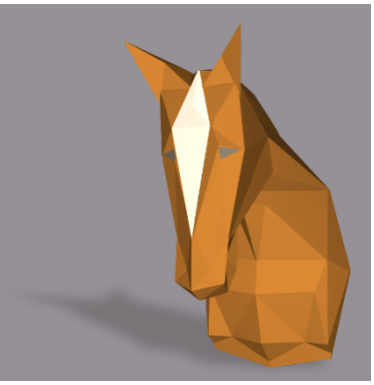
ETHORSE WHITEPAPER

THE WORLD'S FIRST DAPP TO BET ON THE PRICE OF CRYPTOCURRENCIES

DRAFT VERSION 0.26

Table of Contents

ABSTRACT



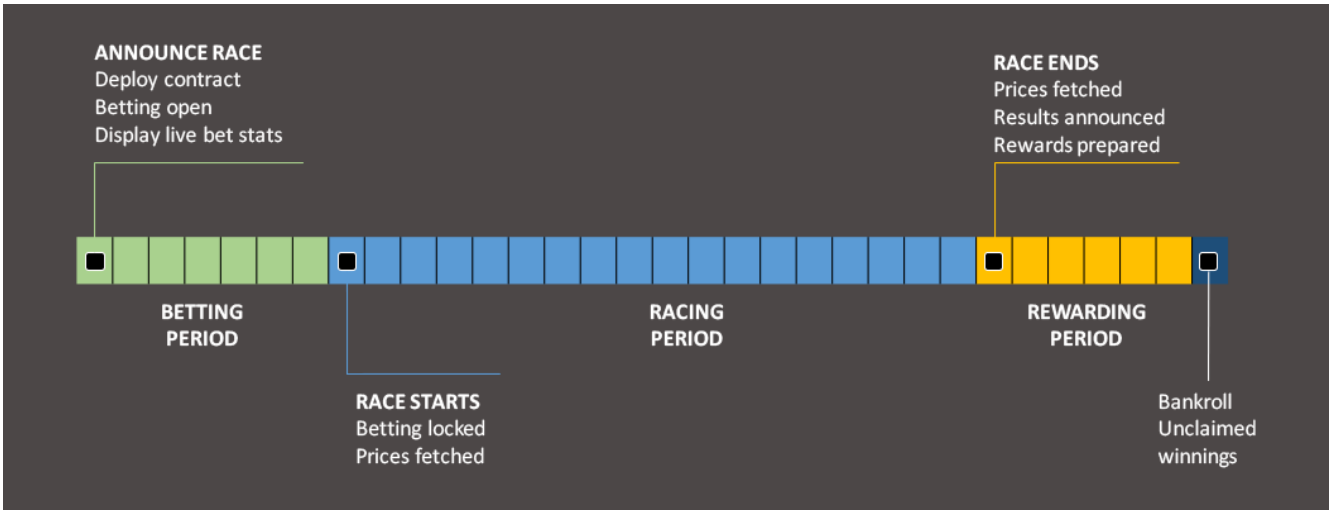
ETHORSE is the ultimate combination of horse betting, blockchain technology, and cryptocurrencies. It is an Ethereum Smart Contract based DApp for betting on the price of Cryptocurrencies and win from everyone who bets against you. Ethorse is an interesting tool for all cryptocurrency traders who like to bet on the price movement of Cryptocurrencies. Users bet with ETH on one of the listed coins or tokens to have the highest gain in a fixed period (Parimutuel Betting). All bets are placed in the blockchain, allowing users to see the total number of bets, the total bet amount for each coin, and the Payoff odds deriving from them. This allows users who follow the crowd to pick a winner easily. However, the Payoff-odds can also be used to bet on an underdog for very large wins. The platform can be used to bet and win ETH in both Bull and Bear markets. Compared to mainstream betting sites and centralized exchanges, Ethorse holds many advantages as it does not require users to signup and deposit funds to an entity. As the development is open source and the funds are secured by the Ethereum Smart contracts, Ethorse enables a completely trustless peer-to-peer betting environment. The prices are pulled from the exchanges using Oraclize and it is done in a way to avoid manipulation. With a user-friendly interface, Ethorse is a groundbreaking platform and has the potential to attract a large set of the trading population in the crypto space.

HORSE, the platform's tokens, are issued to the crowdsale contributors. A 5% takeout from all the betting pools goes to the reward pool, which can be claimed by the HORSE token holders by using our dApp and staking the tokens during a fixed period. The 5% takeout from the bet pools is minimal compared to the usual 15%-25% in horse betting [1]. At the end of each quarter, a smart contract on our dApp site will be used to claim ETH from the reward pool based on the proportion of HORSE tokens staked in a compatible wallet during a particular time period. For example, a wallet staking 10% of the total HORSE token supply in our dApp will receive 10% of the reward pool. As the bets are always placed between the users and bankroll does not pay the winners, the reward pool balance will always remain positive. As the platform gains popularity, Ethorse will allow users to bet using HORSE tokens which can unlock advanced features and enjoy a smaller takeout.

