Network: Ethereum

Address: 0xe24F992D6E34357cF67D741769eAAD7bC44E32DD

https://github.com/krakintgithub/solidity/tree/master/miner

Document Date: December 2020, Version 2.0
Authors: Krakin't Team

Web: http://www.krakint.com

INTRODUCTION

· Although we are using the word **miner**, the Krakin't miner is very similar to **staking**. While mining implies searching for data, we are simply using the time-reference and collective staking to calculate the rewards.

· Since we are relying on the Ethereum miners to mine the data-blocks, we are using the word "miner". As a result, we can mine Krakin't tokens with no electricity or special hardware since Ethereum does all the hard work for us.

· We are applying the proof-of-burn concept to have a clear presentation between the market demands, circulating tokens, and the token market prices. Simply, we don't circulate tokens that are idle, while we inflate the supply automatically and by demand. In this sense, Krakin't is neither a stable-coin, inflationary nor deflationary. It is simply a precise-coin.

Krakin't Miner Doodle

What Makes it Unique ?

Some of networks, such as the Bitcoin, work by applying the proof-of-work. This means that in order for the network to progress the ledger, some computing needs to be performed too. The keyword here is the word "proof". With decentralized nodes, we can compute whether the proof is true or false, and as a result, have the safest and most reliable networks. Ethereum gives us such network-strengths, and it allows us to apply our own creations indulging in its safety. Since all proofs exist with the Ethereum network, Krakin't miner does not need to apply any proof mechanism. We can substitute the word "work" with anything and build any mechanism around it. To have a profit-oriented network built with Ethereum, the only required work is inserting the Krakin't tokens into a miner, and letting the miner do everything else.

We have narrowed the miner function to just two transactions; one is destroying (or burning) the tokens, and other is creating (or minting) the same tokens. With Ethereum safety, we can simply track all actions, and reward the miners accordingly... With the passing time, we can simply reward the accounts for having the tokens burned with the miner. Except for the burning part, this kind of approach is equivalent to staking.

In the previous version of Krakin't miner, we have mentioned a mechanism that balances impact of minting and burning tokens by using the miner. However, we have removed this mechanism and simplified the mining process. We gave some advantages to early miners, while the mining difficulty is slowly increasing with time. Furthermore, the mining difficulty does not decrease, unless we do it manually. Since it is impossible to update everyone's tables without wasting too much Ethereum GAS, we have introduced an **asynchronous mining difficulty**.

Asynchronous mining difficulty works by making two calculations. First, we keep a track of the global time that passed since the mining contract got deployed on the Ethereum network. We use the Ethereum block numbers instead of the time-units such as seconds. Mining one block on Ethereum takes 12-20 seconds. As soon as the miner deposits or withdraws tokens from the miner contract, we increase the mining difficulty for them by assigning them a new Ethereum block number. As long as they do not make any changes (this includes adding more tokens to a miner), their mining difficulty will remain the same. This is how the mining difficulty is asynchronous.

The second calculation involves a further change of the mining difficulty variables. To adjust the mining to respond to a token demand, we are also changing the mining difficulty variable as soon as the miner initiates a withdraw The mining difficulty starts by giving the 2% earning per day. If the miners were constantly withdrawing their tokens, it would take 20 years to decrease the earnings to 2% per year.

This design, however, begs one question. What if someone used the miner simply to mint tokens at an exponential rate and increase the mining difficulty for everyone? We have increased our potential for the mining statistics and we can track such behaviour. Although it is highly improbable that anyone will abuse the mining contract, we have added an option to either blacklist the miner or annul their assets. Furthermore, we have the ability to alter the mining difficulty by hand should the mining need some fine-tuning.

To understand how the mining works, we need to explain how the mining difficulty is calculated.

How do we calculate the mining difficulty or earnings ?

