodied assets involved and decentralized method of enforcing the rules certainly are.

The lack of interactivity comes from the fact that these games are, for the most part, not skill-based. This makes sense; since the entire game flow is encoded directly into smart contracts, all transactions must have deterministic outcomes, computation is limited, and time constraints can be only roughly enforced. The winner of a PvP fight between two NFTs will be determined by the one that has the higher stats, perhaps combined with some randomness and basic player input. This approach does not work for skill-based games that require high levels of (typically very time-sensitive) computation. For instance, Call of Duty's user-controlled gun fights cannot be efficiently encoded into smart contracts.

Despite the growth of current play-to-earn (or "play-and-earn") crypto games, it's become apparent that the "economic gameplay" aspects are attracting users more than any "fun gameplay" aspects are. Plus, with the current setup of having the games entirely encoded into smart contracts, it's possible for mercenary players to turn the game into a "bot-to-earn" arena; Axie Infinity, Aurory, and Crabada have all been working on measures to limit bots. There were over 3 billion active video game players in 2020, and that number is estimated to keep growing. In order to attract *billions* of users to crypto games and create more sustainable in-game economies, we need to improve the gameplay itself. In trying to do that, it appears that current crypto games are running up against a wall in terms of how expressive and dynamic they are. What gives?

Limitations And Solutions

As stated above, using a simple transaction-based architecture with an interface that abstracts away direct contract calls into something more appealing to the eye, we cannot run more complex game mechanics. Imagine playing Skyrim, but every time you wanted to drop an item, you had to pause to send a transaction. Or, to take a more extreme example, imagine playing Fortnite, but every time a bullet makes contact with someone, you have to pause to wait for the damage to be finalized on the blockchain. The *state* that matters, in these cases, is not just your inventory or your weapon/health; it also includes things like your location, the results of your frequent inputs, and the constantly changing state around you (including, for multiplayer games, the state of other players). Highly scalable blockchains such as Solana, ZK-rollups + recursive proofs ("layer 3s"), and app-specific chains allow us to process more transactions at lower costs, but we still can't rely on blockchains to run the much heavier computations needed to verifiably track and update states to the extent that modern games require. What we *can* do is run these more complex computations elsewhere, verify them, and store them on-chain.

At the core of the limitations seen in current crypto games lies the inability to encode more complex computations into on-chain programs. To solve this issue, we need decentralized hosting/computation over longer-lived data that includes robust anti-cheat measures. To quote Kyle Samani in his Infinite Scale article, "Perhaps the most significant limitation is that this model [coordinating activities in a decentralized fashion] is not fully generalizable to support arbitrary computations over persistent data over long periods of time (e.g. running a web server indefinitely)—at least not yet."

What needs to be accomplished is the separation of the more complex, highly time-constrained, state tracking/updating logic from the asset ledger. Computation involved in gun fights, Rocket League-like matches, and other more involved in-game actions needs to happen off-chain. Output from this off-chain computation should be verified, and the applicable pieces should be relayed to the ledger for storage and any additional processing (who won a match, which items a user dropped, etc.). This off-chain computation must also be verifiably sound and it must contain measures to combat cheating. Zero knowledge proofs, more generalizable execution traces, modified data availability sampling (to efficiently search for specific anomalies in large datasets), and economic incentives for reporting bad behavior can all help here.

Of course, it's possible to store user data off-chain using something like Ceramic and then to simply store references on-chain. Some ZK-rollups will, depending on the model, implement a similar process by using off-chain data availability. Regardless of how the data model is set up, more interactive crypto games will require:

(a) Off-chain components to handle more complex logic in a trustworthy/fair manner. This may look like a decentralized network of servers capable of scaling to demand, combined with verifiable proofs and cheat-checking logic to help ensure the game is executing as intended and the players are playing fair. It may look different as well.

(b) Oracles or another form of "relay network" to effectively pass this "outcome data" to wherever it needs to go.

