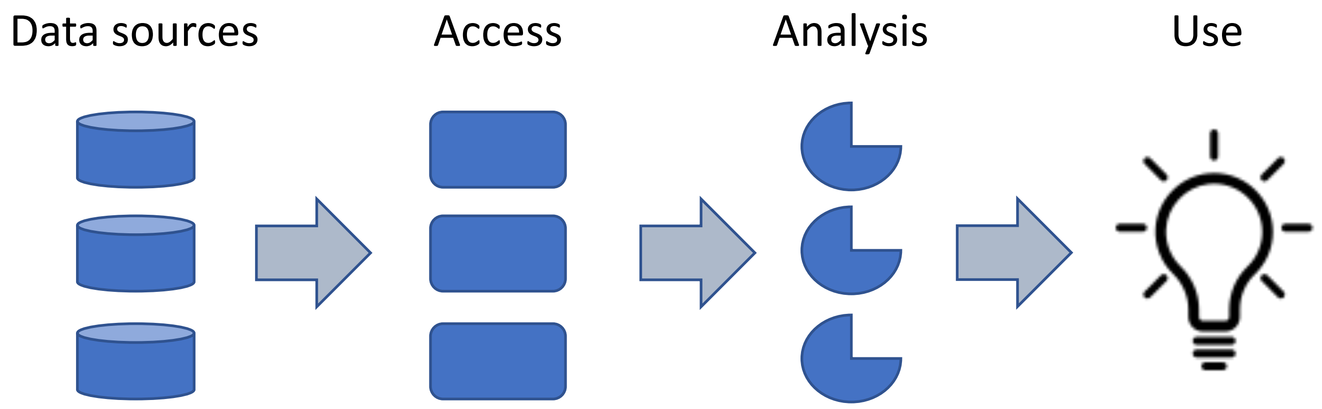
Session 1: Tools, data, methods

Chair: Jari Arkko

Presentations

- Datatracker interface (Sparks)
- BigBang (Benthall)
- SODESTREAM (McQuistin)
- IETF website analytics (Wood)



Relevant papers:

- Using Complex Systems Analysis to Identify Organizational Interventions (Sebastian Benthall)
- The ietfdata Library (Stephen McQuistin, Colin Perkins)
- The RFC Prolog Database (Marc Petit-Huguenin)
- Observations about IETF process measurements (Jari Arkko)
- And this, though not a paper: https://www.ietf.org/policies/web-analytics/ (IETF)

Accessing Datatracker Data

Robert Sparks

IAB AID Workshop

Session 1

What’s available?

- Files (drafts, RFCs, agendas, minutes, photos)
  - Available over HTTPS and through rsync
- Metadata about People, Documents, Groups, Meetings, and more
  - Stored in SQL, structured as Django Models
- Archives of mailing lists
  - Managed by the mailarchive Django project rather than the datatracker

Details at https://notes.ietf.org/iab-aid-data-resources

Two ways to get to the datatracker data

- Set up a local development environment
  - Using docker (note that the first build will take 40-ish minutes)
  - Developer database dump refreshed daily
  - Django shell allows construction of arbitrary querysets
  - Instructions at https://notes.ietf.org/iab-aid-data-resources
- Use the v1 API
  - Built on Tastiepie
  - Can just be browsed (xml or json output)
  - Best accessed with curl and jq
  - Not as capable as the development environment
  - Some ordering care needed when retrieving a large number of records

Preview of what you can do

```
In [3]: Document.objects.filter(documentauthor__person__name= "Robert
Sparks").count()

Out[3]: 258


In [4]: Counter(Document.objects.filter(documentauthor__person__name=
"Robert Sparks").values_list('type_id',flat=True))

Out[4]: Counter({'draft': 67, 'review': 191})
```

Preview of what you can do

```
% curl "https://datatracker.ietf.org/api/v1/doc/document/?states__slug=lc&format=json" \
    | jq "[.objects[] | .name]"
[
  "draft-eastlake-rfc6931bis-xmlsec-uris",
  "draft-ietf-i2nsf-capability-data-model",
  "draft-ietf-i2nsf-nsf-monitoring-data-model",
  "draft-ietf-httpbis-priority",
  "draft-ietf-acme-dtnnodeid",
  "draft-ietf-httpbis-http2bis",
  "draft-ietf-lamps-samples"
]
```

What’s in there?

