PROV 2 BigchainDB Documentation

Release 0.4.0

Martin Stoffers

Jun 29, 2017

Contents

CHAPTER 1

PROV 2 BigchainDB

Introduction

This python module provides three different clients to save W3C-PROV documents into a federation of BigchainDB nodes. All clients are implemented with respect to the proposed concepts from the masters thesis Trustworthy Provenance Recording using a blockchain-like database.

See full documentation at: prov2bigchaindb.readthedocs.io

Software Requirements

- Python 3.5 or Python 3.6
- Python development libraries
- GCC and Make
- A local rethinkdb server

Installation

PyPi

Install it by running:

```
pip install prov2bigchaindb
```

You can view prov2bigchaindb on PyPi's package index

Source

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb
```

```
# Setup virtual environment
pyenv env
source env/bin/activate

# Install dependencies and package into virtual enviroment
make setup
```

Usage

DocumentConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
doc_client = clients.DocumentConceptClient(account_id="ID", host="127.0.0.1",␣
↪port=9984)

# Store a document
tx_id = doc_client.save_document(prov_document)

# Retrieve a document
doc = doc_client.get_document(tx_id)
```

GraphConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
graph_client = clients.GraphConceptClient(host="127.0.0.1", port=9984)

# Store a document
tx_ids = graph_client.save_document(prov_document)

# Retrieve a document
doc = graph_client.get_document(tx_ids)
```

RoleConceptClient

```
from prov2bigchaindb.tests.core import setup_test_files
from prov2bigchaindb.core import utils, clients

test_prov_files = setup_test_files()
prov_document = utils.to_prov_document(content=test_prov_files["simple2"])
role_client = clients.RoleConceptClient(host="127.0.0.1", port=9984)

# Store a document
tx_ids = role_client.save_document(prov_document)

# Retrieve a document
doc = role_client.get_document(tx_ids)
```

License

See LICENSE file

CHAPTER 2

Development

Contribute

Please, fork the code on Github and develop your feature in a new branch split from the develop branch. Commit your code to the main project by sending a pull request onto the develop branch

- Issue Tracker: https://github.com/DLR-SC/prov2bigchaindb/issues
- Source Code: https://github.com/DLR-SC/prov2bigchaindb

Setup

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb

# Setup virtual environment
pyvenv env
source env/bin/activate

# Install dependencies
make dev-setup
```

Execute tests

```
make test
```

Coverage report

```
make coverage
```

Compile documentation

```
make docs
```

CHAPTER 3

Changelog

Version 0.4.0 (2017-06-29)

- Updated bigchaindb components to 1.0.0rc1

Version 0.3.1 (2017-04-07)

- Added travis-ci support
- Updated documentation

Version 0.3.0 (2017-04-01)

- Support for the role-based concept

Version 0.2.0 (2017-03-05)

- Support for the graph-based concept
- Added unit tests

Version 0.1.0 (2017-02-21)

- Support for the document-based concept
- Added basic unit tests
- Intergation of Gitlab-CI

CHAPTER 4

Testing Howto

To run the test local follow the next steps

1. Setup your env

```
# Clone project
git clone git@github.com:DLR-SC/prov2bigchaindb.git
cd prov2bigchaindb

# Setup virtual environment
pyvenv env
source env/bin/activate

# Install dependencies
make dev-setup
```

2. Start your BigchainDB test node

The tests require a running BigchainDB and RethinkDB instance.

```
make run
```

3. Setup environment variables

- BDB_HOST: Default: 127.0.0.1
- BDB_PORT: Default: 9984

If you like to connect to BigchainDB node, hosted on a different port and/or machine:

```
BDB_HOST=127.0.0.1
BDB_HOST=9984
```

4. Run your tests

```
# Change to env
source env/bin/activate
#Start tests
make test
```

CHAPTER 5

prov2bigchaindb

prov2bigchaindb package

Subpackages

prov2bigchaindb.core package

Submodules

prov2bigchaindb.core.accounts module

class prov2bigchaindb.core.accounts.**BaseAccount**(*account_id: str*, *store: prov2bigchaindb.core.local_stores.SqliteStore*)

Bases: object

BigchainDB Base Account

get_id() → str
Get Account id

Returns Internal id of Account

Return type str

get_public_key() → str
Get public key

Returns Public key of Account

Return type str

class prov2bigchaindb.core.accounts.**DocumentConceptAccount**(*account_id: str*, *store: prov2bigchaindb.core.local_stores.SqliteStore*)

Bases: *prov2bigchaindb.core.accounts.BaseAccount*

BigchainDB Document Concept Account

save_asset(*asset: dict*, *bdb_connection: bigchaindb_driver.driver.BigchainDB*) → str
Write asset to BigchainDB

Parameters

- **asset** (*dict*) – Dictonary with asset data
- **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transaction Id

