3DPRC-2 tokenization standard

Authors: PaulS, Michael Co, Mikhail

Date: 2023, September 29

WHAT IS 3DPRC-2?

3DPRC-2 (3Dpass Request for Comments), proposed by PaulS in September 2023, is a standard p2p protocol for the tokenization of objects operating within “The Ledger of Things” decentralized blockchain platform.

3DPRC-2 is implemented as following components:

1. **Advanced version of “Proof of Scan”** - The protocol is weaved into “The Ledger of Things” PoW component in a way to tackle the user objects authentication along with the ones being mined. The protocol ensures for users to get a complete service always resulting as either the object acceptance (the asset is allowed to be created) or its rejection (copy is found on the db). The network is responsible for the user object authentication as much as for any block on the blockchain irrespective to the actual dollar value attached to;

2. **“0 knowledge proof”** - Every judgement provided by miners about the object authenticity is protected by a secret knowledge of its *HASH ID*** being unavailable for them, until they get the object processed. Every proof is being verified by the majority of the network to make a final decision on whether to accept or reject the block containing the judgement;

3. “**PoScan” and “PoscanAssets” modules (storage and API)**: - The “PoScan” ** and “PoscanAssets”** pallets are integrated into the network runtime providing the access to the network decentralized storage by means of the object tokenization API, which allows for:

   - the user object authentication and its protection from being copied to the extent for the recognition algorithm precision;

   - non-fungible digital asset creation;

- property rights definition and its transfers;
- backed cryptocurrency issuance (fungible tokens backed by the asset).

THE OBJECT CATEGORIES AND RECOGNITION ALGORITHMS

This is apparent, that it does not make any sense to compare objects by its HASH ID, provided they got processed with different recognition algorithms/parameters. However, HASH IDs need to be compared in order to guarantee for users the absence of copies on the blockchain data base.

By means of categorization of the object types, we are setting up some “standard” algorithms (presets) to be available for use within each category. And every preset defines the level of precision, at which the object is going to be recognized.

Initial list of categories is presented as follows:

- 3D objects
  - Algo: grid2d_v3a -s 12 -g 8
- 2D drawings
- Music
- Biometrics
- Radio signals
- Movements
- Texts

The categories and presets are being moderated by the *Asset committee*** vote. The committee members are being assigned by the C*ouncil***.

THE OBJECT ADDITIONAL PROPERTIES

3DPRC-2 allows to endow the object with its additional properties, which will be utilized for the object tokenization. The object properties are classified as follows:

1. class: Relative:

Non-fungible (propIdx: 0) - allows for the object to be tokenized as a non-fungible asset. *MaxSupply=1* rule will be applied to the token created for this property

Share (propIdx: 1) - allows for the tokenization of the object share *(%)*, the MaxSupply value format of which is restricted to *10^x*. So, the *MaxSupply=10^x* rule will be applied to the token created for this property.

2. class: Absolute:

Weight (propIdx: 2) - allows for the tokenization of the object weight, the MaxSupply value is limited to *2^128* (the max value in the system).

Any other measurable object property, which could be presented as an absolute value (ex. Length, Square, Volume, Density, Clarity, Pressure, Time, Speed, Frequency, Bandwidth, Amount of symbols and so forth)

THE OBJECT AUTHENTICATION PROTOCOL

The protocol represents a sequence of actions performing by validators and miners actively participating in the network consensus at the time an object is being submitted by user.

1. Submitting an object

While placing an order, 3dpass user provides the following:

- an object to tokenize: *rock.obj (*in this example we are going to take 3D model in *.obj*** format);

- the object HASH ID (**top10** hashes from the Grid2D** recognition algorithm output; preset: *-a grid2d_v3a -s 12 -g 8 -d 10*; *pass3d*** recognition toolkit is being used as the implementation of Grid2d):

```
admin@admin pass3d % ./target/release/pass3d –s 12 –g 8 –a grid2d_v3a
–d 10 –i rock.obj
"7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0"
"460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7"
"8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e"
"5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4"
"b8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9"
"10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7"
"833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218"
"abdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371"
"97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c"
“db2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee"
```

- Object authentication fee *P3D/Byte*: Let’s take *100 P3D* as an example. This fee will be distributed among the miners and validators taking part in the process (*70 miners /30 validators*). The fee will be charged for each confirmation ordered (*1 confirmation = 1 block*). The fee can be set up by *Council*** vote.

