Source Code Deposit Walkthrough

Morane Gruenpeter

Metadata specialist
Software Heritage
morane@softwareheritage.org

04 January 2018

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Source code deposit: the new SWH push feature

First version of our software deposit prototype
https://deposit.softwareheritage.org/

Features

- pushing deposits to the Software Heritage archive
  - software source code + metadata
- full transparency of the loading and downloading processes
- download the deposit by cooking the bundle in the vault

SWORD-compliant

- SWORD v2 protocol for single and multipart deposits
- deposit *MUST*, *SHOULD* and *MAY* contain certain metadata attributes

Deposit walkthrough

Prepare source code for deposit

```
$ tar cvf <je-suis-gpl.tgz> </path/to/je-suis-gpl>
```

Deposit walkthrough

Prepare source code for deposit

```
$ tar cvf <je-suis-gpl.tgz> </path/to/je-suis-gpl>
```

Create metadata file with *MUST* metadata

- the url representing the location of the source *MUST* be provided
- the external-identifier *MUST* be provided
- the name/title of the software deposit *MUST* be provided
- the author/s- one or more authors *MUST* be provided

Deposit walkthrough

Prepare source code for deposit

```
$ tar cvf <je-suis-gpl.tgz> </path/to/je-suis-gpl>
```

Create metadata file with *MUST* metadata

- the url representing the location of the source *MUST* be provided
- the external-identifier *MUST* be provided
- the name/title of the software deposit *MUST* be provided
- the author/s- one or more authors *MUST* be provided

```
<?xml version="1.0"?>
<entry xmlns="http://www.w3.org/2005/Atom"
       xmlns:codemeta="https://doi.org/10.5063/SCHEMA/CODEMETA-2.0">
    <title>Je suis GPL</title>
    <external_identifier>ext-id</external_identifier>
    <codemeta:url>forge.softwareheritage.org/source/jesuisgpl/</codemeta:url>
    <codemeta:description>Yes, this is another implementation of
    "Hello, world!" when you run it.</codemeta:description>
    <codemeta:license>
        <codemeta:name>GPL</codemeta:name>
        <codemeta:url>https://www.gnu.org/licenses/gpl.html</codemeta:url>
    </codemeta:license>
    <codemeta:author>
        <codemeta:name> Reuben Thomas and Sami Kerola </codemeta:name>
        <codemeta:jobTitle> Maintainers </codemeta:affiliation>
    </codemeta:author>
</entry>
```

Deposit walkthrough

Pushing a single deposit with metadata

```
$ swh-deposit --username 'name' --password 'pass' --archive je-suis-gpl.tgz
```

Deposit walkthrough

Pushing a single deposit with metadata

```
$ swh-deposit --username 'name' --password 'pass' --archive je-suis-gpl.tgz
```

Response

```
{
  'deposit_id': '11',
  'deposit_status': 'deposited',
  'deposit_date': 'Jan. 30, 2018, 9:37 a.m.'
}
```

Deposit walkthrough

A multistep deposit with partial status can be:

- updated with new content and metadata

Deposit walkthrough

A multistep deposit with partial status can be:

- updated with new content and metadata

Create an incomplete deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --archive foo.tar.gz
```

Deposit walkthrough

A multistep deposit with partial status can be:

- updated with new content and metadata

Create an incomplete deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --archive foo.tar.gz
```

Response

```
{
  'deposit_id': '11',
  'deposit_status': 'partial',
  'deposit_date': 'Jan. 30, 2018, 9:37 a.m.'
}
```

Deposit walkthrough

A multistep deposit with partial status can be:

- updated with new content and metadata

Create an incomplete deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --archive foo.tar.gz
```

Response

```
{
  'deposit_id': '11',
  'deposit_status': 'partial',
  'deposit_date': 'Jan. 30, 2018, 9:37 a.m.'
}
```

Add content or metadata to the deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --deposit-id 11 \
              --archive add-foo.tar.gz
```

Deposit walkthrough

A multistep deposit with partial status can be:

- updated with new content and metadata

Create an incomplete deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --archive foo.tar.gz
```

Response

```
{
  'deposit_id': '11',
  'deposit_status': 'partial',
  'deposit_date': 'Jan. 30, 2018, 9:37 a.m.'
}
```

Add content or metadata to the deposit

```
$ swh-deposit --username 'name' --password 'secret' --partial \
              --deposit-id 11 \
              --archive add-foo.tar.gz
```

Finalize deposit by omitting - -partial flag

Deposit walkthrough

A multistep deposit with partial status can be:

- completely replaced

Update by replacing archive and metadata

```
$ swh-deposit --username 'name' --password 'secret' \
              --deposit-id 11 \
              --archive updated-je-suis-gpl.tar.gz \
              --replace
```

Deposit walkthrough

A multistep deposit with partial status can be:

- completely replaced

Update by replacing archive and metadata

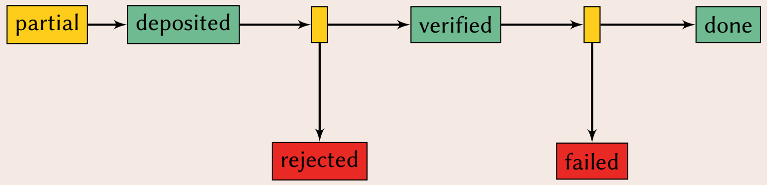
```
$ swh-deposit --username 'name' --password 'secret' \
              --deposit-id 11 \
              --archive updated-je-suis-gpl.tar.gz \
              --replace
```

Response

