EIP 2930: Optional access lists

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 Optional access lists

Title	Optional access lists
EIP	2930
Requires	• EIP 2718: Typed Transaction Envelope • EIP 2929: Gas cost increases for state access opcodes
Author	• Vitalik Buterin (@vbuterin), • Martin Swende (@holiman)
Status	final
Created	2020-08-29
Description	https://github.com/ethereum/EIPs/blob/master/EIPS/eip-2930.md
Specification	https://github.com/ethereum/EIPs/blob/master/EIPS/eip-2930.md#Specification
Category	Core

Simple Summary

Adds a transaction type that contains an access list, a list of addresses, and storage keys that the transaction plans to access. Accesses outside the list are possible but become more expensive.

Abstract

We introduce a new EIP 2718: Typed Transaction Envelope transaction type, with the format **`0x01 || rlp([chainId, nonce, gasPrice, gasLimit, to, value, data, accessList, signatureYParity, signatureR, signatureS]`**).

The **`accessList`** specifies a list of addresses and storage keys; these addresses and storage keys are added into the accessed_addresses and accessed_storage_keys global sets (introduced in EIP-2929). A gas cost is charged, though at a discount relative to the cost of accessing outside the list.

Motivation

This EIP serves two functions:

1. Mitigates contract breakage risks introduced by EIP-2929, as transactions could pre-specify and pre-pay for the accounts and storage slots that the transaction plans to access; as a result, in the actual execution, the SLOAD and EXT* opcodes would only cost 100 gas: low enough that it would not only prevent breakage due to that EIP but also “unstuck” any contracts that became stuck due to EIP 1884.
2. Introduces the access list format and the logic for handling the format. This logic can later be repurposed for many other purposes, including block-wide witnesses, use in ReGenesis, moving toward static state access over time, and more.