First, there is a global variable called Difficulty Constant. Whenever a user initiates a deposit or withdraw, we memorize the current Ethereum block number for that user. We also memorize the current difficulty constant, since this number changes too. Now, let us assume that we want to see how much we earned after depositing N tokens. This is calculated in a following manner:

LET

N = amount of tokens deposited

L = last known block number memorized

C = current block number

D = last known difficulty constant memorized

R = current reward constant

THEREFORE

$$\textbf{Earnings} = [(N*(C-L)*DL/\ R]$$

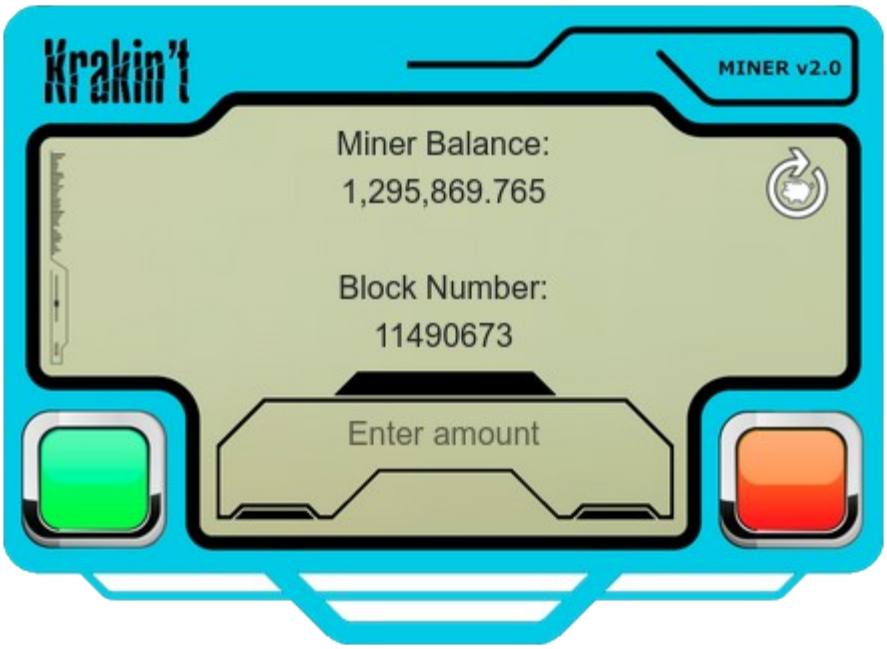
$$\textbf{Total} = N + \textbf{Earnings}$$

Whenever a user initiates withdraw, we simply decrease the difficulty constant by another constant This way, it affects difficulty for every other user who initiates a deposit or a withdraw function.

User Interface

The miner user-interface was changed to match the style of the pre-sale interface.