- Storage fee: P3D/Byte (regular network storage fee)

- Number of confirmations: let’s take *6*, for example. The more confirmations the user orders, the more reliable result He gets, especially, when it comes to the network potentially being attacked at the time. This is unlikely to happen, however there is always a possibility for every blockchain network to get through this sort of experience. It is expected that the user is able to follow the gap between Best block and the Block finalized (normally the *gap = 2 blocks*) to estimate the network state, before submitting the order.

There is a simple copy check on the object submit extrinsic (transaction). If there is a copycat on the blockchain discovered, the transaction will fail.

2. Estimating/Anti-SPAM protection

Once having the object submitted and fees paid, Validators (the most reliable nodes of 3Dpass network) from the current GRANDPA validator set start estimating how long would it take for the object to get processed by the network

in average. They will try to process the object and, if successful, provide the time in milliseconds at which they managed to get it done like this:

- Validator 1 - 128
- Validator 2 - 36
- Validator 3 - 32 …

There is a limit of *3 blocks* for the estimation phase to complete.

The validators will have to import the block containing the object and get it processed once again at the next step when miners have added their judgements. Assuming the block target time in 3dpass set up as *60 sec/block*, there is a time frame limit of *10 seconds* every validator must have finished processing within.

In disregard to the reason why a given validator didn't manage the estimation in time *(10 sec*), its vote won't be taken.

On top of that, every “weird” estimation result (vote) will be statistically ruled out and the average processing time - calculated.

For, example:

- *Validator 1 - 1128 - (ruled out)*
- Validator 2 - 36 +
- Validator 3 - 32 +
- Validator 4 - 0.1- *(ruled out)*
- Validator 5 - 24 +
- Validator 6 - 18 + …

The threshold for the object to pass is established at *“2/3 +1”* out of the actual number of validators in the set. This is the same threshold ratio as the one being utilized for the GRANDPA finalization. The main requirement for this procedure to work correctly would be having *“2/3+1”* validators providing true data on the object processing.

30% of the fee paid by the user will distributed among the validators-estimators in equal. Implying, there is about *400 validators* in the set, *30 P3D/400= 0.075 P3D* each validator gets.

It is assumed, that most of the actual validators are motivated enough to keep the network healthy, otherwise the network as a whole would be considered unsafe. Although, the validators are not providing any substantial proof of their work, which could not be taken for *100%* truth by miners taking its turn at the next step.

This is a snapshot example of the object “Estimated” from the storage (short version, the data is presented partially. See more POSCAN API):


```
[
    [
      4
    ]
    {
      state: {
        Estimated: 2,693
      }
      obj:
0x6f200a7620302e313520302e383520302e320a7620302e313520302e383620....
      category: {
        Objects3D: Grid2dLow
      }
      hashes: [
        0x7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0
        0x460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7
        0x8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e
```

```
    0x5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4
    0xb8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9
    0x10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7
    0x833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218
    0xabdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371
    0x97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c
    0xdb2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee
  ]
  whenCreated: 2,690
  whenApproved:
  owner: d1f5KGsoZ3xzB6Ecmv92DPizD1x5eToNHM1CPfSC1Xu4nzecN
  estimators: [
    [
      d1ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSixnc
      36
    ]
    [
      d1asoD7V6hdff4ExvtjRbdVT398Rg8fKEruYsKFS5P9mkT4fy
      32
    ]
  ]
  estOutliers: []
  approvers: []
```

```
      numApprovals: 6
      estRewards: 70,000,000,000,000,000
      authorRewards: 30,000,000,000,000,000
      prop: [
        {
          "propIdx": 0,
          "maxValue": 1
        }
        {
          "propIdx": 1,
          "maxValue": 100000
        }
      ]
    }
  ]
]
```

The object becomes “NotApproved”, if it doesn’t pass the estimation within 3 blocks timeframe limit.

Gathering confirmations/approvals

Once having the order status at “Estimated”, the actual block author, who found new block, is getting privileged to provide the judgement “Approved” about the object. There is no option for miners to provide a negative judgement available.

Any honest block author is expected to do the processing job on the user object. The proof of work is required at this step to be attached to the block. If the proof turns out to be incorrect or fake, the entire block will be rejected by the majority of the network.

There is an additional copy check on the block import for every full 3dpass node. If there is a copycat of the user object on the blockchain discovered, the block will be rejected, either.