- A lot – the datatracker is a large application, with complex data relationships
  - Over 100,000 documents
  - Nearly 20,000 people
  - Over 1000 meetings (including interims)
- Minimal PII for People
  - Names and email addresses
  - Sometimes Affiliation and Country
  - No addresses (though those can sometimes be mined from drafts)
  - No other explicitly captured demographics
- See https://notes.ietf.org/iab-aid-datatracker-database-overview

What’s in there: History

- Most metadata is saved when a Document, Group, or Person object is modified
- Reasonably complete for recent (10 to 15 years) history
- Poor for older history – data is often incomplete, and is occasionally completely wrong
- The models were designed for tracking the current state of work. Mining the history records can be complicated.
  - It is very hard, for example, to determine when someone stopped being a chair of a given working group.

Backup slides

What’s in there: Code

- Don’t ignore the codebase
- Many utilities exist to make data mining easier
  - Finding the current (or final) IESG ballot state for a document
  - Extracting authors from text Internet-Drafts
  - Finding the chain of all documents that ultimately normatively depend on a given document through dependencies on other documents.

Getting started

- Explore the development environment and the v1 API
- Ask questions:
  - I’m available at email:rjsparks@nostrum.com and on the IETF Slack
  - Consider subscribing to tools-discuss@ietf.org

The ietfdata Library

Stephen McQuistin
Colin Perkins

IAB Workshop on Analyzing IETF Data (AID)
November 29th 2021

This work is funded by the UK Engineering and Physical Sciences Research Council, under grants EP/S033564/1 and EP/S036075/1.

IETF Data

Datatracker

RFC Index

Mail Archives

IETF Data

Datatracker

REST API

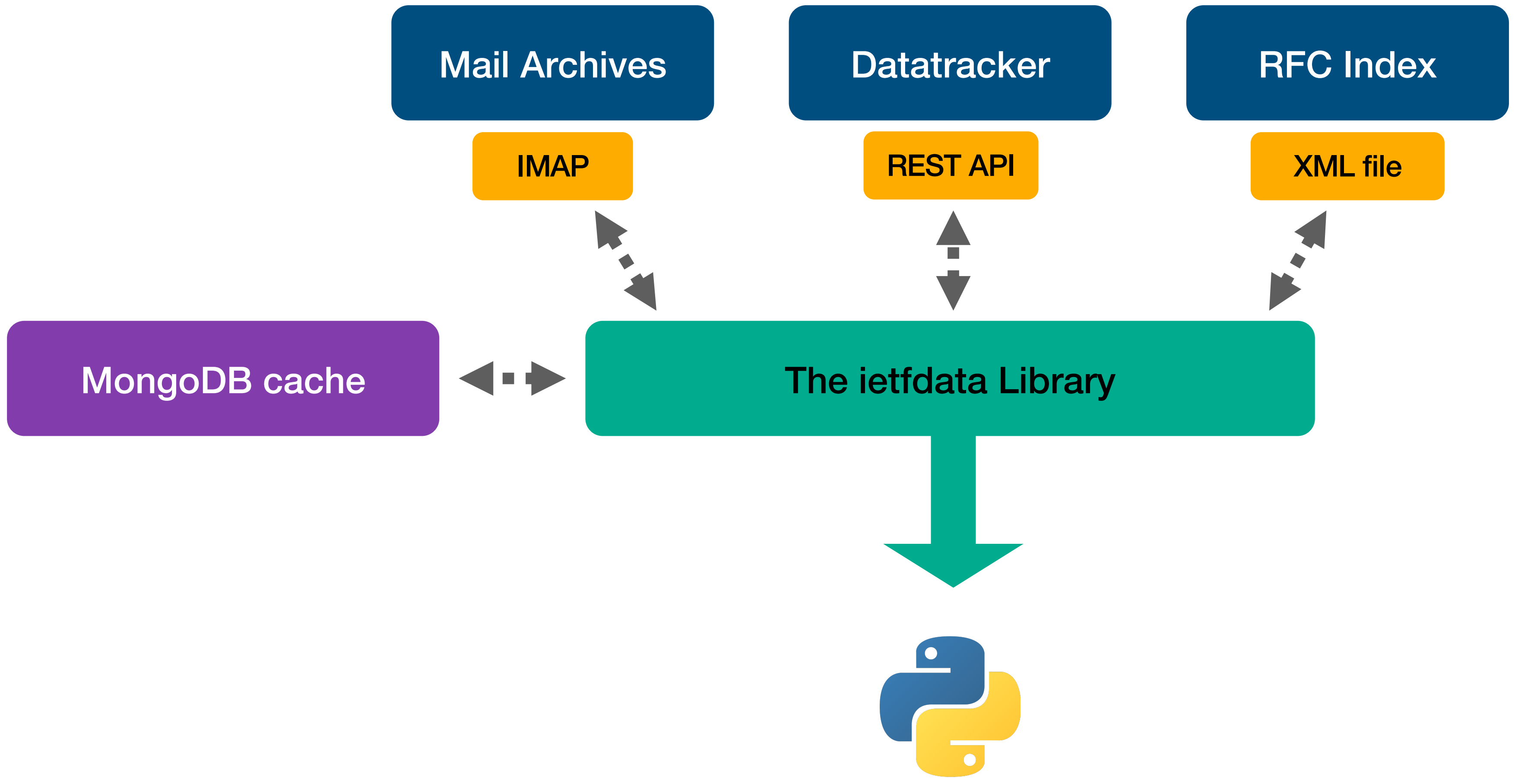
RFC Index

XML file

Mail Archives

IMAP

The ietfdata Library



What data is available?