Return type str

class prov2bigchaindb.core.accounts.**GraphConceptAccount**(*prov_element: prov.model.ProvElement*, *prov_relations: dict*, *id_mapping: dict*, *namespaces: list*, *store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)

Bases: *prov2bigchaindb.core.accounts.BaseAccount*

BigchainDB Graph Concept Account

get_tx_id() → str
Get the tx_id that describes the account in BigchainDB

Returns Transaction id of account

Return type str

has_relations_with_id() → bool
Indicates whether an account has relation with ids or not :return: True if one or more relation does have ids :rtype: bool

has_relations_without_id() → bool
Indicates whether an account has relation without ids or not :return: True if one or more relation does have ids :rtype: bool

save_relations_with_ids(*bdb_connection: bigchaindb_driver.driver.BigchainDB*) → list
Writes all assets with relations (having ids) to BigchainDB

Parameters **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transactions ids of all relations

Return type list

save_relations_without_ids(*bdb_connection: bigchaindb_driver.driver.BigchainDB*) → list
Write all assets with relations to BigchainDB

Parameters **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transactions ids of all relations

Return type list

save_instance_asset(*bdb_connection: bigchaindb_driver.driver.BigchainDB*) → str
Write provenance describing the account to BigchainDB

Parameters **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transactions id of instance

Return type str

class prov2bigchaindb.core.accounts.**RoleConceptAccount**(*agent: prov.model.ProvAgent, relations: list, elements: dict, id_mapping: dict, namespaces: list, store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)

Bases: *prov2bigchaindb.core.accounts.BaseAccount*

BigchainDB Graph Concept Account

get_tx_id() → str
Get the tx_id that describes the account in BigchainDB

Returns Transaction id of account

Return type str

save_elements(*bdb_connection: bigchaindb_driver.driver.BigchainDB*) → list
Writes all elements with assets to BigchainDB

Parameters **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transactions ids of all relations

Return type list

save_instance_asset(*bdb_connection: bigchaindb_driver.driver.BigchainDB*) → str
Write provenance describing the account to BigchainDB

Parameters **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns Transactions id of instance

Return type str

prov2bigchaindb.core.clients module

class prov2bigchaindb.core.clients.**BaseClient**(*host: str = '0.0.0.0', port: int = 9984, num_connections: int = 5, local_store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)

Bases: object

BigchainDB Base Client

test_transaction(*tx: dict*) → bool
Validate a transaction against BigchainDB

Parameters **tx** (*dict*) – Transaction to test

Returns True or Exception

Return type bool

save_document(*document: object*) → object
Abstract method to store a document

Parameters **document** (*object*) – Document to save

Returns id

Return type object

get_document (*document_id: object*) → prov.model.ProvDocument
Abstract method to retrieve a document

Parameters **document_id** (*object*) – Document to save

Return type ProvDocument

class prov2bigchaindb.core.clients.**DocumentConceptClient** (*account_id: str = None, host: str = '0.0.0.0', port: int = 9984, num_connections: int = 1, local_store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)
Bases: *prov2bigchaindb.core.clients.BaseClient*

save_document (*document: str*) → str
Write a document into BigchainDB

Parameters **document** (*str or bytes or ProvDocument*) – Document as JSON/XML/PROVN

Returns Transaction id of document

Return type str

get_document (*tx_id: str*) → prov.model.ProvDocument
Retrieve a document by transaction id from BigchainDB

Parameters **tx_id** (*str*) – Transaction Id of Document

Returns Document as ProvDocument object

Return type ProvDocument

class prov2bigchaindb.core.clients.**GraphConceptClient** (*host: str = '0.0.0.0', port: int = 9984, num_connections: int = 5, local_store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)
Bases: *prov2bigchaindb.core.clients.BaseClient*

static calculate_account_data (*prov_document: prov.model.ProvDocument*) → list
Transforms a ProvDocument into a tuple with ProvElement, list of ProvRelation and list of Namespaces

Parameters **prov_document** – Document to transform

Returns List of tuples(element, relations, namespace)

Return type list

save_document (*document: str*) → list
Write a document into BigchainDB

Parameters **document** (*str or BufferedReader or ProvDocument*) – Document as JSON/XML/PROVN

Returns List of transaction ids

Return type list

get_document (*document_tx_ids: list*) → prov.model.ProvDocument
Retrieve a document by a list transaction ids from BigchainDB

Parameters **document_tx_ids** (*list*) – Transaction Ids of Document

Returns Document as ProvDocument object

Return type ProvDocument

class prov2bigchaindb.core.clients.**RoleConceptClient**(*host: str = '0.0.0.0'*, *port: int = 9984*, *num_connections: int = 5*, *local_store: prov2bigchaindb.core.local_stores.SqliteStore = <prov2bigchaindb.core.local_stores.SqliteStore object>*)