Old user interface for the miner version 1.0

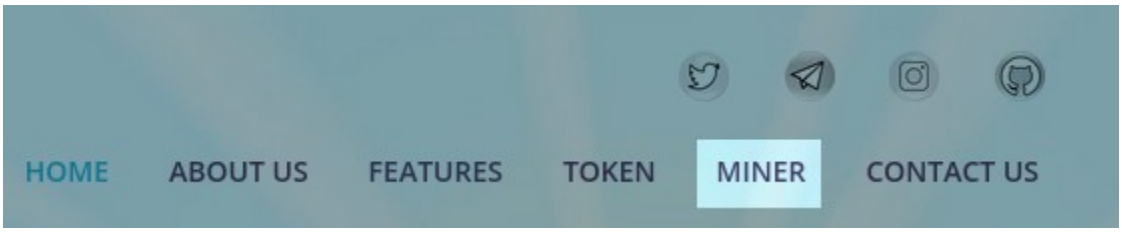


New user interface for the version 2.0

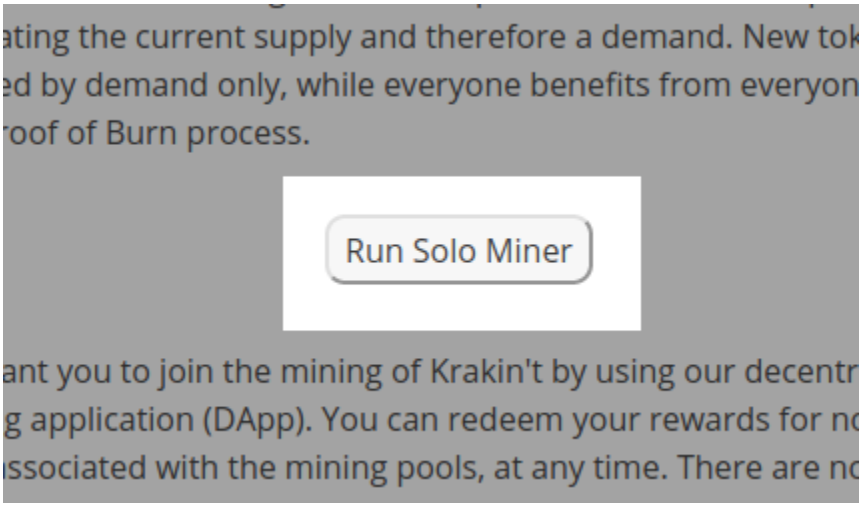
The idea is still the same. There are two buttons, one is for deposit (green) and other is for a withdrawal (red). If the miner is not communicating with Metamask, it will display message on the screen. Once Metamask is connected, you have to manually reset by pressing the reload icon placed in the upper-right corner of a display screen. As you hover over the buttons with your mouse, you can see description of a button's function displayed on the screen.

Accessing from the www.krakint.com

To access the miner, please go to a main-page and select "Miner" from the upper-right corner.



Next, click on the "Run Solo Miner" button, and the miner will pop-up.



Other miner's functions

Although the mining contract can do much more, we have narrowed it down the mining and the user-interface to two simple functions. One is a deposit, other is withdraw. For this, we need just two fields and two (or just one) submit button. Other functions are meant to allow us to keep the clean statistics regarding the accounts and their earnings. The functions that are worth mentioning are:

For calculating the mining difficulty:

- getRewardConstant() - Reward constant.
- getDifficultyConstant() - Difficulty constant.
- getDecreaseDifficultyConstant() - Constant by which difficulty decreases.
- getMintDecreaseConstant() - Constant by which the minting decreases.
- getCreationBlock() - The block number when the miner was deployed.
- getUserDifficultyConstant(address minerAddress) - Miner's unique difficulty constant.

For user tracking:

- getPivot() - The last registered miner index
- getIdFromAddress(address minerAddress) - Converts miner address to index, if registered.
- getAddressFromId(uint id) – Converts index to a registered miner address.
- getDepositedTokens(address minerAddress) – Amount of deposited KRK tokens for miner address.

For general statistics:

- getTotalMinted() - Total amount of KRK minted (withdrawn).
- getTotalBurned() - Total amount of KRK deposited to a miner.
- getCirculatingTokens() - The amount of tokens currently deposited to a miner.
- getUserTotalMinted(address minerAddress) – Amount of tokens a specific user has minted.
- getUserTotalBurned(address minerAddress) – Amount of tokens a specific user has deposited.
- getUserNumOfDeposits(address minerAddress) – Number of deposits user made.
- getUserNumOfWithdrawals(address minerAddress) – Number of withdrawals user made.

Other:

showEarned(address minerAddress) – Amount of tokens the user earned

showReward(address minerAddress) – Amount of tokens the user gets as a mining reward.

Please note, these functions are not available through the miner user interface, and are to be used for tracking the statistics.

The calls to a contract are called every 3-6 seconds, and the total that can be withdrawn is displayed per new block. Therefore, the Krakin't miner can be accessed by anyone and used by anyone, regardless their technical skills. Nevertheless, since we do need a third-party plugin to accompany the interface, the learning curve may be a bit steep for the beginners.

CONCLUSION

The purpose of the Light-Paper is to be as simple and up to a point as possible. Nevertheless, the source-code is open to anyone and every function and a variable are documented as a Readme.md.

Please see the GitHub link:

https://github.com/krakintgithub/solidity/tree/master/miner

We hope that you will enjoy the simple interface and the miner which uses no electricity and no hardware to work, except the working Internet connection to access it when necessary.