

Durable Transaction Nonces in the Solana CLI

Durable transaction nonces are a mechanism for getting around the typical short lifetime of a transaction's [recent_blockhash](#). They are implemented as a Solana Program, the mechanics of which can be read about in the [proposal](#).

Usage Examples

Full usage details for durable nonce CLI commands can be found in the [CLI reference](#).

Nonce Authority

Authority over a nonce account can optionally be assigned to another account. In doing so the new authority inherits full control over the nonce account from the previous authority, including the account creator. This feature enables the creation of more complex account ownership arrangements and derived account addresses not associated with a keypair. The `--nonce-authority` argument is used to specify this account and is supported by the following commands

- `create-nonce-account`
- `new-nonce`
- `withdraw-from-nonce-account`
- `authorize-nonce-account`

Nonce Account Creation

The durable transaction nonce feature uses an account to store the next nonce value. Durable nonce accounts must be [rent-exempt](#), so need to carry the minimum balance to achieve this.

A nonce account is created by first generating a new keypair, then create the account on chain

- Command

```
solana-keygen new -o nonce-keypair.json solana create-nonce-account nonce-keypair.json 1 * Output
```

2SymGjGV4ksPdpsbaqWFiDoBz8okvtiik4KE9cnMQgRHrRLySSdZ6jrEcpPifW4xUpp4z66XM9d9wM48sA7peG2XL To keep the keypair entirely offline, use the [Paper Wallet](#) keypair generation [instructions](#) instead [Full usage documentation](#)

Querying the Stored Nonce Value

Creating a durable nonce transaction requires passing the stored nonce value as the value to the `--blockhash` argument upon signing and submission. Obtain the presently stored nonce value with

- Command

```
solana nonce nonce-keypair.json * Output
```

8GRipryfxcsxN8mAGjy8zbFo9ezaUsh47TsPzmZbuytU [Full usage documentation](#)

Advancing the Stored Nonce Value

While not typically needed outside a more useful transaction, the stored nonce value can be advanced by

- Command

```
solana new-nonce nonce-keypair.json * Output
```

44jYe1yPKrjuYDmoFTdgPjg8LFpYyh1PFKJqm5SC1PiSyAL8iw1bhadcAX1SL7KDmREEkmHpYvreKoNv6fZgfvUK [Full usage documentation](#)

Display Nonce Account

Inspect a nonce account in a more human friendly format with

- Command

```
solana nonce-account nonce-keypair.json * Output
```

balance: 0.5 SOL minimum balance required: 0.00136416 SOL nonce:
DZar6t2EaCFQTbUP4DHKwZ1wT8gCPW2aRfkVWwhydkBvS [Full usage documentation](#)

Withdraw Funds from a Nonce Account

Withdraw funds from a nonce account with

- Command

```
solana withdraw-from-nonce-account nonce-keypair.json ~/.config/solana/id.json 0.5 * Output
```

3foNy1SBqwXSsfSfTdmYKDuhnVheRnKXpoPySiUDBVeDEs6iMVokgqm7AqfTjbk7QBE8mqomvMUMNQhtdMvFLide Close
a nonce account by withdrawing the full balance [Full usage documentation](#)

Assign a New Authority to a Nonce Account

Reassign the authority of a nonce account after creation with

- Command

```
solana authorize-nonce-account nonce-keypair.json nonce-authority.json * Output
```

3F9cg4zN9wHxLGx4c3cUKmqpej4oa67QbALmChsJbfxTgTffRiL3iUehVhR9wQmWgPua66jPuAYeL1K2pYYjbNoT [Full usage documentation](#)

Other Commands Supporting Durable Nonces

To make use of durable nonces with other CLI subcommands, two arguments must be supported.

- --nonce
- , specifies the account storing the nonce value
- --nonce-authority
- , specifies an optional [nonce authority](#)

The following subcommands have received this treatment so far

- [pay](#)
- [delegate-stake](#)
- [deactivate-stake](#)

Example Pay Using Durable Nonce

Here we demonstrate Alice paying Bob 1 SOL using a durable nonce. The procedure is the same for all subcommands supporting durable nonces

- Create accounts

First we need some accounts for Alice, Alice's nonce and Bob

```
solana-keygen new -o alice.json solana-keygen new -o nonce.json solana-keygen new -o bob.json
```

- Fund Alice's account

Alice will need some funds to create a nonce account and send to Bob. Airdrop her some SOL

```
solana airdrop -k alice.json 1 1 SOL
```

- Create Alice's nonce account

Now Alice needs a nonce account. Create one

Here, no separate [nonce authority](#) is employed, soalice.json has full authority over the nonce account solana create-nonce-account -k alice.json nonce.json 0.1

```
3KPZr96BTsL3hqera9up82KAU462Gz31xjqJ6eHUAjF935Yf8i1kmfEbo6SVbNaACKE5z6gySrNjVRvmS8DcPuwV
```

- A failed first attempt to pay Bob

Alice attempts to pay Bob, but takes too long to sign. The specified blockhash expires and the transaction fails

```
solana transfer -k alice.json --blockhash expiredDTaxfagttWjQweib42b6ZHADSx94Tw8gHx11 bob.json 0.01 [ 2020 -01-02T18:48:28.462911000Z ERROR solana_cli::cli ] lo ( Custom { kind: Other, error: "Transaction \"33gQQaoPc9jWePMvDAeyJpcnSPiGUAdtVg8zREWv4GiKjkcGNufgpcbFyRKRRa25NkgjZySEeKue5rawyeH5TzsV \" failed:
```

None"

```
} ) Error: Io ( Custom { kind: Other, error: "Transaction \"
33gQQaoPc9jWePMvDAeyJpcnSPiGUAdtVg8zREWv4GiKjkcGNufgpcbFyRKRRa25NkgjZySEeKue5rawyeH5TzsV \" failed:
None"
} )
```

- Nonce to the rescue

Alice retries the transaction, this time specifying her nonce account and the blockhash stored there

Remember, `alice.json` is the [nonce authority](#) in this example solana nonce-account `nonce.json` balance: 0.1 SOL minimum balance required: 0.00136416 SOL nonce: `F7vmkY3DTaxfagttWjQweib42b6ZHADSx94Tw8gHx3W7` solana transfer `-k alice.json --blockhash F7vmkY3DTaxfagttWjQweib42b6ZHADSx94Tw8gHx3W7 --nonce nonce.json bob.json 0.01` `HR1368UKHVZyenmH7yVz5sBAijV6XAPeWbEiXEGVYQorRMcoijeNAbzZqEZiH8cDB8tk65ckqeegFjK8dHwNFgQ`

- Success

The transaction succeeds! Bob receives 0.01 SOL from Alice and Alice's stored nonce advances to a new value

`solana balance -k bob.json 0.01 SOL` solana nonce-account `nonce.json` balance: 0.1 SOL minimum balance required: 0.00136416 SOL nonce: `6bjroqDcZgTv6Vavhqf81oBHTv3aMnX19UTB51YhAZnN` [Previous Solana CLI: Deploy a Program](#)
[Next Solana CLI: Offline Transaction Signing](#)