

EIP 1559: Fee market change for ETH 1.0 chain

[Return to Ethereum ERCs](#)

: **Note:** The following is an excerpt from the official Ethereum site. It is provided here as a convenience and is not authoritative. Refer to the original document as the authoritative reference.

Table 1: Data sheet for Fee market change for ETH 1.0 chain

Title	Fee market change for ETH 1.0 chain
EIP	1559
Author	• Vitalik Buterin (@vbuterin), • Eric Conner (@econoar), • Rick Dudley (@AFDudley), • Matthew Slipper (@mslipper), • Ian Norden (@i-norden), • Abdelhamid Bakhta (@abdelhamidbakhta)
Status	Final
Created	2019-04-13
Description	https://github.com/ethereum/EIPs/blob/master/EIPS/eip-1559.md
Specification	https://github.com/ethereum/EIPs/blob/master/EIPS/eip-1559.md#Specification
Category	Core

Simple Summary

A transaction pricing mechanism that includes a fixed-per-block network fee that is burned and dynamically expands/contracts block sizes to deal with transient congestion.

Abstract

We introduce a new EIP-2718 transaction type, with the format `0x02 || rlp([chain_id, nonce, max_priority_fee_per_gas, max_fee_per_gas, gas_limit, destination, amount, data, access_list, signature_y_parity, signature_r, signature_s])`.

There is a base fee per gas in the protocol, which can move up or down each block according to a formula that is a function of the gas used in the parent block and the gas target (block gas limit divided by elasticity multiplier) of the parent block. The algorithm results in the base fee per gas increasing when blocks are above the gas target, and decreasing when blocks are below the gas target. The base fee per

gas is burned. Transactions specify the maximum fee per gas they are willing to give to miners to incentivize them to include their transaction (aka: priority fee). Transactions also specify the maximum fee per gas they are willing to pay total (aka: max fee), which covers both the priority fee and the block's network fee per gas (aka: base fee). The transaction will always pay the base fee per gas of the block it was included in, and they will pay the priority fee per gas set in the transaction, as long as the combined amount of the two fees doesn't exceed the transaction's maximum fee per gas.

Motivation

Ethereum historically priced transaction fees using a simple auction mechanism, where users send transactions with bids (“gasprices”) and miners choose transactions with the highest bids, and transactions that get included pay the bid that they specify. This leads to several large sources of inefficiency:

- **A mismatch between the volatility of transaction fee levels and the social cost of transactions:** bids to include transactions on mature public blockchains, that have enough users so that blocks are full, tend to be extremely volatile. It's absurd to suggest that the cost incurred by the network from accepting one more transaction into a block actually is 10x more when the cost per gas is 10 nanoeth compared to when the cost per gas is 1 nanoeth; in both cases, it's a difference between 8 million gas and 8.02 million gas.
- **Needless delays for users:** because of the hard per-block gas limit coupled with natural volatility in transaction volume, transactions often wait for several blocks before getting included, but this is socially unproductive; no one significantly gains from the fact that there is no “slack” mechanism that allows one block to be bigger and the next block to be smaller to meet block-by-block differences in demand.
- **Inefficiencies of first-price auctions:** The current approach, where transaction senders publish a transaction with a bid a maximum fee, miners choose the highest-paying transactions, and everyone pays what they bid. This is well-known in mechanism design literature to be highly inefficient, and so complex fee estimation algorithms are required. But even these algorithms often end up not working very well, leading to frequent fee overpayment.
- **Instability of blockchains with no block reward:** In the long run, blockchains, where there is no issuance (including Bitcoin and Zcash) at present, intend to switch to rewarding miners entirely through transaction fees. However, there are known issues with this that likely lead to a lot of instability, incentivizing mining “sister blocks” that steal transaction fees, opening up much stronger selfish mining attack vectors, and more. There is at present no good mitigation for this.

The proposal in this EIP is to start with a base fee amount which is adjusted up and down by the protocol based on how congested the network is. When the network exceeds the target per-block gas usage, the base fee increases slightly and when capacity is below the target, it decreases slightly. Because these base fee changes are constrained, the maximum difference in base fee from block to block is predictable. This then allows wallets to auto-set the gas fees for users in a highly reliable fashion. It is expected that most users will not have to manually adjust gas fees, even in periods of high network activity. For most users, the base fee will be estimated by their wallet and a small priority fee, which compensates miners taking on orphan risk (e.g. 1 nanoeth), will be automatically set. Users can also manually set the transaction max fee to bind their total costs.

An important aspect of this fee system is that miners only get to keep the priority fee. The base fee is always burned (i.e. it is destroyed by the protocol). This ensures that only ETH can ever be used to pay for transactions on Ethereum, cementing the economic value of ETH within the Ethereum platform and reducing risks associated with miner extractable value (MEV). Additionally, this burn counterbalances Ethereum inflation while still giving the block reward and priority fee to miners. Finally, ensuring the miner of a block does not receive the base fee is important because it removes miner's incentive to manipulate the fee in order to extract more fees from users.

From:
<https://www.omgwiki.org/dido/> - **DIDO Wiki**

Permanent link:
https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:xapend:xapend.b_stds:default:ethereum:eip:eip_1559&rev=1653153157

Last update: **2022/05/21 13:12**