INTRODUCTION

Ethorse provides a user-friendly interface to interact with the betting smart contract that handles all operations in a secure way, including collecting bets, monitoring prices, calculating winners, and rewarding. The betting system used by Ethorse is called Parimutuel Betting [2], and it is popular in horse betting and many sports where participants finish in ranking order. In Parimutuel Betting, all wagered amounts are placed together in a pool, the house or race track takes a cut from the pool, then the reward is shared among everyone who bets on the winner in proportion to the amount they bet. Ethorse uses this betting system where users bet on the price of cryptocurrencies by choosing a winner among multiple cryptocurrencies. A winner is a coin or a token that has the highest price increase % among all the other competing coins or tokens in a specific time period. Ethorse implements this purely on the blockchain without any interaction with the outside world except to fetch the prices from the exchanges using Oraclize. A race is published or announced when a contract is deployed. All the variables are set at this time such as the three sequential periods explained below. No one can interfere or influence the contract once a contract is deployed and a race is made public.

FIG 1. TIMELINE OF A RACE IN ETHORSE



BETTING PERIOD - It is the time period following the announcement of a race during which users can place their bets using the Ethorse DApp in a Metamask™ (https://metamask.io/) compatible browser such as Google Chrome™, Firefox® or an application from Ethereum foundation such as Mist or Geth. During this period, users can see in real time all the information about placed bets such as the wagered amount and the number of bets on each token/coin. Win odds are calculated and updated as bets are placed. Win odds are explained with algebraic representation in the below section.

RACING PERIOD - At the beginning of this period, the contract is locked for further betting. Any transactions trying to place a bet would fail, not deducting the corresponding balance from the user's wallet. Price of each coin/token is fetched from the exchanges at the beginning of this period and displayed on the betting page as "Race Start Price".

REWARDING PERIOD - At the end of the racing period, prices are fetched again from the exchanges and displayed on the website as "Bet Close Price". The price increase % from the "Bet Lock Price" to "Bet Close Price" is used to decide the winner. The winning coin is announced in the web interface. Users can use the function "Check Result" to view their winnings and use the "Claim" function to execute a transaction without sending any ETH and include only the gas (or transaction fee). This triggers the smart contract to send their winnings in the form of ETH. Users have a set period (currently 30 days, subject to change with a notice) to claim their winnings. One of the Ethorse Betting contract addresses used in November 2017 that is verified in Etherscan™ is 0x4524e8425760AF95DA970A066bd74092b36f6DB0. All unclaimed winnings go to the bankroll and are included in the reward pool. In a race with a set of n cryptocurrencies listed for betting, with a total bet pool of each cryptocurrency B1, B2, ..., Bn, the total pool of money on the event is: [2]

$$B_T = \sum_{i=1}^{n} B_i$$

After a takeout of "t" from the pool, the amount remaining to be distributed among the successful bettors is $B_R = B_T(1 - t)$. Those who bet on the winning outcome "w" will receive a payout of B_R/B_w for every ETH they bet on it.

The web interface enables users to browse through past games, check their results, and claim their accumulated rewards. The interface also presents information on the status of ongoing races, races that are open for betting, and a schedule of upcoming races. Users will be able to choose from multiple games that are open for betting, view the status of on-going races, and claim their rewards from completed races. There are several types of games categorized by the race period (24hr, 1 week, etc.) and bet type (Win, Place, Show [3]). For the beta release on Testnet, the most popular version of horse betting - "Win" was provided to the users. In this type of bet, users pick a winner for the #1 spot. A 24-hour betting period (upon the announcement), 72-hour race period to find a winner, and then a 30-day reward claim period was applied. A countdown timer will always be running on the race page to show the beginning or end of the corresponding time period the race is currently on. All the terms of the race are transparent for the public to verify anytime, thanks to the nature of the Blockchain and Ethereum. Below shown is a demonstration of a possible scenario from a single race to help understand the bets and payout.

FIG 2. EXAMPLE RACE SCENARIO (BETA AS OF OCT 2017)

Select a coin	Pool Total	Odds(Profit for 1 ETH)	Number of bets	Bet Lock Price	Bet Close Price
BTC	676.42	1.34	117	6524.12	6604.44
ETH	833.56	0.9	128	342.87	351.12
LTC	157.54	9.06	31	61.05	62.86

Ξ | Place bet

Check result | You have won 33.72 ETH. | Claim

(A screenshot from the working beta [4])

Total Pool = B_T = 1,667.52 ETH
Takeout Rate = t = 0.05
Total Payout = $B_R = B_T(1 - t)$ = 1,584.14
Winning Pool = LTC with 2.96% price increase (BTC +1.23%, ETH +2.41%)
Winner Payout = B_R/B_w = 1584.14/157.54 = 10.06 ETH, i.e., 9.06 ETH profit for 1 ETH bet on LTC