In order to get a proof the block author should get the object processed with the same preset as the user actually have, with the only exception for the ***top11*** hashes are to be claimed from *pass3d*:

```
admin@admin pass3d % ./target/release/pass3d -s 12 -g 8 -a grid2d_v3a
-d 11 -i rock.obj
"7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0"
"460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7"
"8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e"
"5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4"
"b8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9"
"10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7"
"833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218"
"abdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371"
"97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c"
"db2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee"
"ab198306dcee347bd84a942ffc9ab06c8458865fee03c5d1b20ab45b807dc1c5"
```

Miner knows nothing about this *11th hash*, until having the object processed. This secret, called *“0 knowledge proof”*, will be leveraged by the network majority while importing the block.

It is totally up to the block author whether or not to trust this collective estimation result coming from validator set. So, there is always a risk for him to loose his block rewards. And there is a benefit to get rewarded on top by user, if the judgement is accepted.

It is assumed, that every miner will rely on himself and make an independent decision about both aspects the object authenticity and the processing time. If any of those fail, the judgement should never be provided.

If the block author decides to skip, the block will not contain the judgement, meaning, there is no confirmation will be gained for the object.

If the block was accepted by the network majority, then *"+1"* confirmation the object gets. Now the next block author takes its turn, and the proof of work will be the ***12th hash*** from the grid2d output:

```
admin@admin pass3d % ./target/release/pass3d -s 12 -g 8 -a grid2d_v3a
-d 12 -i rock.obj
"7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0"
"460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7"
"8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e"
"5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4"
"b8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9"
"10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7"
"833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218"
"abdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371"
"97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c"
"db2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee"
"ab198306dcee347bd84a942ffc9ab06c8458865fee03c5d1b20ab45b807dc1c5"
"82fd9527291ba30cdddeb5786bac083f8092987e379c9433d81a2eb1bb8c447a"
```

Again, the miner knows nothing about this *12th hash*, until having the object processed. If there is no hash below the top11 available, *"null"* will be taken for proof.

This procedure repeats itself, up until the block at which required number of confirmations is reached. In this particular example the number is 6, so the final 6th judgement will contain the HASH ID expanded to ***top16*** hashes:

```
admin@admin pass3d % ./target/release/pass3d -s 12 -g 8 -a grid2d_v3a
-d 16 -i miner/rock.obj
"7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0"
"460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7"
"8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e"
"5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4"
"b8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9"
"10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7"
"833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218"
"abdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371"
"97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c"
"db2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee"
"ab198306dcee347bd84a942ffc9ab06c8458865fee03c5d1b20ab45b807dc1c5"
"82fd9527291ba30cdddeb5786bac083f8092987e379c9433d81a2eb1bb8c447a"
```

"10eac3abc33a75b16c1ca33aaf97db92542a452453265bf5c088dc33be750137"
"fdbf00e59e8b4d7ae44950751694ebeb2bb6ac969e166d4500d9bfbd886d7523"
"2405d8048e0fead22be4aa4394104136e15c0ca578299068c2a9dbf3c7c68b59"
“905f5f8032ff6c956422ac0560431820e2bfa47bb0cf58368fd49dbc1e23e48c"

Once the last confirmation ordered by user is approved by the majority of the network, the object becomes “Approved” and available for further operations with the asset.

The object becomes “NotApproved” *5 blocks* after last confirmation or the block, at which it was “Estimated”, if there is still no new confirmation available.

Every block author provided the network with correct judgement is getting rewarded with its share, which is equal to *70%*fee/n*, where n is the number of confirmations ordered by user. In this example *n=6* and *70% fee = 70 P3D* So, every honest miner gets *11.666666666666 P3D*.