Bases: *prov2bigchaindb.core.clients.BaseClient*

static calculate_account_data(*prov_document: prov.model.ProvDocument*) → list
Transforms a ProvDocument into a list of tuples including: ProvAgent, list of ProvRelations from agent, list of ProvElements associated to ProvAgent, list of Namespaces

Parameters **prov_document** – Document to transform

Returns List of tuples(ProvAgent, list(), list(), list())

Return type list

save_document(*document: str*) → list
Write a document into BigchainDB

Parameters **document** (*str or BufferedReader or ProvDocument*) – Document as JSON/XML/PROVN

Returns List of transaction ids

Return type list

get_document(*document_tx_ids: list*) → prov.model.ProvDocument
Returns a document by a list transaction ids from BigchainDB

Parameters **document_tx_ids** (*list*) – Transaction Ids of Document

Returns Document as ProvDocument object

Return type ProvDocument

prov2bigchaindb.core.exceptions module

exception prov2bigchaindb.core.exceptions.**Prov2BigchainDBException**
Bases: Exception

exception prov2bigchaindb.core.exceptions.**CreateRecordException**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**ParseException**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**NoAccountFoundException**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**AccountNotCreatedException**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**NoRelationFoundException**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**TransactionIdNotFound**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

exception prov2bigchaindb.core.exceptions.**BlockIdNotFound**
Bases: *prov2bigchaindb.core.exceptions.Prov2BigchainDBException*

prov2bigchaindb.core.local_stores module

class prov2bigchaindb.core.local_stores.**BaseStore**(*db_name: str = None*)
Bases: object

clean_tables()
Delete all entries from all tables (Used for unit tests)

write_account(*account_id: str*, *public_key: str*, *private_key: str*, *tx_id: str = None*)
Writes a new account entry in to the table accounts

Parameters

- **tx_id** (*str*) – Transactions id
- **account_id** (*str*) – Id of account
- **public_key** (*str*) – Public key of account
- **private_key** (*str*) – Private key of account

get_account(*account_id: str*) → tuple
Returns tuple of account from data by account_id

Parameters **account_id** (*str*) – Id of account

Returns Tuple with account_id, public_key, private_key and tx_id

Return type tuple

write_tx_id(*account_id: str*, *tx_id: str*)
Writes tx_id for given account_id

Parameters

- **account_id** (*str*) – Id of account
- **tx_id** (*str*) – Transaction id, which represents the account in BigchainDB

class prov2bigchaindb.core.local_stores.**SqliteStore**(*db_name: str = ':memory:'*)
Bases: *prov2bigchaindb.core.local_stores.BaseStore*

clean_tables()
Delete all entries from all tables (Used for unit tests)

write_account(*account_id: str*, *public_key: str*, *private_key: str*, *tx_id: str = None*)
Writes a new account entry in to the table accounts

Parameters

- **tx_id** (*str*) – Transactions id
- **account_id** (*str*) – Id of account
- **public_key** (*str*) – Public key of account
- **private_key** (*str*) – Private key of account

get_account(*account_id: str*) → tuple
Returns tuple of account from data by account_id

Parameters **account_id** (*str*) – Id of account

Returns Tuple with account_id, public_key, private_key and tx_id

Return type tuple

write_tx_id(*account_id: str*, *tx_id: str*)
Writes tx_id for given account_id

Parameters

- **account_id** (*str*) – Id of account
- **tx_id** (*str*) – Transaction id, which represents the account in BigchainDB

prov2bigchaindb.core.utils module

prov2bigchaindb.core.utils.**to_prov_document**(*content: str*) → prov.model.ProvDocument
Takes a string, bytes or ProvDocument as argument and return a ProvDocument The strings or bytes can contain JSON or XML representations of PROV

Parameters **content** – String or BufferedReader or ProvDocument

Returns ProvDocument

Return type ProvDocument

prov2bigchaindb.core.utils.**wait_until_valid**(*tx_id: str*, *bdb_connection: bigchaindb_driver.driver.BigchainDB*)
Waits until a transaction is valid in BigchainDB

Parameters

- **tx_id** (*str*) – Id of transaction to wait on
- **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

prov2bigchaindb.core.utils.**is_valid_tx**(*tx_id: str*, *bdb_connection: bigchaindb_driver.driver.BigchainDB*) → bool
Checks once if a transaction is valid

Parameters

- **tx_id** (*str*) – Id of transaction to check
- **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns True if valid

Return type bool

prov2bigchaindb.core.utils.**is_block_to_tx_valid**(*tx_id: str*, *bdb_connection: bigchaindb_driver.driver.BigchainDB*) → bool
Checks if block with transaction is valid

Parameters

- **tx_id** (*str*) – Id of transaction which should be included in the
- **bdb_connection** (*BigchainDB*) – Connection object for BigchainDB

Returns True if transactions is in block and block is valid

Return type bool

Module contents

Submodules

prov2bigchaindb.version module

Module contents

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index

R

S

T

W