Bankroll (HORSE holders) receive 83.38 ETH (B_T-B_R) and any unclaimed winnings to claim at the end of the quarter from this one race. If Ethorse can run one race similar to the this demonstrated race every day, HORSE holders can earn ~7500 ETH at the end of the quarter. As Ethorse scales, it will have much more user bet amount, coins to bet, and types of bets. **This shows a huge potential market value for the Ethorse platform.**

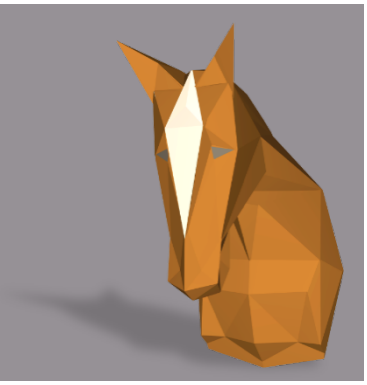
AVOIDING MANIPULATION

The prices fetched from the exchanges using API calls through Oraclize [5] plays a critical role in determining the winner. Hence, the best techniques and algorithms are used to avoid any manipulation. For this purpose, the prices fetched from multiple exchanges will be aggregated instead of using one source. The trading volume of the coin or token in exchange will be taken into account. The time at which the price from the exchanges is pulled will be randomized within a specific short time period to avoid possible price manipulation in exchange orders. In the beta, prices are pulled from the Coinmarketcap™ API [6].

SAFETY OF FUNDS & REFUNDS

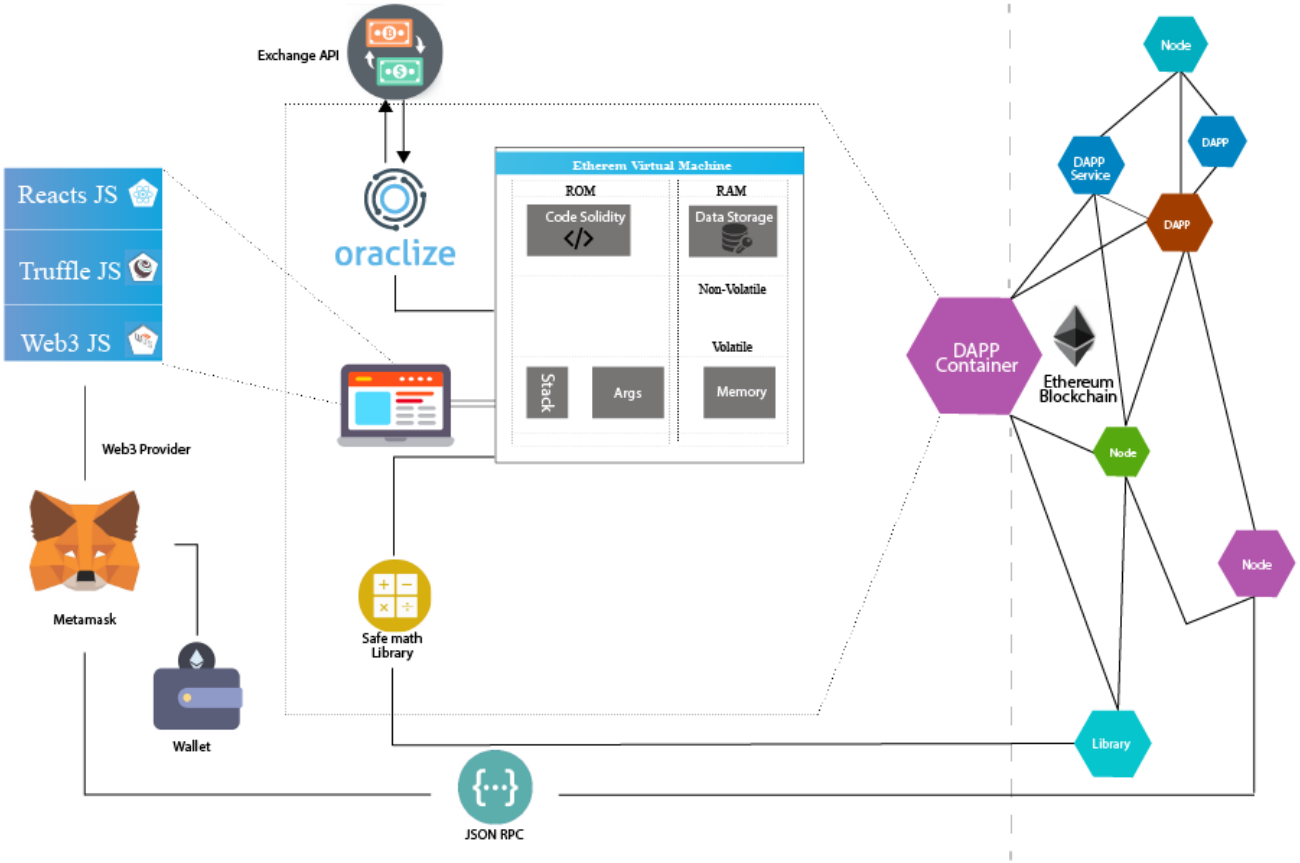
If the contract fails for any reason, one being an issue with fetching the price from the exchange, there are safety mechanisms in place to cancel the race and issue a full refund to all the bettors after the gas costs for the transaction. It can be publicly verified in the open source smart contract code that no one including the contract owner will be able to take control of the funds by breaking the contract. The only action that a contract owner can take to unlock the funds for anyone is to make them available for all bettors to withdraw them. This works in a similar way with the *"Claim Reward"* function as explained in the previous section. More technical details about its implementation are available in the following section.

