

# Access Keys

In blockchain, using an account refers to using a private key to sign [transactions](#).

NEAR Accounts have the unique feature of holding multiple [public/private key pairs](#), called Access Keys, each with its own set of permissions.

Access Keys are similar to [OAuths](#), enabling you to grant limited access over your account to third-parties.

## Benefits of Access Keys

Since the concept of Access Keys is unique to NEAR it is worth understanding why and how they can be used.

### Using An App Safely

Imagine that you want to play a web3 game consisting of a web-app and a smart contract. You can create a key that only allows you to call specific methods in the game's contract.

You can safely give the key to the game, so it can sign game-related transactions for you without needing to interrupt the gameplay on each transaction.

### Replacing Keys

If you think any of your keys could be compromised, you can simply remove them or swap them with a new one. Just as how you can change your password on any website.

### Implementing Key Recovery

You could implement a key-recovery [contract](#) in your account, and create a "recovery key" for someone you trust. Such a key could only be used to start the recovery.

When needed, that third-party component could trigger the recovery process, helping to create a new master for you.

## Key Types

NEAR implements two types of access keys: FullAccess keys and FunctionCall keys.

### Full Access Keys

As the name suggests, FullAccess keys have full control of an account, similar to having administrator privileges on your operating system.

Particularly, Full Access keys can be used to sign transactions doing [any action](#) in your account's behalf:

1. Create immediate [sub-accounts](#)
2. and [top-level accounts](#)
3. if the account ID's length is at least 32 characters.
4. Delete your account (but not
5. sub-accounts, since they have their keys).
6. Add or remove Access Keys.
7. Deploy a smart contract in the account.
8. Call methods on any contract (yours or others).
9. Transfer NEAR (Ⓝ).

If you hand a FullAccess to someone, they will have total control over the account.

tip You add the first Full Access Key of the account when [the account is created](#).

### Function Call Keys

FunctionCall keys only have permission to call specific methods on a contract (potentially all methods) but do NOT allow to attach NEAR (Ⓝ) to the call.

FunctionCall keys are defined by three attributes:

1. receiver\_id
2. : The contract

3. which the key allows to call. No other contract can be called using this key.
4. method\_names
5. (Optional): The contract's methods
6. the key allows to call. If omitted, all methods can be called.
7. allowance
8. (Optional): The amount of  $\text{NEAR}$
9. allowed to spend on [gas](#)
10. . If omitted, the key can consume UNLIMITED
11.  $\text{NEAR}$  as gas.

Function Call keys main purpose is to be handed to apps, so they can make contract calls in your name.

NEAR simplifies creating and giving FunctionCall keys to dApps by implementing [a sign-in](#) process. Briefly, dApps can ask you to sign-in using the [wallet](#) , which automatically creates and gives a FunctionCall key to the dApp.

With the FunctionCall key, the dApp will then be able to call specific methods in your account's behalf , with a default allowance of 0.25  $\text{NEAR}$  for gas.

If the dApps requests to transfer any amount of tokens with the FunctionCall key, the user will be prompted once more by the wallet to authorize the transaction .

## Locked Accounts

If you remove all keys from an account, then the account will become locked , meaning that no external actor can perform transactions in the account's name.

In practice, this means that only the account's smart contract can transfer assets, create sub-accounts, or update its code.

Locking an account is very useful when one wants to deploy a contract, and let the community be assured that only the contract is in control of the account. [Edit this page](#) Last updated on Jan 31, 2024 by gagdiez Was this page helpful? Yes No

[Previous Address \(Account ID\)](#) [Next Smart Contract](#)