ORCID and repository integrations

Gabriela Mejias

LIBSENSE Community Call

25 May 2021

ORCID Connecting Research and Researchers

@gabioshka
@ORCID_Org

ORCID APIs I

All of the ORCID APIs are based on the same set of technologies:

- **REST**: ORCID APIs are "RESTful"; they follow the constraints of the REST architectural style
- **OAuth**: ORCID APIs use the OAuth 2.0 authentication protocol to grant client applications access to users' ORCID records
- **XML/JSON**: ORCID APIs support data exchange in XML and JSON, as well as RDF, GZIP and Turtle

https://info.orcid.org/documentation/api-tutorials/
https://github.com/ORCID/ORCID-Source/tree/master/orcid-api-web

ORCID APIs II

➔ Sandbox (Testing)
https://orcid.org/content/register-client-application-sandbox

➔ **Production (Live Registry)**
https://orcid.org/content/register-client-application-production-trusted-party

Public API Freely available	•Authenticate: Get a user's authenticated ORCID iD •Read (Public): Search/retrieve public data •Create: Facilitate creation of new ORCID records (via on-demand process)
Member API - Basic Basic membership	•Read (Limited): Search/retrieve limited-access data •Add: Post new items to a record •Update: Edit or delete items you previously added
Member API - Premium Premium & Consortium membership	•Webhooks: Receive notifications of updates •Monthly reports (including email stats) •Access to monthly public data file

https://members.orcid.org/api
https://orcid.org/about/membership/comparison

Best practices

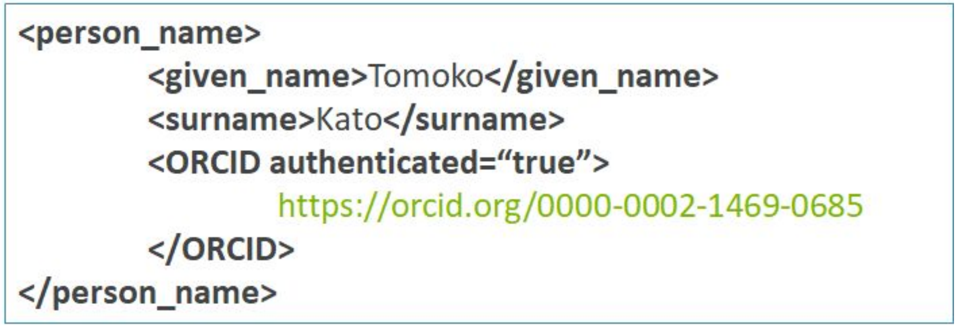
- Support **collecting authenticated ORCID iDs**
  - Support other ways of obtaining ORCID iDs, including in mediated deposits and bulk uploads by repository managers, as well as automated deposits from other systems
- Allow administrators to request authenticated ORCID iDs and ORCID record update permission from authors and co-authors, in cases where iDs are missing or have not been authenticated
- **Support displaying ORCID iDs** wherever user/contributor information is displayed
- **Support pulling and pushing information to and from ORCID**
- Provide testing, logging and reporting features to help administrators troubleshoot issues and manage ORCID-related data in their repository
- **Support exposing ORCID iDs in metadata outputs**, such as OAI-PMH XML, wherever possible
- **Provide documentation about ORCID features, for both administrators and end users**

https://doi.org/10.23640/07243.7777274.v3

Collecting iDs

- **Via direct interaction with a user:** An authenticated iD (and, optionally, permission to update the user's ORCID record) is obtained from a user through an OAuth interaction, facilitated by the repository platform
- **Via mediated deposit:** An unauthenticated iD is associated with an item or user by an individual other than the iD owner. Since an OAuth interaction did not take place, the repository platform instance does not have permission to update the user's ORCID record
- **Via external 3rd party import/deposit:** An iD is extracted from metadata obtained via import or deposit from an external source. This iD may or may not have been authenticated by the metadata source. Since an OAuth interaction did not take place, the repository platform instance does not have permission to update the user's ORCID record.

Displaying iDs



```
<person_name>
        <given_name>Tomoko</given_name>
        <surname>Kato</surname>
        <ORCID authenticated=“true”>
                https://orcid.org/0000-0002-1469-0685
        </ORCID>
</person_name>
```

- ORCID iDs should be displayed where user/contributor information is displayed
- Authenticated iDs should be displayed according to the our guidelines
- Unauthenticated iDs should be displayed with an indication such as “(unconfirmed)”, “(unverified)” or “(unauthenticated)”, or the equivalent in the local language
- Include ORCID iDs in machine-readable metadata

Pulling/Pushing data

PULL

- Request & use permission to read information
    - Public API: read-public
    - Member API: read-limited
    - Auto-populate your systems & forms to save users' time

PUSH

- Request permission to write data
    - Member API: /activities/update and /person/update
    - Store put codes
    - Include a work identifier

Integrations out of the box

- Brocade
- DSpace via KAUST solution
  - Part 1 Part 2
- Eprints
- Figshare via DataCite auto update (user enabled)
- Haplo
- Islandora
- Zenodo via DataCite auto update (user enabled)
- Via BASE (user enabled)

Custom integrations

- Get access to the ORCID Sandbox API

  https://info.orcid.org/register-a-client-application-sandbox-member-api/
- Plan/ test our recommended workflow

  https://info.orcid.org/documentation/workflows/repository-systems/
- Launch and communicate

And don’t forget to let us know about your use case!👇🏾
https://spaces.wacren.net/display/LIBSENSE/PID+Implementation+use+cases

Thanks! Questions?

ℹ️ https://support.orcid.org/hc/en-us

📩 g.mejias(at)orcid.org

🐦 @gabioshka