arc: 000CS title: Algorand Standard Asset Parameters Conventions for CO2e Tokens

status: Draft

Algorand Standard Asset Parameters Conventions for CO2e Tokens

Summary

This document introduce conventions for the parameters of Algorand Standard Assets (ASAs) CO2e Tokens (CSTs).

Abstract

The goal of these conventions is to make it simpler for block explorers, wallets, exchanges, marketplaces, and more generally, client software to display the properties of a given CO2e ASA.

Specification

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in RFC-2119.

> Comments like this are non-normative.

An ARC-CS ASA has an associated JSON Metadata file, formatted as specified below, that is stored off-chain.

ASA Parameters Conventions

The ASA parameters should follow the following conventions:

- *Unit Name* (`un`): no restriction but **SHOULD** be related to the name in the JSON Metadata file
- *Asset Name* (`an`): **MUST** be:
  - (**NOT RECOMMENDED**) either exactly `arc000CS` (without any space)
  - (**NOT RECOMMENDED**) or `<name>@arc000CS`, where `<name>` **SHOULD** be closely related to the name in the JSON Metadata file:
    - If the resulting asset name can fit the *Asset Name* field, then `<name>` **SHOULD** be equal to the name in the JSON Metadata file.
    - If the resulting asset name cannot fit the *Asset Name* field, then `<name>` **SHOULD** be a reasonable shorten version of the name in the JSON Metadata file.
  - (**RECOMMENDED**) or `<name>` where `<name>` is defined as above. In this case, the Asset URL **MUST** ends with `#arc000CS`.
- *Asset URL* (`au`): a URI pointing to a JSON Metadata file.
  - This URI as well as any URI in the JSON Metadata file:
    - **SHOULD** be persistent and allow to download the JSON Metadata file forever.
    - **MAY** contain the string `{id}`. If `{id}` exists in the URI, clients **MUST** replace this with the asset ID in decimal form. The rules below applies after such a replacement.
    - **MUST** follow RFC-3986 and **MUST NOT** contain any whitespace character
    - **SHOULD** use one of the following URI schemes (for compatibility and security): *https* and *ipfs*:
      - When the file is stored on IPFS, the `ipfs://...` URI **SHOULD** be used. IPFS Gateway URI (such as `https://ipfs.io/ipfs/...`) **SHOULD NOT** be used.
    - **SHOULD NOT** use the following URI scheme: *http* (due to security concerns).
    - **MUST** be such that the returned resource includes the CORS header

```
Access-Control-Allow-Origin: *
```

if the URI scheme is *https*

> This requirement is to ensure that client JavaScript can load all resources pointed by *https* URIs inside an ARC-000CS ASA.

  - **MAY** be a relative URI when inside the JSON Metadata file. In that case, the relative URI is relative to the Asset URL. The Asset URL **SHALL NOT** be relative. Relative URI **MUST** not contain the character `:`. Clients **MUST** consider a URI as relative if and only if it does not contain the character `:`.
- If the Asset Name is neither `arc000CS` nor of the form `<name>@arc000CS`, then the Asset URL **MUST** end with `#arc000CS`.
- If the Asset URL ends with `#arc000CS`, clients **MUST** remove `#arc000CS` when linking to the URL. When displaying the URL, they **MAY** display `#arc000CS` in a different style (e.g., a lighter color).
- If the Asset URL ends with `#arc000CS`, the full URL with `#arc000CS` **SHOULD** be valid and point to the same resource as the URL without `#arc000CS`.
  > This recommendation is to ensure backward compatiblity with wallets that do not support ARC-000CS.

- *Asset Metadata Hash* (`am`):
  - If the JSON Metadata file specifies extra metadata `e` (property `extra_metadata`), then `am` is defined as:

    ```
    am = SHA-512/256("arc000CS/am" || SHA-512/256("arc000CS/amj" || content of JSON Metadata file) || e)
    ```

    where `||` denotes concatenation and SHA-512/256 is defined in NIST FIPS 180-4. The above definition of `am` **MUST** be used when the property `extra_metadata` is specified, even if its value `e` is the empty string. Python code to compute the hash and a full example are provided below (see "Sample with Extra Metadata").

    > Extra metadata can be used to store data about the asset that needs to be accessed from a smart contract. The smart contract would not be able to directly read the metadata. But, if provided with the hash of the JSON Metadata file and with the extra metadata `e`, the smart contract can check that `e` is indeed valid.

  - If the JSON Metadata file does not specify the property `extra_metadata`, then `am` is defined as the SHA-256 digest of the JSON Metadata file as a 32-byte string (as defined in NIST FIPS 180-4)

There are no requirements regarding the manager account of the ASA, or its the reserve account, freeze account, or clawback account.

> Clients recognize ARC-000CS ASAs by looking at the Asset Name and Asset URL. If the Asset Name is `arc000CS` or ends with `@arc000CS`, or if the Asset URL ends with `#arc000CS`, the ASA is to be considered an ARC-000CS ASA.