This is a snapshot example of the object approved from the storage (short version, the data is presented partially. See more POSCAN API):


```
[
    [
      4
    ]
    {
      state: {
        Approved: 2,699
      }
      obj: 0x6f200a7620302e313520302e383520302e320a7620302e313520302e383620....
      category: {
        Objects3D: Grid2dLow
      }
```

```
hashes: [
  0x7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0
  0x460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7
  0x8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e
  0x5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4
  0xb8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9
  0x10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7
  0x833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218
  0xabdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371
  0x97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c
  0xdb2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee
]
whenCreated: 2,690
whenApproved: 2,699
owner: d1f5KGsoZ3xzB6Ecmv92DPizD1x5eToNHM1CPfSC1Xu4nzecN
estimators: [
  [
    d1ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSixnc
    36
  ]
  [
    d1asoD7V6hdff4ExvtjRbdVT398Rg8fKEruYsKFS5P9mkT4fy
    32
  ]
]
```

```
      estOutliers: []
      approvers: [
        {
          accountId: d1jygGfK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8KTrgnjs6
          when: 2,695
          proof:
0xab198306dcee347bd84a942ffc9ab06c8458865fee03c5d1b20ab45b807dc1c5
        }
        {
          accountId: d19ouGfK97U9edj69nZzZ9dUf1W1XUA4bYVC2Zre8K9Jgclj
          when: 2,696
          proof:
0x82fd9527291ba30cdddeb5786bac083f8092987e379c9433d81a2eb1bb8c447a
        }
      ]
      numApprovals: 6
      estRewards: 70,000,000,000,000,000
      authorRewards: 30,000,000,000,000,000
      prop: [
        {
          "propIdx": 0,
          "maxValue": 1
        }
        {
          "propIdx": 1,
          "maxValue": 100000
```

```
          }
        ]
      }
    ]
]
```

New algorithm requirements

This protocol is proposed to be a standard for the tokenization of objects corresponding to the categories specified (see more “THE OBJECT CATEGORIES AND RECOGNITION ALGORITHMS”).

- Every recognition algorithm to add must provide “0 knowledge proof” about the object, so that the network can verify and agree upon its authenticity. 3D object authentication procedure utilizing the grid2d recognition algorithm can be used as a reference.
- Every recognition algorithm must be open source and free from license restrictions, which could prevent its distribution for free.
- Every recognition algorithm must be able to get objects processed within the timeframe limit of 10 sec, while running on average full 3dpass node.
- Every recognition algorithm must rely on public data, stored on 3dpass blockchain (objects, HASH IDs, zero knowledge proof).

PROPERTY RIGHTS DEFINITION

One of the most important challenges related to the tokenization of objects all across the entire blockchain world is how could the property rights be initially defined. Simply speaking, how could we verify the actual owner of the object/ asset at the time the object is being submitted by someone claiming they have all the rights required for.

3DPRC-2 proposes a trustworthy procedure for the property rights definition and its correction after any potential changes, which might had happened as an

outcome of any argument or adjudication, any other legal action leading to the rights correction outside of the blockchain storage, which needs to be taken by “The Ledger of Things” for direct order and, finally, got executed.

Let’s classify the property rights into the following aspects:

1. **Common public assets** - Intangible or tangible assets the property rights has expired for (50 years after the author’s death has actually lasted by the time the asset appears on “The Ledger of Things”). For example, it could be a piece of classic music or Venus de Milo statue. Those assets are most commonly priceless and obtained by the humanity as a whole, representing the legacy of human civilization. Everyone is allowed for its replications commercial use (ex, anyone can listen to and make a copy of any piece of classic music without any license fee requirements/restrictions). Although, the original object properties can be identified by the recognition algorithms (its shape, weight, melody, voice, etc.)

2. **Private assets** - Intangible or tangible assets, the property rights of which are established and still valid by the time the asset appears on “The Ledger of Things”. The asset is obtained by either a person/group of persons or an organization. In terms of the world wide intellectual property legislation, the property rights have been valid since the first reliable publication made by its author. Tangible assets are usually published on the government registries (real state, vehicles, etc).

3. **Unpublished assets** - Intangible or tangible assets, the property rights of which are NOT established by the time the asset appears on “The Ledger of Things”. For example, it could have been something like new 3D statue or new pop music track, anything unpublished yet.

Common public assets

3DPRC-2 will always allow for anyone to issue crypto currencies (fungible and non-fungible) backed by any common public asset stored on “The Ledger of Things”, due to the fact that there is no restrictions could be applied.

Private assets

3DPRC-2 will always guarantee for anyone, who is the actual object owner, to establish their property rights towards the asset stored on “The Ledger of Things”, provided the ownership has been verified by means of special property rights extension(option) of The Object authentication protocol (see more THE OBJECT AUTHENTICATION PROTOCOL).

The extended option of the protocol will use a list of trustworthy external resourced, which data can be considered reliable enough for making decisions about the on-chain private assets, to verify the asset owner.

For example:

- https://www.wipo.int/ - WIPO (World Intellectual Property Organization)**

The list of resources can be moderated by the *Asset committee*** vote.