ARCHITECTURE



Ethorse is a Decentralized Application built completely inside the Ethereum blockchain [7]. The entire DApp lives inside the Ethereum Virtual Machine (EVM) [8]. The EVM has two types of components, the volatile and non-volatile. Ethorse DApp contains the code, data, stack, arguments, and memory. Since the EVM lives inside the Ethereum network, any read or write of data or operation has to go through the blockchain. In the following sections, the workflow and implementation of the smart contract are discussed in detailed. While it uses the architecture of the development done so far for the beta, a scaling solution towards the release of a full suite of cryptocurrency price betting is being meticulously worked on by the team.

FIG 3. ETHORSE ARCHITECTURE



BACKEND

The backend component consists of the core logic behind the DApp, which is a smart contract [9] in the Ethereum world. The smart contracts for Ethorse have been written in Solidity [10] language. Truffle [11] framework is used for the administration and management of the contract and the frontend. The DApp structure of Truffle is followed. All the contracts in the Ethorse environment are built by following the OpenZeppelin™ [12] secure smart contract protocol. Apart from the core logic, there are also additional libraries and services used by the DApp. One of the main supporting components is Oraclize. Oraclize is used as a data carrier. Since the DApp resides in a closed environment, the only way to get in touch with the external Web API is by using Oraclize. Oraclize works as a bridge between the DApp and external APIs. External APIs are used to pull prices from the exchanges, Coinmarketcap™ (for beta), and random.org™ for unbiased random number generation used in randomizing the time at which the price is pulled from the exchanges.