JSON Metadata File Schema

> The JSON Medata File schema follow the Ethereum Improvement Proposal ERC-1155 Metadata URI JSON Schema with the following main differences:
>
> - Support for integrity fields for any file pointed by any URI field as well as for localized JSON Metadata files.
> - Support for mimetype fields for any file pointed by any URI field.
> - Support for extra metadata that is hashed as part of the Asset Metadata Hash (`am`) of the ASA.
> - Adding the fields `external_url`, `background_color`, `animation_url` used by OpenSea metadata format.

Similarly to ERC-1155, the URI does support ID substitution. If the URI contains `{id}`, clients **MUST** substitute it by the asset ID in *decimal*.

> Contrary to ERC-1155, the ID is represented in decimal (instead of hexadecimal) to match what current APIs and block explorers use on the Algorand blockchain.

The JSON Metadata schema is as follows:

```
{
    "title": "Token Metadata",
    "type": "object",
    "properties": {
        "name": {
            "type": "string",
            "description": "Identifies the afforestation or reforestation asset to which this token
```

```
represents"
        },
        "description": {
            "type": "string",
            "description": "Describes the forestation asset to which this token represents"
        },
        "satellite_image": {
            "type": "string",
            "description": "A URI pointing to a file with MIME type image/* representing the area of interest
regarding the assets forestation area to which this token represents."
        },
        "satellite_image_integrity": {
            "type": "string",
            "description": "The SHA-256 digest of the file pointed by the URI image. The field value is a
single SHA-256 integrity metadata as defined in the W3C subresource integrity specification
(https://w3c.github.io/webappsec-subresource-integrity)."
        },
        "satellite_image_mimetype": {
            "type": "string",
            "description": "The MIME type of the file pointed by the URI external_url. It is expected to be
'image/tif' in almost all cases."
        },
        "external_url": {
            "type": "string",
            "description": "A URI pointing to an external website presenting the asset."
        },
        "external_url_integrity": {
            "type": "string",
            "description": "The SHA-256 digest of the file pointed by the URI external_url. The field value
is a single SHA-256 integrity metadata as defined in the W3C subresource integrity specification
(https://w3c.github.io/webappsec-subresource-integrity)."
        },
        "external_url_mimetype": {
            "type": "string",
            "description": "The MIME type of the file pointed by the URI external_url. It is expected to be
'text/html' in almost all cases."
        },
        "forestation_attributes": {
            "type": "object",
            "description": "Specific attributes describing the forestation project. Values may be strings,
numbers, object or arrays.",
            "properties": {
                "central-coordinates": {
                    "type": "array",
                    "value": ["x-coordinate", "y-coordinate"],
                    "description": "X and Y coordinate of the center of the area of interest for the
forestation project. Represents the ability to determine the forestation location in conjunction with the
satellite imagery.",
                },
                "country": {
                    "type": "string",
                    "description": "Country name of the forestation projects location."
                },
                "county": {
                    "type": "string",
                    "description": "County name of the forestation projects location."
                },
                "project_id": {
                    "type": "integer",
                    "description": "Name of the project for identifying corresponding tokens."
                },
                "projected_carbon_stock": {
                    "type": "float",
                    "description": "Overall projected amount of additionally generated carbon stock over the
project lifetime in tonnes CO2e."
                },
            },
        },
        "extra_metadata": {
            "type": "string",
            "description": "Extra metadata in base64. If the field is specified (even if it is an empty
string) the asset metadata (am) of the ASA is computed differently than if it is not specified."
        },
    }
}
```

All the fields are **OPTIONAL**. But if provided, they **MUST** match the description in the JSON schema.

URI fields (`satellite_image`, `external_url`, `animation_url`, and `localization.uri`) in the JSON Metadata file are defined similarly as the Asset URL parameter `au`. However, contrary to the Asset URL, they **MAY** be relative (to the Asset URL). See Asset URL above.

Integrity Fields

Compared to ERC-1155, the JSON Metadata schema allows to indicate digests of the files pointed by any URI field. This is to ensure the integrity of all the files referenced by the ASA. Concretly, every URI field `xxx` is allowed to have an optional associated field `xxx_integrity` that specifies the digest of the file pointed by the URI.

The digests are represented as a single SHA-256 integrity metadata as defined in the W3C subresource integrity specification. Details on how to generate those digests can be found on the MDN Web Docs (where `sha384` or `384` are to be replaced by `sha256` and `256` respectively as only SHA-256 is supported by this ARC).

It is **RECOMMENDED** to specify all the `xxx_integrity` fields of all the `xxx` URI fields, except for `external_url_integrity` when it points to a potentially mutable website.

Any field with a name ending with `_integrity` **MUST** match a corresponding field containing a URI to a file with a matching digest. For example, if the field `hello_integrity` is specified, the field `hello` **MUST** exist and **MUST** be a URI pointing to a file with a digest equal to the digest specified by `hello_integrity`.

MIME Type Files

Compared to ERC-1155, the JSON Metadata schema allows to indicate the MIME type of the files pointed by any URI field. This is to allow clients to display appropriately the resource without having to first query it to find out the MIME type. Concretly, every URI field `xxx` is allowed to have an optional associated field `xxx_integrity` that specifies the digest of the file pointed by the URI.