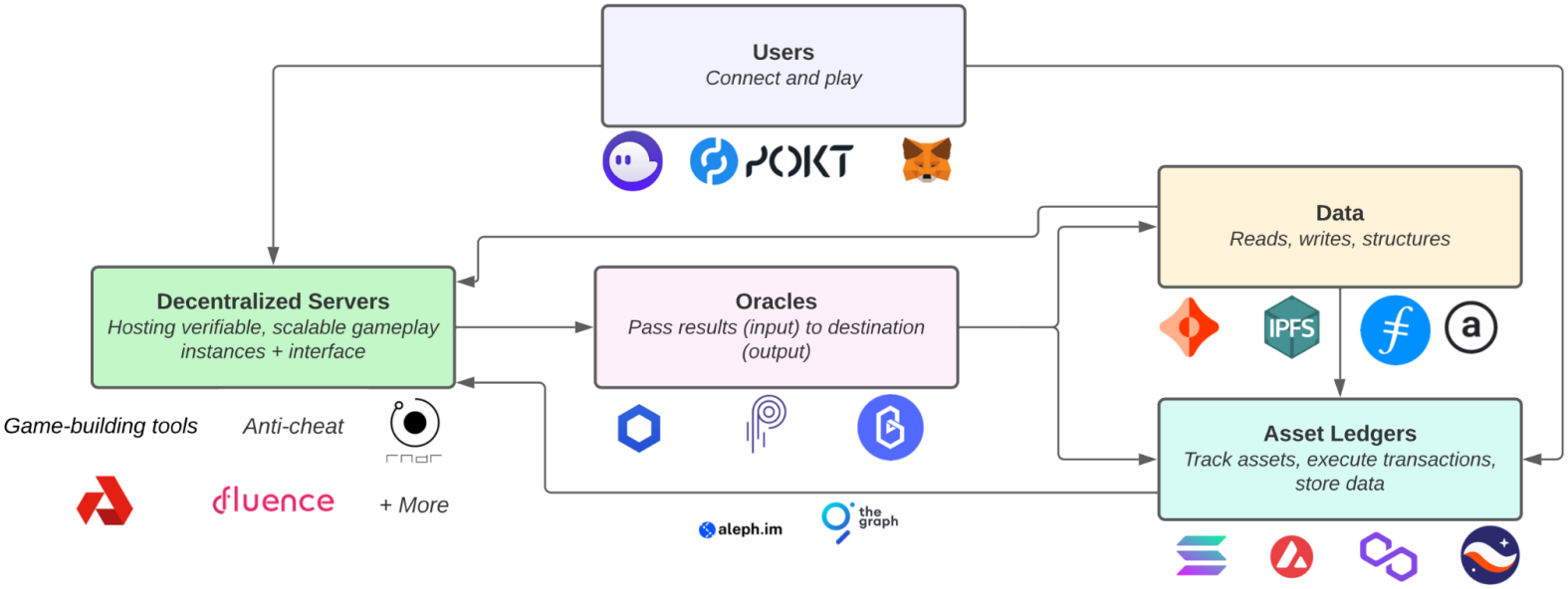

Figure 2: One potential "dGame" architecture (many more are possible)

Although we aren't there yet, it certainly looks like progress is being made in this area. Many of the technologies mentioned in the "*How Do We Get Decentralized Games?*" section, combined with advances in cryptography (more efficient proofs) and constantly improving hardware (Moore's Law), are opening the door for scalable, verifiable, off-chain computing done via decentralized networks to be realized in the future. Economic incentives and verifiable proofs

such as those found in Render Network, Chainlink (2.0), and other off-chain technologies can ensure that these solutions are sustainable and scalable as well.

We could probably build, ignoring some potential UI problems introduced by current wallet and key management solutions, a decentralized Twitter with current technologies. Scalable data storage and the ability to host decentralized static interfaces are already, to an extent, here. To take things even further and build the future of the metaverse, however, we'll need scalable, verifiable off-chain computation with robust anti-cheat measures.

Conclusion

This topic is something that I've been thinking about for a while now. It is apparent that others are as well, given some of Multicoin Capital's writings and tweets such as the one below.

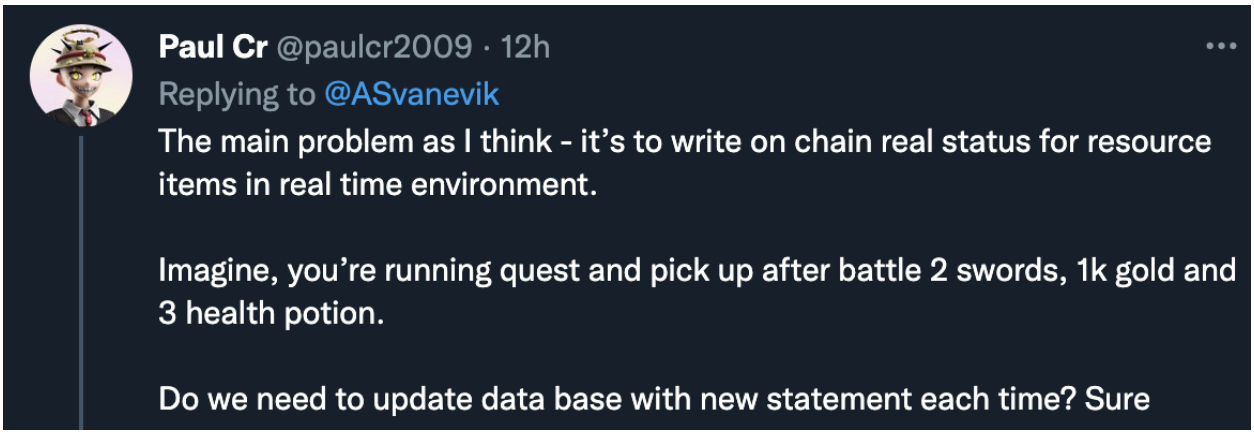

Figure 3: Musings about this issue

The interim solution is likely a combination of (a) games that take the current, simple transaction flow and build rich universes/experiences on top of it, such as Aurory, Illuvium, and DeFi Kingdoms' upcoming multi-chain settings, and (b) games that use centralization to enable more complex gameplay with anti-cheat at the expense of some of the benefits we'd get from decentralization and some probable regulatory issues.

Looking ahead, the potential for a fully decentralized metaverse is certainly a bright one. It will be up to builders creating new technologies and networks, forward-thinking investors backing the builders, users adopting these technologies for their advantages over centralized solutions, and regulators implementing laws that enable this open-source future to prosper to make this vision come to life. Once the building blocks are there and regulators come to support the tremendous advantages and benefits that could be realized as a result, then we can focus on getting the big gaming studios in…but, until then, imagining that we are "just a great development studio away from a great crypto game!" is living in a fantasy land. There are other challenges to solve first.