The ownership verification protocol:

1. By means of leveraging *pass3d*** recognition toolkit, the object owner must have created the multi-object HASH ID, the seed of which would be presented as a combination of the following components:

   A. The object tokenized (ex. 3D model in *.obj* format)

   B. The owner biometric data (optional)

2. By means of using *3dpass wallet*** signature, the object owner must have composed a message in the format as follows:

```
-- Start message --

object:
0x6f200a7620302e313520302e383520302e320a7620302e313520302e383620...
. category: Objects3D: Grid2dLow hashes:
7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0
460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7
8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e
5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4
b8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9
10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7
```

```
833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218
abdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371
97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c
db2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee

-- End message --


-- Start P3D wallet signature --

0xfa020a57c5ef765a65610d5985b7bb391f95abf79adb9eaa598e751d5f9ea02b0
0673d879f826f347565473db512ce472711c6f7742bf546df3723ec610f928c

-- End P3D wallet signature --


-- Start public key --

d1ELrwRkSw5bXwg7NVSbpWMdjdc7oBF32ESx2cZpCUfqXZwhe

-- End public key --
```

The message must be signed with the owner P3D account (in the example above d1ELrwRkSw5bXwg7NVSbpWMdjdc7oBF32ESx2cZpCUfqXZwhe).

3. The object owner must have published the message on the db of any reputable resource from the actual list available on the network storage (for example https://www.wipo.int/ - WIPO). The message must be accessible via the open public API (as an additional properties, for example), so that the validators can verify the data from all around the globe automatically.

4. Once having all these 3 steps above done, the object owner submits the authentication order (see more THE OBJECT AUTHENTICATION PROTOCOL) with the option *'ownership.proof'*. A public link/reference to the object on the external data base must be provided as the proof.

5. The object authentication protocol will be executed as it is described in the THE OBJECT AUTHENTICATION PROTOCOL chapter, but with the couple additional rules. The rules are:

- The network nodes will try to verify the message via the API (external worker (oracle) will be used on the node side). The judgement “Approved” will be accepted if the message will be verified on block import, otherwise the block is going to be rejected;
- If any copycat of the object has been discovered, the actual owner/owners will be applied.

Unpublished assets

The combination of such aspects as the timestamp, copy protection, data change protection make “The Ledger of Things” one of the most reliable decentralized data bases to make the first publication for any new object created. It would be enough for its owner to get the object through the authentication procedure and claim they have all the rights required.

BACKED CURRENCY ISSUANCE

By means of dealing with the object properties it is possible to turn the object into either *Fungible* or *Non-fungible* asset, depending on the purpose of its tokenization.

For example, the tokenization of the object *Share* (as well as such properties as *Weight*, *Square*, *Volume*, *Length*, etc) will always stand for its collective ownership or *ICO (Initial Coin Offering)*. These properties will always be tokenized as *Fungible assets*, the *MaxSupply* of which is limited to the property value attached to the object. For example, if the object weight is *1000 gram*, then the token *MaxSupply=1000* limit will be set up for the token created (you won't be able to issue more than *1000 minimum indivisible units*). While transferring tokens, the object share is being transferred accordingly.

There is a specific property named *Non-Fungible*, which is leveraged to get the object tokenized as a *Non-fungible* asset. If chosen, the *MaxSupply = 1* limit will be applied to the token created. Whereas *1* is the *minimum indivisible uni*t of The Ledger of Things. By means of transferring this unit, the ownership of the entire object is being transferred.

ASSET COLLECTIVE MANAGEMENT

Is being worked on…

THE OBJECT TOKENIZATION API

The object tokenization API is provided by both ***poScan*** pallet and ***poscanAssets*** pallet integrated with one another to ensure the backed assets created to be in compliance with the object properties. ***poscanAssets*** is responsible for assets actions (creation, minting, transfers, etc) will be in control with ***poScan*** pallet protecting such aspects as: the object authenticity, backed asset *MaxSupply* limits and ownership transfers.

The poScan pallet API is presented as following methods:

```
1. putObject(

        category,

        obj,

        numApprovals,

        hashes

   )
```

This method allows to put an object into the poScan storage. The object authentication procedure will be triggered, as well. Either, the object will be **Approved** or **NotApproved** as a result (see more THE OBJECT AUTHENTICATION PROTOCOL).

If “Approved”, the object will be allowed for any further operation with the asset (property rights transfers, backed currency issuance, etc), and the copy protection will be applied. The object will be available on the network storage with all the authentication history data attached.

“NotApproved” keeps the object and all the authentication history data available on the network storage, however, all further operations will be prohibited, and the copy protection will not be applied.