It is **STRONGLY RECOMMENDED** to specify all the `xxx_mimetype` fields of all the `xxx` URI fields, except for `external_url_mimetype` when it points to a website. If the MIME type is not specified, clients **MAY** guess the MIME type from the file extension or **MAY** decide not to display the asset at all.

Clients **MUST NOT** rely on the `xxx_mimetype` fields from a security perspective and **MUST NOT** break or fail if the fields are incorrect (beyond not displaying the asset image or animation correctly). In particular, clients **MUST** take all necessary security measures to protect users against remote code execution or cross-site scripting attacks, even when the MIME type looks innocuous (like `image/png`).

> The above restriction is to protect clients and users against malformed or malicious ARC-3.

Any field with a name ending with `_mimetype` **MUST** match a corresponding field containing a URI to a file with a matching digest. For example, if the field `hello_mimetype` is specified, the field `hello` **MUST** exist and **MUST** be a URI pointing to a file with a digest equal to the digest specified by `hello_mimetype`.

Examples

Basic Example

An example of an ARC-3 JSON Metadata file for a song follows. The properties array proposes some **SUGGESTED** formatting for token-specific display properties and metadata.

```
{
    "name": "Stolberg",
    "description": "Reforestation of a specific amount of forest",
    "satellite_image": "https://carbonstack.de/stolberg.png",
    "satellite_image_integrity": "sha256-47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=",
    "satellite_image_mimetype": "image/png",
    "external_url": "https://carbonstack.de/projects/stolberg",
    "forestation_attributes": {
        "property": {
            "central-coordinates": "[53.1, 11.0]",
            "country": "Germany",
            "county": "Lower Saxony",
            "project_id": "1",
            "projected_carbon_stock": "1205.0"
        }
    }
}
```

An example of possible ASA parameters would be:

- *Asset Unit*: tonnes CO2e
- *Asset Name*: `Stolberg`
- *Asset URL*: `https://example.com/forestation#arc0CS` or `https://carbonstack.de/projects/stolberg#arc0CS`
- *Metadata Hash*: the 32 bytes of the SHA-256 digest of the above JSON file
- *Total Number of Units*: 100
- *Number of Digits after the Decimal Point*: 2

> IPFS urls of the form `ipfs://QmWS1VAdMD353A6SDk9wNyvkT14kyCiZrNDYAad4w1tKqT#arc3` may be used too but may cause issue with clients that do not support ARC-000CS and that do not handle fragments in IPFS URLs.

Example with Relative URI and IPFS

> When using IPFS, it is convenient to bundle the JSON Metadata file with other files references by the JSON Metadata file. In this case, because of circularity, it is necessary to use relative URI

An example of an ARC-000CS JSON Metadata file using IPFS and relative URI is provided below:

```
{
    "name": "Stolberg",
    "description": "Reforestation of a specific amount of forest",
    "satellite_image": "https://carbonstack.de/stolberg.png",
    "satellite_image_integrity": "sha256-47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=",
    "satellite_image_mimetype": "image/png",
    "external_url": "https://carbonstack.de/projects/stolberg",

}
```

If the Asset URL is `ipfs://QmWS1VAdMD353A6SDk9wNyvkT14kyCiZrNDYAad4w1tKqT/metadata.json`:

- the `satellite_image` URI is `ipfs://QmWS1VAdMD353A6SDk9wNyvkT14kyCiZrNDYAad4w1tKqT/mysong.png`.

Rationale

These conventions are heavily based on Ethereum Improvement Proposal ERC-1155 Metadata URI JSON Schema to facilitate interoperability.

The main differences are highlighted below:

- Asset Name and Asset Unit can be optionally specified in the ASA parameters. This is to allow wallets that are not aware of ARC-3 or that are not able to retrieve the JSON file to still display meaningful information.
- A digest of the JSON Metadata file is included in the ASA parameters to ensure integrity of this file. This is redundant with the URI when IPFS is used. But this is important to ensure the integrity of the JSON file when IPFS is not used.
- Similarly, the JSON Metadata schema is changed to allow to specify the SHA-256 digests of the localized versions as well as the SHA-256 digests of any file pointed by a URI property.
- MIME type fields are added to help clients know how to display the files pointed by URI.
- When extra metadata are provided, the Asset Metadata Hash parameter is computed using SHA-512/256 with prefix for proper domain separation. SHA-512/256 is the hash function used in Algorand in general (see the list of prefixes in https://github.com/algorand/go-algorand/blob/master/protocol/hash.go). Domain separation is especially important in this case to avoid mixing hash of the JSON Metadata file with extra metadata. However, since SHA-512/256 is less common and since not every tool or library allows to compute SHA-512/256, when no extra metadata is specified, SHA-256 is used instead.
- Support for relative URI is added to allow storing both the JSON Metadata files and the files it refers to in the same IPFS directory.

Valid JSON Metadata files for ERC-1155 are valid JSON Metadata files for ARC-3. However, it is highly recommended that users always include the additional RECOMMENDED fields, such as the integrity fields.

The asset name is either `arc3` or suffixed by `@arc3` to allow client software to know when an asset follows the conventions.

Copyright