The Dagger (XDAG) cryptocurrency

- Community site: https://xdag.io
- The Main net was launched January 5, 2018 at 22:45 GMT.

Principles:

- Directed acyclic graph (DAG), not blockchain
- Block = transaction = address
- Original idea and implementation
- Mineable, no premine, no ICO
- Mining new money every 64 seconds

Install and run (Linux):

- Install dependencies:

```
$ sudo dnf install git gcc openssl-devel
or
$ sudo apt-get install git gcc libssl-dev
```

- Clone from the git repository:

```
$ git clone https://github.com/XDagger/xdag
```

- Make:

```
$ cd xdag/client
$ make
```

- or automake

```
$ cd xdag/automake
$ autoreconf -if
$ ./configure
$ make
```

- Run, for example, the miner with 2 CPU mining threads, in daemon mode, connected to the pool put.xdag.server.here:13654

```
$ ./xdag -m 2 -d put.xdag.server.here:13654
Enter random characters: [enter]
```

- Already have an account

```
Put your wallet.dat, dnet_key.dat and storage folder in this folder.
Then run below command
$./xdag -m 2 -d put.xdag.server.here:13654
```

- Run terminal connected to the daemon in the same folder:

```
$ ./xdag -i
xdag> help
[see help]
```

- See if you are connected to the pool:

```
xdag> state
[see state]
```

- See your balance:

```
xdag> balance
[balance]
```

- See your address:

```
xdag> account
[address]
```

- Transfer funds to another address:

```
xdag> xfer [amount] [address]
```

Run (Mac OS):

- Install: Download binary file from release page.

Unzip the zip file to what folder you want.

- Run, for example, the miner with 2 CPU mining threads, in daemon mode, connected to the pool put.xdag.server.here:13654

```
$ ./xdag -m 2 -d put.xdag.server.here:13654
Enter random characters: [enter]
```

- Already have an account Put your wallet.dat, dnet_key.dat and storage folder in this folder. Then run below command

```
$./xdag -m 2 -d put.xdag.server.here:13654
```

- Run terminal connected to the daemon in the same folder:

```
$ ./xdag -i
xdag> help
[see help]
```

- See if you are connected to the pool:

```
xdag> state
[see state]
```

- See your balance:

```
xdag> balance
[balance]
```

- See your address:

```
xdag> account
[address]
```

- Transfer funds to another address:

```
xdag> xfer [amount] [address]
```

Main chain idea:

Every block in DAG has up to 15 links to another blocks (inputs and outputs). Block B is *referenced* by another block A if we can reach B from A by following the links. *Chain* is a sequence of blocks each of which is referenced by the previous block. Chain is called *distinct* if every its block belongs to separate 64-seconds interval. *Difficulty*of*block* is 1/hash where *hash* is sha256(sha256(block)) regarded as little-endian number. *Difficulty*of*chain* is sum of difficulties of blocks. *Main*chain_ is the distinct chain with maximum difficulty. Blocks in main chain are called *main*blocks_.

Daggers are mined in every main block. For first 4 years 1024 XDAG are mined in each main block. For second 4 years - 512 XDAG, and so on. So, maximum XDAG supply is approximately power(2,32). Each dagger is equal to power(2,32) cheatoshino. Transaction is *valid* if it is referenced by a main block. Valid transactions are strictly ordered depending on main chain and links order. Double spending is prohibited because only first concurrent transaction (by this order) is applied.

Structure of block:

The on-disk format will change in the future. Consider this the network protocol. Each block has a fixed size of 512 bytes. Block consists of 16 fields each of which has length 32 bytes. Field 0 is header, it consists of 4 quadwords: - transport-layer header - types of all 16 fields, 4 bits for one type - timestamp of the block, in seconds from Unix era * 1024 - block fee in cheatoshi

Types of fields:

1. nonce
2. header
3. transaction input: 24 lower bytes of block hash and 8 bytes of input amount
4. transaction output, structure is the same as input
5. half of block signature; ECDSA number r or s; digest for signature is hash of (block concate public key)
6. half of output signature; only owner of this key can use this block as input
7. public key (x) with even y
8. public key with odd y
9. ... 15. are reserved for future usage.

Transport layer:

The dnet network is used as transport layer. *A new transport layer will come in the future.*

Maintainers:

Evgeniy (XDAG: gKNRtSL1pUaTpzMuPMznKw49ILtP6qX3, BTC:

1Jonano4esJzZvqNtUY6NwfPme3EMpVs7n)
Frozen (XDAG: +L5dzSh1QZv1We3wi8Of31M8eHwQJq4K)
trueserve (rvKaJSbP9DE6sg6XetYtSpaK+2aDbUq8)