- category - the object category

    Objects3D,

        Grid2dLow - Grid2d algorithm (low precision), preset: -s 12 -g 8 -a grid2d_v3a (see more pass3d recognition toolkit and Grid2D algo parameters: https://github.com/3Dpass/pass3d)

        Grid2dHigh - Grid2d algorithm (high precision)

    Drawings2D,

    Music,

    Biometrics,

    Biometrics,

    Movements,

    Texts

- obj - the object to authenticate (ex. 3D model in .obj format)
- numApprovals: u8 - the number of confirmations in blocks to order (1-255). The object authentication loop will repeat itself as much times as it is requested.
- hashes: Option<Vec<H256>> - the hashes (10 at max) of the object HASH ID (ex. the top10 hashes Grid2D output).

2. setAlgoTime(

        algoTime: u32

)

This method allows to set up the time frame limit in seconds for the objets to get processed. This value can be set up by the *Council*** vote.

- `algoTime: u32 - the timeframe limit in seconds (10 seconds is set up by default)`

3. `setFeePerByte(`

   `fee`

   `)`

- `fee: u64 - P3D/Byte for user to pay`

This method allows to set up the object authentication fee. This value can be set up by the *Council*** vote.

4. `approve(`

   `author,`

   `objIdx,`

   `proof`

   `)`

This method is utilized by new block authors (miners) to provide their judgement on objects “Estimated” (after the estimation procedure, performed by Validators, is completed successfully). If the estimation procedure was not successful or fully complete (ex, the object is still at “Estimating” or “Created”), the judgement and the block will be rejected by the majority of the network.

- `objIdx: u32 - the object index on the poScan storage`
- `proof: Option<H256> - zero-knowledge proof of work hash (see more THE OBJECT AUTHENTICATION PROTOCOL).`

RPC API methods to fetch the data from the storage:

1. objects(

        u32

)

This method allows to get the objects and its authentication history from the storage.

- Option<u32> - the object index value (optional)

The data example:

```
[
    [
      4
    ]
    {
      state: {
        Approved: 2,699
      }
      obj: 0x6f200a7620302e313520302e383520302e320a7620302e313520302e383620....
      category: {
        Objects3D: Grid2dLow
      }
      hashes: [
        0x7fbe267b208d0ab9c34fa184d4593d0ae39d5bb56852ecc633cb1f6d9eb5aae0
        0x460c81f58510e211d162d458459f81575e630817c1b87bc3c0d2c3eec3ae68b7
        0x8047c7a4d2a75ce466c9510edb37ac1e98826ef383ab5d4c860b5010e42faa6e
```

```
    0x5001723ef6d85862733d25765f0700e381cc3a1f6d6b7dbb4efb8aeccc4615d4
    0xb8efd617a07099edcdbf2596d10787a6e16a4c59a5e36e46adedc1dd2424b8c9
    0x10c8d4984cf840845a7d7785871c70790bf1c55e11a3da54a41d3f21be5a7ea7
    0x833a2fc1f26ab9eb9bcc5b4e668f5d6bb91d6f87fb0798f99ce6bb4730dfc218
    0xabdedd28e03147d94e0d054d0bf24781bac01fdd81401cd3fb226b5a6e1a5371
    0x97c538a99641202385572df245d9cfb300f895a6c94e7caf074023a5ead1c62c
    0xdb2453f53270c9003a63b4d643d9bcc991ad7630d70a0d1511781f74a7a00fee
  ]
  whenCreated: 2,690
  whenApproved: 2,699
  owner: d1f5KGsoZ3xzB6Ecmv92DPizD1x5eToNHM1CPfSC1Xu4nzecN
  estimators: [
    [
      d1ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSixnc
      36
    ]
    [
      d1asoD7V6hdff4ExvtjRbdVT398Rg8fKEruYsKFS5P9mkT4fy
      32
    ]
  ]
  estOutliers: []
  approvers: [
    {
      accountId: d1jygGfK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8KTrgnjs6
```

```
          when: 2,695
          proof:
0xab198306dcee347bd84a942ffc9ab06c8458865fee03c5d1b20ab45b807dc1c5
        }
        {
          accountId: d19ouGfK97U9edj69nZzZ9dUf1W1XUA4bYVC2Zre8K9Jgclj
          when: 2,696
          proof:
0x82fd9527291ba30cdddeb5786bac083f8092987e379c9433d81a2eb1bb8c447a
        }
      ]
      numApprovals: 6
      estRewards: 70,000,000,000,000,000
      authorRewards: 30,000,000,000,000,000
      prop: [
        {
          "propIdx": 0,
          "maxValue": 1
        }
        {
          "propIdx": 1,
          "maxValue": 100000
        }
      ]
    }
  ]
```

