DARKWINDS

A trading card game on the Ethereum Blockchain

DRAFT

March 31, 2018

1 Introduction

> "I happily played World of Warcraft during 2007-2010, but one day Blizzard removed the damage component from my beloved warlock's Siphon Life spell. I cried myself to sleep, and on that day I realized what horrors centralized services can bring."
>
> —Vitalik Buterin, Creator of Ethereum

Collectible card games have existed in popular demand since decades. After the massification of the Internet, many of them were successfully digitized like *Magic: The Gathering* and *Pokemon* and new ones were created emulating the look and feel of the physical collectibles in form of a multiplayer videogame. However, being of electronic nature, it's physical existence is an entry on a database, and the amount of each card each player owns is a counter inside a central server. Using databases to create trading card games can produce a fun experience at the cost of conceptual problems:

- Only the owner of the software can allow/deny the use of the digital card within it's ecosystem.
- The digital card existence depends on the game systems to be online.
- The game owner can prohibit trading or markets for cards, which is usually enforced.
- The game owner can control the issuance of cards witout transparency to the user.

Thanks to Blockchain technology we can make a trading card game where all cards are stored

in a Ethereum smart contract, where players obtain true ownership of the game object and are free to trade in an outside the game program.

By being a blockchain asset, each card will have the following qualities:

- It's securely stored in an Ethereum wallet.
- The owner and only the owner has perpetual rights for transfer, trade or sale of the token
- It can be read by other applications, like a mobile wallet for ERC721 tokens.
- It's existence doesn't depend on the availability of the game owner systems.

Each card is composed of a serial number and a model id that corresponds to a card model. There can be many cards of the same model, with different serial numbers. Card models are contain metadata for it's appearance (an image file) and it's game action properties (ex: deals 3 damage points to enemy target).

The smart contract also controls immutable rules on the issuance of cards, including the total limit of cards to be issued and the rarity of card models.

2 The Game

Darkwinds is an online trading card game where two players confront each other in a sea battle of ships.

While the smart contract handles the ownership of tokens, game matches occur off-chain, in a webGL website running the Metamask web extension or compatible thin wallets. A game server is responsible for matchmaking between two adversaries, validating the signature of both players, thus verfying the ownership of both player decks. While the official game server will be the only endorsed way of playing Darkwinds, other developers are free to read the ABIs and access players cards to create different game modes, tournaments or applications that connect to the game.

Game servers only require a signed message from the user wallet to verify ownership. The user private keys are never read or stored.

3 Anatomy of a Darkwinds token

The card smart contract performs all the functions according to the ERC721 [1] standard. The cards are generated with a payable function called getBoosterPack and the amount of cards returned to the owner is determined by the price of cards, which is set by the owner using the setCardPrice function

3.1 getCard

Gets all the metadata for the card relevant to the game mechanics (actions, cost of invocation, effects, etc.)

3.2 getBoosterPack

Spends a determined amount of ETH to get a booster pack. Normally it will contain 10, 25 or 50 cards but the contract can be excecuted for a specific amount of cards multiplying the desired amount of cards with the cost.

3.3 changeCardCost

A function designed for the owner to change the price of the card, and by consequence the price of booster packs. By default is 1 finney.

4 Card Generation

Cards are generated with a KECCAK-256 operation on the last block timestamp and a modulo operation of 50. Every 5 cards we add another 50 modulo on the contract nonce in attempt to make a rare card. If both modulo operations are larger than 25, a rare card appears. The higher the model ID, the rarer the card is as shown in Figure A.

While it's possible to determine exactly when cards are being released, efforts are probably not practical.

The smart contract stops generating cards when the hard cap of 1,000,000 is reached.

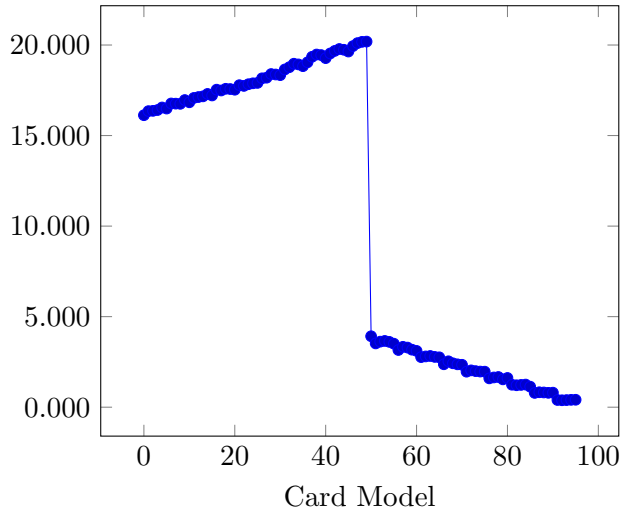

Figure 1: Distribution of 1 million cards estimated by a Monte Carlo simulation

References

[1] The ERC721 non-fungible token standard. https://github.com/ethereum/eips/issues/721, Dieter Shirley, 2017.