OpenBudgets.eu: Fighting Corruption with Fiscal Transparency

Project Number: 645833 Start Date of Project: 01.05.2015 Duration: 30 months

Deliverable D3.4

Packaging and signing specifications

Dissemination Level	Public
Due Date of Deliverable	Month 18, 31.10.2016
Actual Submission Date	31.10.2016
Work Package	WP3, Budget and Spending Data Visualisation and Exploration
Task	T 3.4
Type	Demonstration
Approval Status	Draft
Version	0.1
Number of Pages	2
Filename	Deliverable 3.4 - Packaging and signing specifications

Abstract: This document documents packaging and signing features for visualisations in the OpenBudgets.eu platform.

Project funded by the European Union's Horizon 2020 Research and Innovation Programme (2014 – 2020)

History

Version	Date	Reason	Revised by
0.1	01.01.2017	First contribution	Paul Walsh
		Review	

Author List

Organisation	Name	Contact Information
OKFGR	Lazaros Ioannidis	larjohn@okfn.gr
OKI	Paul Walsh	paul.walsh@okfn.org

Table of Contents

1.Introduction

Public fiscal datasets accessed via open data platforms such as OpenBudgets.eu enable new user interactions and use cases for the data that were not previously possible. This includes the downloading of that data for offline use.

However, reuse of the data, and presentation of it in new contexts, does not provide any guarantees as to the authenticity of the new presentation of the data.

There is little to stop a reuser from manipulating a data source, by taking it out of the original context (the platform), changing it, and presenting it elsewhere as authoritative.

In this deliverable for packaging and signing specifications, a specification for packaging and signing data for offline (off platform) use is presented, one where a user can easily and simply get a single package downloaded to her computer that represents the state of a particular visualisation, and providing basic data provenance guarantees from the platform, via the signing of the data with encryption keys, to provide a mechanism of asserting that a particular dataset has not been tampered with.

2.Specification

There are two related aspects to creating signed, offline visualisations of fiscal data from the OpenBudgets.eu platform:

- Exporting a particular visualisation state for offline usage
- Signing the exported state to provide a mechanism of verifying that it is what it says

The most relevant component to implement such functionality for users is in the OS Viewer app. This app allows users to interactively build a set of visualisations from a dataset, and is a core interface for user interaction on the OpenBudgets.eu platform.

OS Viewer is an ideal target for the following reasons:

- The user can create a static output from anything she can do in the "workspace" of the viewer, which means one or many visualisations
- While mostly a client side app, it has a lightweight node.js server that can be used for preprocessing and creating the export if needed

Additionally, a supporting service for providing the signing functionality and the production of PNG and PDF versions of the interface is required, and the implementation of this is called Sealer. This service is not user-facing. Rather, it is designed for programmatic use, based on actions made by the user in the OS Viewer user interface.

User workflow

The following user workflow should be implemented to enable packaging and signing of data, and final download of the signed package to the user's computer.

- User interacts with OS Viewer to create a set of visualisations, after having already selected a dataset to interact with
- When user has a visualisation state that is useful, she click a "export/download a verified copy of this visualisation" button
- This starts a process that creates the required outputs, zips them, and prompts for download to the user's download folder

Proposed Implementation

A narrative description of the implementation is as follows:

OS Viewer app

- The visualisation state to be exported is generated from the OS Viewer app
- The button to download is added to the Viewer
- Some way to get the state's data and expose as JSON

Sealer app

- Generates PNG of OS Viewer state
- Gets the data as JSON
- Gets the PNG
- Holds private key
- Creates signature
- Creates zip
- Exposes zip for download

Expected Outputs

The implementation should result in the following outputs:

- A single PNG file that reproduces the state of the OS Viewer "workspace"
- A JSON file that represents the data used to create the Viewer "workspace"
- A key pair to sign the data (public key distributed with the PNG and JSON)
- A zip file to contain all of the above. This is the file downloaded by the user

3.Implementation

Sealer was implemented in Javascript, utilizing the Express.js framework. There are three endpoints in the current version:

1. GET /sign?uri=<an OpenSpending embed page URI> returns a zip file which contains a signature and an additional zip file containing the rendered visualizations and the data used
2. GET /store/sign?uri=<an OpenSpending embed page URI> only returns a zip file containing the data and the renderings, without any signature
3. POST verify, with a signed package file uploaded, verifies the file against the signature

To be able to render the visualizations on time and capture them, an additional provision has been implemented in OpenSpending - global JavaScript variables signify the readiness of the web page.

In the default configuration, the application uses SHA256 for hashing and 2048-bit RSA for signing.