]

– - -

4 - the object index on the db

state: {Approved: 2,699} - current status of the authentication process (ex. Estimated: 2,693 - the block at which the estimation finished)

obj: - the object submitted by user (ex. 3D model in .obj format)

category: {Objects3D: Grid2dLow} - the object category and the algorithm preset used for its authentication

hashes:[] - the HASH ID submitted by user (ex. the top 10 hashes from Grid2d output)

whenCreated: 2,690 - the block number the object was created at

whenApproved: 2,699 - the block number the object was approved at

owner: <P3D address> - the object owner (initially, this is a P3D account the object was submitted with)

estimators: [P3D address, time in mSec] - the list of Validators (P3D addresses), who voted for the object to pass the estimation, the object processing time provided

estOutliers:[] - the validators ruled out, due to the weird processing time

approvers: [{accountId:<P3D address>; when: 2,696; proof: <0knowledge proof of work hash>}] - the list of block authors provided their judgement on the object authenticity

numApprovals: 6 - the number of confirmations ordered by user

estRewards: 70,000,000,000,000,000 - the validator share of rewards (in min indivisible units “Crumbs”, 1 Crumb = 0.0000000000001 P3D), which is to be distributed among the validators(estimators)

authorRewards: 30,000,000,000,000,000 - the block author share of rewards (in min indivisible units “Crumbs”), which is to be distributed among the the miners (approvers)

prop: [{“propIdx": 0, "maxValue": 1}] - the list of the object properties:

- “**propIdx**": **0** - the property index id on the poScan pallet storage (“propIdx": 0 is Non-fungible). See more BACKED CURRENCY ISSUANCE.

- "**maxValue**": **1** - the max token supply limit set up for the tokenization of this property. See more BACKED CURRENCY ISSUANCE.

2. poscan_getPoscanObject

The objects are getting compressed with zip, before they put on the storage. This method allows to get the object and its history without compression (unzipped).

```
curl -H "Content-Type: application/json" -d

'{"id":1,

"jsonrpc":"2.0",

"method": "poscan_getPoscanObject",

"params": [0]

}' http://localhost:9933/
```

where the [0] is the object index value.

The output provides the “result” parameter containing the *.obj* file content and all the object history previously described:


```
{"jsonrpc":"2.0","result":{"state":{"Approved":101},"obj":
[111,32,10,118,32,48,46,48,57,32,48,46,56,50,32,45,48,46,48,49
,10,118,32,48,46,48,56,32,48,46,56,49,32,48,46,48,51...],"comp
ressed_with":null,"category":
{"Objects3D":"Grid2dLow"},"hashes":
["0x2c998a9919725d59e477fcb823f811e1ac5806722653906f5b5d7ec933
ef6bdf","0x80af29c76bf63b639efd8fe8ce368a27266d538de0a47a39725
bc0bb0e13a865","0xb46a7668a64c09b19ff9f42e9b68d78268f11715a35c
a125ddf48bceee5097e4","0xd9d8dbf4258bc7de9c7de7ac5d5dbec806bcb
04ca995fd13482390f047003bc","0x8d1b99511af0a0c6cb43a7546b75fd
7d7a247cbd3e4a6259ac0b28cff67172a2","0x24aad89b485f3e84c4c37e1
```