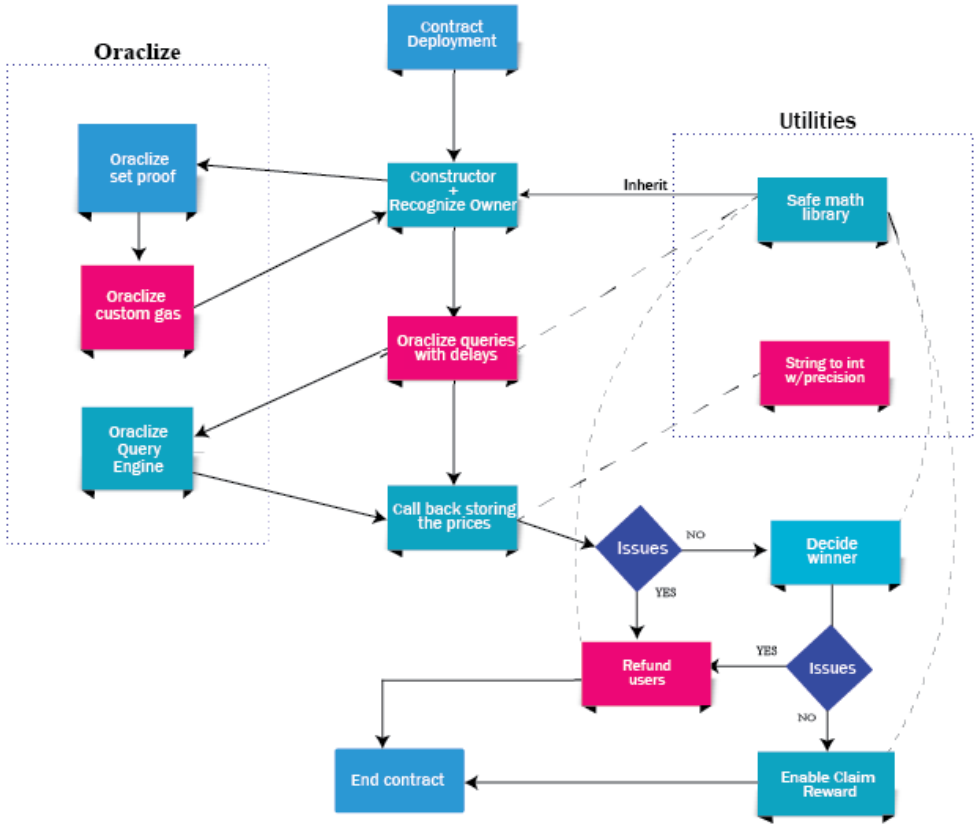
FRONTEND

React and Redux are used for the frontend development. The choice of React derives from the need of working with the Ethereum blockchain, which makes a state change for every action or operation performed in the blockchain. Every state change that occurs in the blockchain is tracked and the changes are reflected in the DApp for the users. Though this could be achieved with vanilla HTML CSS and JS stack, for a seamless integration and a smooth user experience the power of React is harnessed. The UI interacts with the blockchain using any type of web3 provider. The frontend code has been tested to work with some of the frequently used web3 providers such as Metamask™ browser extension on Google Chrome™ or Firefox® browsers and Mist from Ethereum foundation. The frameworks that are used to interact with the frontend include Ethereum web3 JavaScript framework and Truffle™ Suite frameworks such as *truffle-contract* [14] and *truffle-artifactor* [15].

WORKFLOW

Ethorse is designed with a simple and straightforward workflow to minimize any security risks. By using the appropriate and absolutely necessary functionality, the complexity of the application has been masked by the optimal betting algorithm.

FIG 4. CONTRACT WORKFLOW



The starting point of the contract is the constructor. The constructor is the method which gets invoked when the contract is deployed in the blockchain. When deploying the contract, the constructor sets the gas amount required to execute the Oraclize queries. This predetermined gas is termed as “Starter Gas” and it is fed to the contract by the Ethorse Bankroll. The constructor also sets the notary for the Oraclize service and calls the update method. The update method is responsible for setting delays and placing the Oraclize queries. The delay in the Oraclize calls determines the betting period, racing period, and rewarding period. Once a race is selected, the user picks a coin that they want to bet on and enters the amount to place a bet. This action involves a transaction that contains the user’s selection of coin and the value as payload. The state change is made in the blockchain by the smart contract. The contract not only logs the event of placing a bet but also stores all the information of the bettor in a *structure*.

All the bets are placed and locked at the end of the betting period. At this time, the Oraclize queries placed by the contract result in fetching the price of the coins at a randomized time within a predefined period. Similarly, the bet closing price is fetched at the end of the racing period, after which the winning pool is identified, and the rewards are prepared for collection. If there are any issues with the price returned from Exchange API calls or from Oraclize, there is a refund mechanism already built in the code. Once the “Bet Closing Price” is received, the gas amount for claiming the reward is determined by the *estimateGas* method provided by the contract instance. The bettors can claim all their unclaimed rewards from previous races by verifying with their wallet used for betting.

IMPLEMENTATION

The website interacts with the smart contract using the Truffle library. It makes use of an artifact to interact with it. A race can have a minimum bet amount and a maximum bet amount as stated. Once a bet is placed, the amount is converted to Wei [16] before the bet is stored in the smart contract and logged as an event. After the betting period is over, “Place Bet” action is disabled in the user interface. This is to help users not to spend gas unnecessarily. The “Place Bet” button will also be disabled whenever a bet is placed until a transaction ID for the previous bet is returned by the network. Inside the contract, there are several variables, structures, and methods exposed to the public. The *getter* methods defined in the contract are for exposing the data in the contract to the frontend. There are three structures present in the contract. The first one stores the betting information. Each bet that is placed will have its data stored in this structure. The second structure is used for storing information about each listed coin and the corresponding bet pool information. The third structure is used for storing the information of every bettor. All the bets that are placed by the bettor are stored in an array of betting information structure. The contract also leverages the usage of hashmaps. There are three hashmaps that are being used.

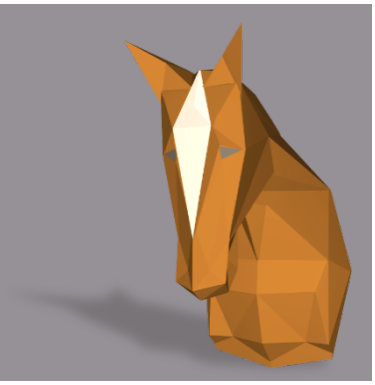
The first hashmap is for mapping the Oraclize IDs with its corresponding coin. Since the Oraclize works by using the *callback* functionality, it requires a methodology to track the query for which the callback has been invoked. In addition to this, there are several logs created to log every event that makes a state change in the contract. All the events used are indexed. Indexed events are used to help when there are a large number of events logged, as it will be useful in searching efficiently while identifying winners and rewarding. There are modifiers that are defined to enforce a restriction on the access to several methods. The update function is where the Oraclize queries for fetching the prices are implemented. There are two Oraclize query calls for every coin source: the “bet locking price” and the “bet close price”. All the coin prices are requested and fetched in the form of a JSON and the price of each coin is parsed from it using a JSON parser library for Solidity. The framework used for this is *“JsmnSolLib”.* Once the prices start arriving in the callback method, the *coin_info* structure is updated with the data. There are also Boolean flags present to segregate the “bet lock” and “bet close” prices. A logic is implemented to check if the price-data meets the requirements for the reward method to be invoked. Once all the price data arrive, the reward method is invoked internally.