```
{
  'deposit_id': '11',
  'deposit_status': 'deposited',
  'deposit_date': 'Jan. 30, 2018, 9:37 a.m.'
}
```

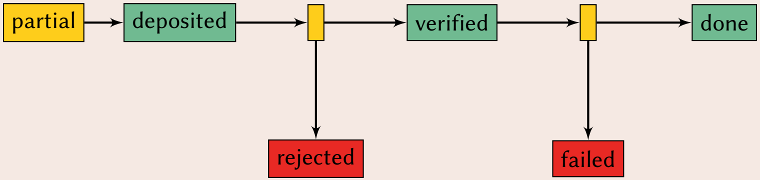
Deposit walkthrough

What's your status?



Deposit walkthrough

What's your status?

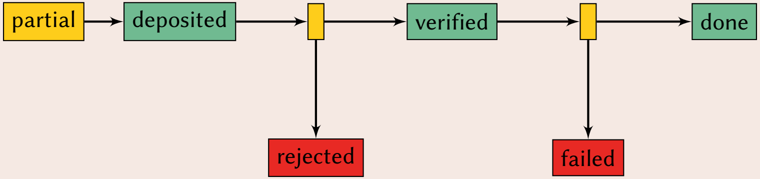


Checking the deposit's state

```
$ swh-deposit --username 'name' --pass 'secret' \
              --deposit-id '11' \
              --status
```

Deposit walkthrough

What's your status?



Checking the deposit's state

```
$ swh-deposit --username 'name' --pass 'secret' \
              --deposit-id '11' \
              --status
```

```
{
  'deposit_id': 11,
  'deposit_status': 'done',
  'deposit_status_detail': The deposit has been successfully loaded
                           into the Software Heritage archive',
  'deposit_swh_id': 'swh:1:rev:608757ea9bd8494d729732cc9a414948c160bd3c'
}
```

Deposit completed and loaded

The deposit was succesfully pushed and integrated into the archive. You can browse it using the deposit's *swh-id*:

- archive.softwareheritage.org/browse/<swh-id>

Deposit completed and loaded

The deposit was succesfully pushed and integrated into the archive. You can browse it using the deposit's *swh-id*:

- archive.softwareheritage.org/browse/<swh-id>

Download

Now we want to download the content with the Vault

Vault walkthrough

Software identifier to request download

The swh-id swh:1:rev:608757ea9bd8494d729732cc9a414948c160bd3c is composed of:

- the object type swh:1:rev
- and the sha1 hash as the object identifier 608757ea9bd8494d729732cc9a414948c160bd3c

We will use the object identifier to create a bundle to download

Vault walkthrough

Software identifier to request download

The swh-id swh:1:rev:608757ea9bd8494d729732cc9a414948c160bd3c is composed of:

- the object type swh:1:rev
- and the sha1 hash as the object identifier 608757ea9bd8494d729732cc9a414948c160bd3c

We will use the object identifier to create a bundle to download

Requesting download with swh-id

```
$ curl -X POST /api/1/vault/revision/608757ea.../gitfast
```

Vault walkthrough

Software identifier to request download

The swh-id swh:1:rev:608757ea9bd8494d729732cc9a414948c160bd3c is composed of:

- the object type swh:1:rev
- and the sha1 hash as the object identifier 608757ea9bd8494d729732cc9a414948c160bd3c

We will use the object identifier to create a bundle to download

Requesting download with swh-id

```
$ curl -X POST /api/1/vault/revision/608757ea.../gitfast
```

Email notification

optionally, an email POST parameter containing an e-mail to notify when the bundle cooking has ended.

Vault walkthrough

Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Vault walkthrough

Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Response

```
{
  'fetch_url': '/api/1/vault/revision/608757ea.../gitfast/raw/',
  'progress_message': None,
  'status': 'pending',
  'id': 4,
  'obj_id': '608757ea9bd8494d729732cc9a414948c160bd3c',
  'obj_type': 'revision_gitfast'
}
```

Vault walkthrough

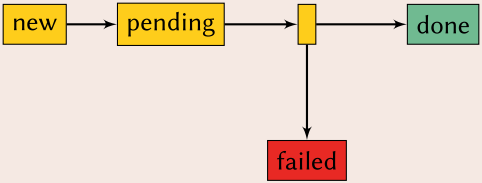
Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Response

```
{
  'fetch_url': '/api/1/vault/revision/608757ea.../gitfast/raw/',
  'progress_message': None,
  'status': 'pending',
  'id': 4,
  'obj_id': '608757ea9bd8494d729732cc9a414948c160bd3c',
  'obj_type': 'revision_gitfast'
}
```

What's your status?



Vault walkthrough

Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Vault walkthrough

Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Response

```
{
  'fetch_url': '/api/1/vault/revision/608757ea.../gitfast/raw/',
  'progress_message': None,
  'status': 'done',
  'id': 4,
  'obj_id': '608757ea9bd8494d729732cc9a414948c160bd3c',
  'obj_type': 'revision_gitfast'
}
```

Vault walkthrough

Checking progress

```
$ curl /api/1/vault/revision/608757ea.../gitfast
```

Response

```
{
  'fetch_url': '/api/1/vault/revision/608757ea.../gitfast/raw/',
  'progress_message': None,
  'status': 'done',
  'id': 4,
  'obj_id': '608757ea9bd8494d729732cc9a414948c160bd3c',
  'obj_type': 'revision_gitfast'
}
```

Download available when status is marked *done*

```
$ curl /api/1/vault/revision/608757ea.../gitfast/raw/ \
  -O path/to/revision.gitfast.gz

$ git init
$ zcat path/to/revision.gitfast.gz | git fast-import
$ git checkout HEAD
```