Data retention of data managed by data engineering

This document describes the retention policy and nature of data that is collected from institutes and organizations as of September 8 2023 under the most recent releases[1] at that date. It's a description of the business rules that have been set in place at the request of SURF. The document does not go into detail about agreements and contracts that have been made between SURF and data processing services like Amazon Web Services (AWS), because these details are not known by the programmers, but it will give a very short summary of what is clear from the Terraform code.

Harvest, proxy and staff data

There are three types of data that we process. We give a short description of the data for each type to give a better picture of what is being processed.

The harvest data is all data that comes from institutes and organizations and which gets stored in various places on AWS infrastructure. Currently only research products and learning materials get stored, the data engineering team (DET) uses the term "product" to describe both use cases. This data is known as "publications" under SURF Sharekit and it contains different metadata about some digital object related to higher education. Part of this data is the author which may, apart from name information, specify one or more personal identifiers like: ISNI, DAI, ORCID or email addresses. The purpose for this data is to identify authors across organizations as well as for endusers to get into contact with authors. The access policy for the metadata is OpenAccess. So it's assumed that the given information is released to the public and it's possible to search through this data, but making a complete copy of the data should currently only be possible by authorized users. Radically Open Security (ROS) has been given the assignment to do penetration tests in this regard, but the programmers haven't seen any results related to this.

One important role the harvester plays is to provide a data standard, where no data standard exists. Due to for instance innovative use cases that standards haven't embrassed yet or the refusal by software manifacturers to comply with the standard (Elsevier's Pure for example). This pseudo standard is important to make it easier to use the data in various context like search portals or machine learning systems. The alternative would be that each consumer of data would write their own ETL pipeline. However this role of enforcing a pseudo standard and the role of storing the data are independant from each other. There are currently two entities "persons" and "projects" where DET provides an interface for other services to get data in a consistent format from many different sources, but the storage of this data is handled by the consumers. This is what we refer to as proxy data and it's handled by the sources-middleware repository instead of the harvester. Please note that previous managers have expressed the desire that DET would also handle the storage for "persons" and "projects" as well as other entities like "organizations" and "collections". Over the summer of 2023 a big harvester refactor of about 12 working days took place to start to facilitate this feature.

The third data type is IP logging, (system) user profiles and authentication tokens. This data only gets processed for SURF employers or freelancers. You at least need a SURF email address to be able to access most of the systems, because EduVPN using the "all traffic" mode is required to pass the firewalls. Theoretically an enduser could use a search API endpoint directly as well as health check endpoints, but the system is designed to be used by frontends like Edusources and Publinova who are expected to do the actual handling of user interactions and the data that generates.

1 Harvester version 1.38.56 and Sources version 1.2.5

Storage of data

The DET team means to store only what we call “harvest” data as described above. The contracts with clients should reflect that this is OpenAccess data. Other data like IP addresses are of course being processed, but it’s opaque for the team which agreements have been made with AWS regarding the processing (and potentially storage) of this data.

There are three places where DET stores data. We have a Postgres database that uses transactions to manage async write operations. This Postgres database is setup using the Relational Database Service (RDS) and has a retention period of 30 days. Within the database the data gets stored under a storage model named DatasetVersion. Every night a new DatasetVersion gets created which is partially based on the DatasetVersion of the day before. Some sources can only provide the data that has changed the previous day and in these cases a lot of data gets copied between DatasetVersions, while other sources require the download of all data every night and currently no data gets copied when that is the case. However during the major refactor of the summer of 2023 the harvester will detect which products disappear as compared to yesterday to be able to pass on deletes as well as upserts to consuming services even though the source of the data doesn’t work with these concepts. These DatasetVersions have a retention period that can be configured using environment variables. The defaults for production are specified in a YAML configuration file and can be easily modified, but I’ll assume these defaults to describe the current business rules. DatasetVersions are kept for the last 14 days except those versions that were made with older code versions. The DatassetVersion stores which code version was used to create the data. We will always keep at least one DatasetVersion for the last three code versions. So if our code versions are 0.0.1, 0.0.2 and 0.0.3 we at most keep the 14 DatasetVersions that were made with 0.0.3 in the last fourteen days as well as one version for 0.0.1 and 0.0.2. These last version will be stored indefinitely until code versions 0.0.4 and 0.0.5 get released and start to harvest. In practice we release so often that we have 14 DatasetVersions in retention at any given time.

Apart from writing the data from sources to our Postgres database we also make the data available for search/reading through the OpenSearch service of AWS. This data is schemaless, but to provide a consistent data format to consuming services, as well as a consistent search experience, there are some self emposed restrictions/formats. For every DatasetVersion we create three indices. One for NL, one for EN and one for other languages. This also happens at night. Apart from that there are background tasks that fire every 30 seconds to keep the Postgres and OpenSearch in sync when new data comes in through webhooks. In practice this means the system ocassionally writes one Postgres row to OpenSearch. The DET hasn’t seen more writes to OpenSearch than two rows every 30 seconds since the start of 2023 and this is very rare. When old dataset versions get deleted the corresponding indices also get deleted. Since it is very cheap to build these indices from scratch based on any DatasetVersion and since OpenSearch is very reliable SURF hasn’t seen reason to create OpenSearch snapshots. This would however be a possibility if they change their mind.

Last but not least we store data dumps and data fixtures in Simple Storage Service (S3). These dumps get created through commands that DET members can execute at will. Previous dumps usually get overridden except when large schema changes take place, then we keep versions for both schema’s and the old version will need to be removed manually through the AWS Web Console. To run automated tests we also store fake responses of API calls to the source systems maintained by our clients. We change the names of test data that we use in the tests, but very old tests that haven’t changed in a long time may still mention authors from before this was a practice. Any data nog mentioned in test assertions will be stored as-is. The dump and test data is only available to programmers, Github (which executes the tests) and of course AWS.