The calculation of finding the coin with the highest gain is performed by calculating the percentage price increase from the “Bet Lock Price” to “Bet Close Price”. This is termed as the “Coin Delta (CD)”. The CD is calculated to a precision of 5 decimals. In case, the CD of two or more coins is equal and larger than the CD of remaining coins, all the coins with the largest CD are declared as winners and rewards are provided for everyone who bet on any of those coins. The method for calculating the reward has been deliberately made a constant function so that the bettors do not have to spend gas to check their bet results. If the user decides to claim the reward, the user invokes the *claim reward* method. This method uses some amount of gas similar to the *placebet* method. As mentioned in the previous section, if there is any technical issue in the contract, the contract owners will not be able to withdraw the user funds to their wallets. The only possible action is to unlock the fund for the bettors to receive a full refund.

CODE AUDIT & PROGRESS

The first beta of Ethorse DApp (http://ethorse.github.io/Betting/) [17] was released on October 26, 2017. This release was announced on Reddit [18] to collect feedback from the developer community (r/EthDev). The DApp was deployed and tested in Ropsten Testnet [19]. Both positive and constructive feedback was received to improve the app. It provided the team with ideas to improve the contract logic and workflow, and also encouraged the team to add some new features. After some internal testing of new features added, on November 4, 2017, Ethorse announced the first public bug bounty for the developer community. The contract was deployed on Ropsten Testnet. Several Solidity developers reviewed the code. Issues are reported by users on the GitHub [20]. While the team is proud that there were not any successful attacks made to withdraw the user funds, the participants exposed some potential vulnerabilities. The most severe vulnerability reported is if the contract owners are to play a bad actor role, that they had the ability to influence the race. It is very important for the team to fix such issues as the vision is building a truly decentralized and transparent DApp by not allowing anyone to control the contract once a race is published. All the participants who discovered qualified bugs are awarded ETH. Getting the smart contract code reviewed by multiple independent and experienced Solidity developers have proven to be more useful than working with one auditing business. Here is a recent bug bounty that we conducted: https://www.reddit.com/r/Ethorse/comments/86gx86/ethorse_smart_contract_bug_bounty/

ROADMAP



Oct 2017

- Developer Beta on Testnet (Completed)

Nov 2017

- Updates, Public Bug Bounty for Betting Smart Contract (Completed)

Dec 2017

- Crowdsale announcement & contract dev, audit, Public Testnet Beta

Jan 2018

- Phase 1 & Phase 2 Crowdsale, Token Distribution, Mainnet soft launch

Quarter 1 2018

- Exchange listing, Full launch, Recurring bets, UI/UX enhancements

Quarter 2 2018

- Gas optimization, UIUX enhancement, Seeded races, in-house Oracle, staking/payout claim portal

Quarter 3 2018

- Price index, additional races and coins to bet, Exchange listing, Marketing and Partnerships

Quarter 4 2018

- Additional language support (Chinese, Korean, etc), and promotions, Mobile support

HORSE TOKEN & CROWDSALE

HORSE is an ERC 20 standard token [21] created with a strictly limited supply issued to the crowdsale contributors. A takeout is taken from the bet pool of each race and deposited in to the reward pool before the winners can claim their winnings (83.38 ETH in the race demonstrated in Section II). The current takeout rate is at 5%. At the end of each quarter, a smart contract on our dApp site will be used to claim ETH from the reward pool based on the proportion of HORSE tokens staked in a compatible wallet during a particular time period. For example, a wallet staking 10% of the total HORSE token supply in our dApp will receive 10% of the reward pool. As the bets are always placed between the users and bankroll does not pay the winners, the reward pool balance will always remain positive. As the platform gains popularity, Ethorse will allow users to bet using HORSE tokens which can unlock advanced features and enjoy a smaller takeout. As the platform scales, a monthly reward staking and claim mechanism will be rolled out. HORSE tokens will be used for betting as the platform gains adoption and users will enjoy discounted takeout rates in the races where HORSE tokens are used. HORSE tokens will also unlock advanced features for the bettors.