ba26355c577879123bb02fb010f576ac352ce88b1","0x0691088a9b80ca19
9efb27a06a043a35500de3807d557575901305 47f5bb24f7","0xfaf0c5025
b9573c0a06ce7f0c7e3737717005f92866f0f25b45b57cb400b90c2","0x76
c10fcff1dfda6c3ee665396f61543a26bdd9f73bf4b656f116495aecc9f53b
","0xe465088c87c5c8cf2f33436ee0152107439bdb97335920abeff19d1c3
eb6abf7"],"when_created":91,"when_approved":101,"owner":"d7f5K
GsoZ3xzB6Ecmv92DPizD1x5eToNHM1CPfSC1Xu4nzecN","estimators":
[["d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSixnc",53],
["d7asoD7V6hdff4ExvtjRbdVT398Rg8fKEruYsKFS5P9mkT4fy",5]],"est_
outliers":[],"approvers":
[{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSi
xnc","when":96,"proof":"0x93b64823cb53e8c08b1ddf1b30bed611b11e
0912b3b28aa3e122b2fb5d418be8"},
{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSix
nc","when":97,"proof":"0xcff4f0b718ee26934fa6a7739e2111d307e2e
7be491047277160f92bd5dee1fb"},
{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSix
nc","when":98,"proof":"0xa0de6a64564f85e481fe8fbea5f8c6ddf13b3
e7e66c00254f27a290407d1d99b"},
{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSix
nc","when":99,"proof":"0x961d9110c11ceb62c447edb8576fc8721550a
829814e8f69e1732f5e101e5dd4"},
{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSix
nc","when":100,"proof":"0x9dec30de83339dffa02093cf5c2d3e110c47
e2f9dc05a237f73af8cae6b89b0f"},
{"account_id":"d7ePg4fK97U913xAnZzZ9dUf1W1XUA4bYVC2Zre8K9vjSix
nc","when":101,"proof":"0x145f0fd64c2865e05f5a727d6dc316ac6164
111a785b5bba8ff4fddec1c03171"}],"num_approvals":6,"est_rewards
":70000000000000000,"author_rewards":30000000000000000}

The poscanAssets pallet API is presented as following methods:

This method, provided by the ***poscanAssets*** module, allows for the object owner to create a backed currency (a token backed by the object property). Only one of the object properties is allowed to be tokenized, as long as the object is Approved at the authentication stage (see more *poscan.putObject*).

```
1. create (
    id,
    admin,
    minBalance,
    objDetails
)
```

id: Compact<u32> - the index id for the asset

admin: AccountId - the admin P3D account

minBalance: u128 - min balance in tokens to keep any account alive (accounts going below min balance are going to be removed)

objDetails:

- **objIdx: u32** - the object index id on the poScan module storage

- **propIdx: u32** - the property index id on the poScan module storage

- **maxSupply: u128** - MaxSupply limit in tokens, which is going to be apply to the asset. Must not exceed the object property MaxValue (see more *poscan.putObject*).

The *poscanAsset* logic gets the asset bounded to the user object and thus guarantees to have the token *MaxSupply to be in* compliance with the limit set up by the property value.

```
2. setMetadata(
        id,
        name,
        symbol,
        decimals
```

)

Setting up the asset metadata:

id: Compact<u32> - the index id of the asset, actually existing on the poscanAssets module storage, you are about to set up metadata for

name: Bytes - the asset name (ex. "My very expensive diamond shares")

symbol: Bytes - the asset symbol (ex. XYZ - 1000 XYZ)

decimals: u8 - the number of decimals applied to the asset (ex. decimals: 4 will set up min indivisible unit at 0.0001 XYZ)

3. mint(

   id,

   amount

)

Once having the asset created and set up its metadata, it is possible to use this method to mint some tokens. *MaxSupply* limit the asset has been created with cannot be exceeded.

id: Compact<u32> - the index id of the asset, actually existing on the poscanAssets module storage, you are about to mint tokens for

amount: Compact<u128> - amount to mint

4. transfer(

       id,

       target,

       amount

)

This method allows to transfer minted tokens from one account to another. Some P3D is required to cover the Ledger of Things transaction fee.

id: Compact<u32> - the index id of the asset, actually created on the poscanAssets module storage

target: AccountId - P3D account to receive the transfer

amount: Compact<u128> - amount to transfer

REFERENCE**

- *Grid2d* recognition algorithm: https://3dpass.org/grid2d
- HASH ID: The object identity comes out as a result of the object recognition (https://3dpass.org/features#3drecognition-hash-id)
- *pass3d* recognition toolkit: https://github.com/3Dpass/pass3d
- *.obj* format: https://en.wikipedia.org/wiki/Wavefront_.obj_files
- *PoScan* module: https://github.com/3Dpass/3DP/tree/main/pallets/poscan
- *PoscanAssets* module: https://github.com/3Dpass/3DP/tree/main/pallets/poscan-assets
- *Asset committee:* A group of reputable members presented as 3dpass network accounts elected by *Council*** vote and eligible to moderate the assets-related content available, such as:
    1. Asset categories and algorithms;
    2. The list of external resources for the verification of ownership.
- *Council*: A group of reputable members presented as 3dpass network accounts elected by the majority of P3D holders (https://3dpass.org/governance#council)
- WIPO: https://en.wikipedia.org/wiki/World_Intellectual_Property_Organization

- 3dpass wallet: https://wallet.3dpass.org