- Author list
- Stream
- IETF working group and area information, if appropriate
- Status (at publication, and current)
- Updates/obsoletes relationships between RFCs

RFC Index

Mail Archives

Datatracker

What data is available?

- IETF mailing lists, and mirrors, from around 1995
- Messages grouped by mailing list
- Library provides a thread abstraction

RFC Index

Mail Archives

Datatracker

What data is available?

- Documents
  I-Ds, agendas, bluesheets, charters, minutes, recordings, …
- Groups
  Events, milestones, roles (chairs and ADs), URLs …
- Intellectual property disclosures
- Mailing list subscriptions
- Meetings
  Registrations, schedule, session details, …
- People
  Names, e-mail addresses, biographies, …
- Reviews
  Requests, reviews, assignments, review teams/directorates, …

RFC Index

Mail Archives

Datatracker

Example: Meeting registrations

```
from ietfdata.datatracker import *

dt = DataTracker()

p = dt.person_from_email("sm@smcquistin.uk")
print("Name: {}".format(p.name))

for reg in dt.meeting_registrations(person=p):
    meeting = dt.meeting(reg.meeting)
    if dt.meeting_type(meeting.type) == dt.meeting_type_from_slug("ietf"):
        print(F"Registered for IETF {meeting.number} in {meeting.city}")
        print(F"  Name: {reg.first_name} {reg.last_name}")
        print(F"  Affiliation: {reg.affiliation}")
        print(F"  Email: {reg.email}")
```

Example: Meeting registrations

```
Name: Stephen McQuistin
Registered for IETF 94 in Yokohama
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 96 in Berlin
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 101 in London
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 103 in Bangkok
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: stephen.mcquistin@glasgow.ac.uk
Registered for IETF 105 in Montreal
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
...
```

Example: Meeting registrations

```
Name: Stephen McQuistin
Registered for IETF 94 in Yokohama
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 96 in Berlin
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 101 in London
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
Registered for IETF 103 in Bangkok
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: stephen.mcquistin@glasgow.ac.uk
Registered for IETF 105 in Montreal
  Name: Stephen McQuistin
  Affiliation: University of Glasgow
  Email: sm@smcquistin.uk
...
```

Finds registrations by person, even if a different e-mail address was used

Summary

- The ietfdata library provides a Python API for accessing email archives, the Datatracker, and the RFC Index
- Support for caching, to improve performance and reduce load on the IETF's infrastructure
- Available via PyPI, with code and examples on GitHub

Installation via PyPI:

```
pip install ietfdata
```

Code and examples:
https://github.com/glasgow-ipl/ietfdata

Examples

More at github.com/glasgow-ipl/ietfdata/tree/master/examples

Example: Bluesheets

```
from ietfdata.datatracker import *

dt = DataTracker()

bluesheets = dt.document_type_from_slug("bluesheets")
quic = dt.group_from_acronym("quic")

for doc in dt.documents(doctype = bluesheets, group = quic):
    print(doc.title)
    print(doc.url())
    print("")
```

Example: Organisational chart

```python
from ietfdata.datatracker import *

dt = DataTracker()

def print_group(group : Group, level : int):
    for i in range(0, level):
        print("  ", end="")
    print(group.name)
    for g in dt.groups(parent = group, state = dt.group_state_from_slug("active")):
        print_group(g, level + 1)

print_group(dt.group_from_acronym("ietf"), 0)
print_group(dt.group_from_acronym("irtf"), 0)
```

Example: Group roles

```
from ietfdata.datatracker import *

dt = DataTracker()

def group_roles(group: Group):
    print(F"Group: {group.name}")
    for gr in dt.group_roles(group = group):
        e  = dt.email(gr.email)
        p  = dt.person(gr.person)
        rn = dt.role_name(gr.name)
        print(F"  {rn.name}: {p.name} <{e.address}>")
    print("")


for g in [dt.group_from_acronym("ietf"),
          dt.group_from_acronym("irtf"),
          dt.group_from_acronym("iesg"),
          dt.group_from_acronym("irsg"),
          dt.group_from_acronym("quic")]:
    group_roles(g)
```