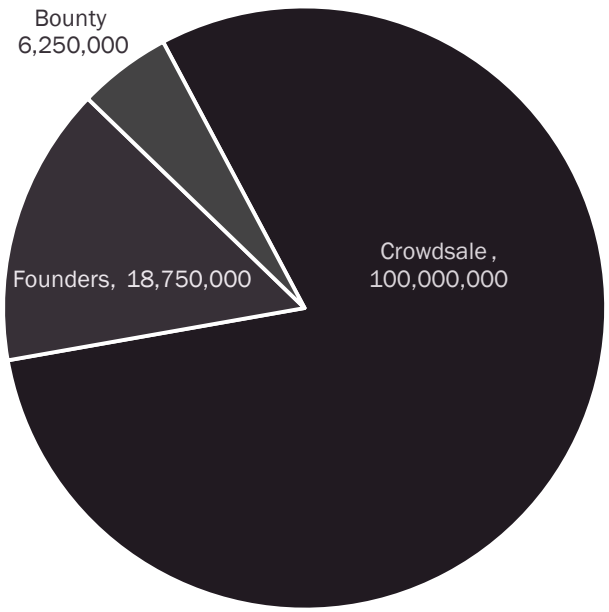


FIG 5. HORSE TOKEN DISTRIBUTION

Token Symbol	HORSE
Type	Ethereum ERC 20 Standard
Contract Address	Will be made available in the website before crowdsale begins
Total Fixed Supply	125,000,000 HORSE tokens minted
Crowdsale Hard Cap*	100,000,000 HORSE tokens
Crowdsale Soft Cap**	400 ETH
Founders Token***	18,750,000 (Vested over 2 years)

*Crowdsale Hard cap will be 4,000 ETH if we sold all tokens during Phase-1 sale.
**If there is no sufficient interest from the crowdsale, Ethorse will refund all the ETH collected from the crowdsale upon community consensus.
***Founders' tokens are time-locked with an Ethereum Smart Contract, vested over 2 years with a cliff period of 6 months.

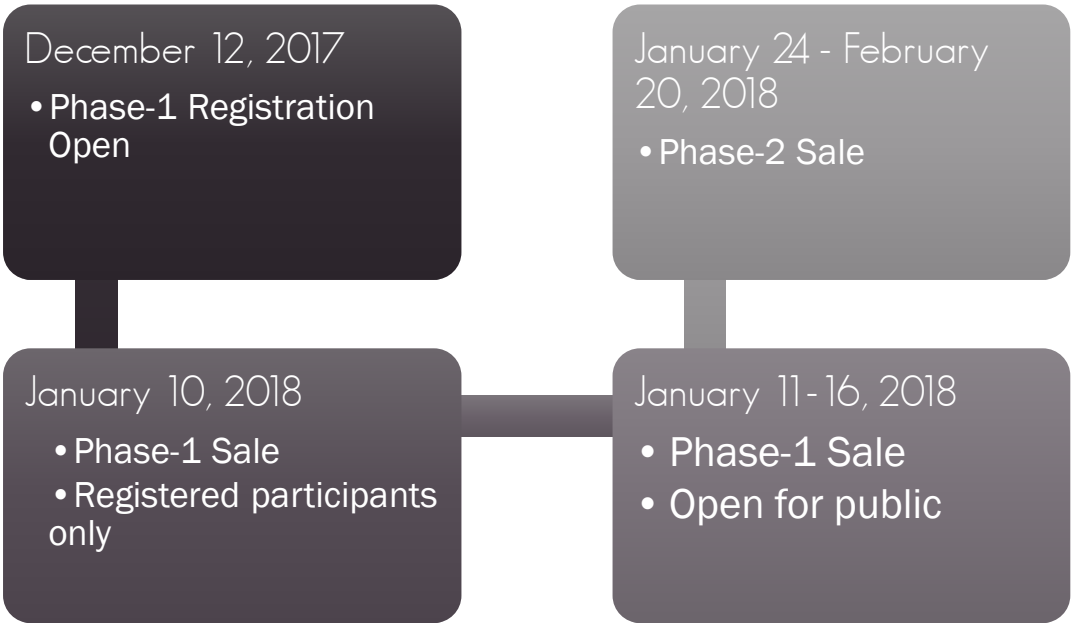
All unsold tokens are burnt either a week from the time the hard cap is reached or a week after the end of the crowdsale period, whichever occurs first. In this case, Founders' tokens will also be burnt to maintain the distribution rate of 15% to the founders. There will not be any mintage of HORSE tokens in the future. Tokens will appear in the wallet within a week from the crowdsale contribution.

CONVERSION RATE

Period	ETH	HORSE	Bonus
Phase 1*	1 ETH	25,000	100%
Phase 2 - Day 1	1 ETH	18,750	50%
Phase 2 - Week 1	1 ETH	15,000	20%
Phase 2 - Week 2	1 ETH	13,750	10%
Phase 2 - Weeks 3 & 4	1 ETH	12,500	0%

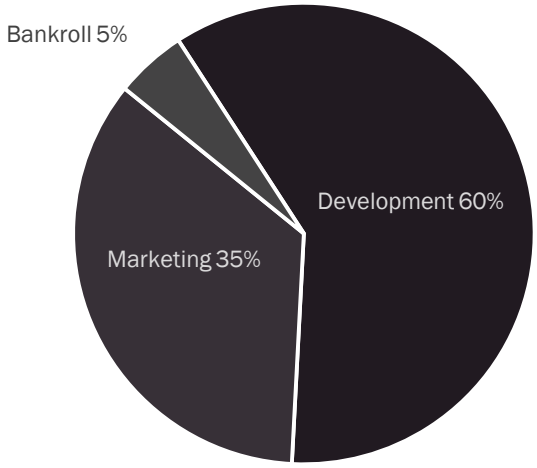
*People registered for the Phase 1 in our website are given priority to participate in the Phase 1 sale before it is opened for the public.
To maintain a healthy token price upon listing on exchanges, bonus tokens are unlocked for transfer 30 days after the base tokens are unlocked. If the crowdsale ends in Phase-1, bonus tokens are unlocked along with the base tokens.

SALE TIMELINE



*Phase-2 sale subject to change based on Phase-1 progress

FIG 6. ALLOCATION OF FUNDS COLLECTED IN CROWDSALE

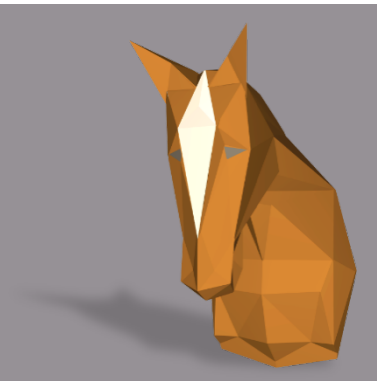


Funds raised from crowdsale will be used to bet on the first few races. This liquidity will attract users to try and bet on the Crypto prices using the DApp.

THE TEAM

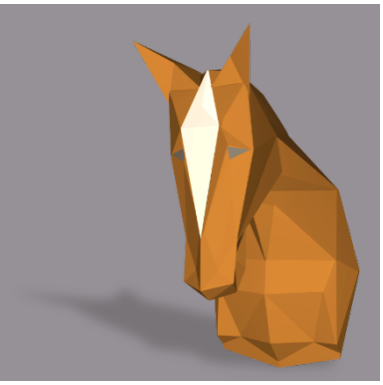
The core team comprises of three members with experience in information security and blockchain development. They share the responsibilities of programming in Solidity (Smart Contracts), React.js and Redux.js (web3), HTML, CSS, graphics design, strategy, business development and community management. The team prefers to be anonymous for the benefit of the project until there is a need. An article published by the team on December 29, 2017 provides more clarity and transparency about the need for anonymity (Link: https://medium.com/@ethorse/ethorse-team-transparency-d9e944c1b4d2). Inspired by projects such as Etheroll, the team wants to follow a similar path. All the work that is produced so far is done in-house, including the working dApp that is released to the public. As of November 20, the team has made 117 commits to the open source code repository on GitHub and it's growing with the team being active almost every day (https://github.com/ethorse [22]). This credibility proves that the team is fully capable of executing this project. The team will continue to deliver as per the roadmap. The vision is for Ethorse (and HORSE) to be owned and supported by the community. Discussions and inputs are always welcome in the dedicated communication channels. Ethorse welcomes anyone to build their own applications using the Ethorse smart contracts.

DISCLAIMER



HORSE tokens should not be considered as an investment that provides guaranteed returns or increase in value. It also does not grant any controller ownership, equity, direction, or decision-making of Ethorse platform as a whole. You should not participate in the Crowdsale if you are a Resident or Citizen of the United States of America, including Puerto Rico, Virgin Islands (U.S.) and any other U.S. possession, Singapore, Hong Kong, Peoples Republic of China, or any other jurisdiction in which it is not permissible to participate in token crowd contributions or acting on behalf of any of them. Crowdsale participants and platform users should follow the rules of their jurisdiction to abide by their local security and online gambling laws. As with any other cryptocurrency technology, there are risks of theft, hacking, lack of adoption and technical issues that can make people lose ETH, HORSE tokens, or their market value. Since the crowdsale is performed by collecting ETH, Ethorse team may not be able to fund and continue developing the platform if the market value of ETH declines drastically.

REFERENCES



1. http://horseworlddata.com/pmtrcks.html
2. https://en.wikipedia.org/wiki/Parimutuel_betting
3. https://www.bovada.lv/help/win-place-show
4. https://ethorse.github.io/Betting/
5. http://www.oraclize.it
6. https://coinmarketcap.com/api/
7. https://ethereum.org/
8. http://ethdocs.org/en/latest/introduction/what-is-ethereum.html#ethereum-virtual-machine
9. https://en.wikipedia.org/wiki/Smart_contract
10. https://solidity.readthedocs.io/en/develop/
11. http://truffleframework.com/
12. https://github.com/OpenZeppelin/zeppelin-solidity
13. https://etheroll.com
14. https://github.com/trufflesuite/truffle-contract
15. https://github.com/trufflesuite/truffle-artifactor
16. http://ethdocs.org/en/latest/ether.html
17. http://ethorse.github.io/Betting
18. https://reddit.com/r/ethdev/
19. https://ropsten.etherscan.io/address/0xc183960d62A3db3e3eC9e85fA89Da9C2E8F313B1#code
20. https://github.com/ethorse/ethorse-core/issues
21. https://theethereum.wiki/w/index.php/ERC20_Token_Standard
22. https://github.com/ethorse
23. https://reddit.com/r/Ethorse