IETF 123 Hackathon
RPP (RESTful Provisioning Protocol)

Sebastian Jung (DENIC)
Pawel Kowalik (DENIC)
Christian Simmen (DENIC)
Tobias Theel (DENIC)

Stéphane Bortzmeyer (Afnic)
Q Misell (MPI-INF)
Ruth Trevor-Allen (Nominet)
Maarten Wullink (SIDN)

Goals

RPP

- Alternative to EPP, for domain names creation/management

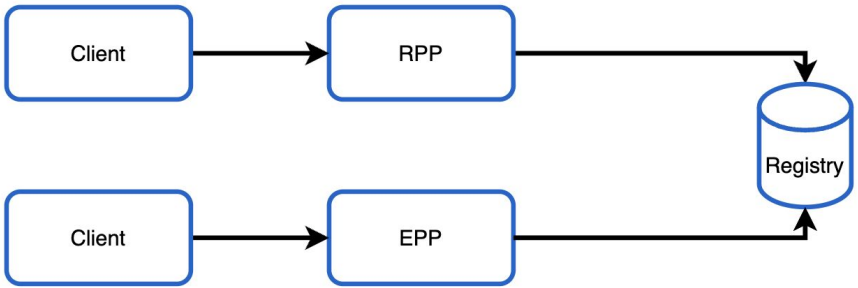
Prototyping

- Experimental implementations of RPP (draft-wullink-rpp-core)
- Experiments with alternative representation of DNS data draft-simmen-rpp-dns-data

Discussion:

- Data modelling for provisioning objects
- Refinement of RPP drafts:
  - draft-ietf-rpp-requirements-01
  - draft-kowalik-rpp-architecture-02
  - draft-wullink-rpp-core-01
  - draft-simmen-rpp-dns-data-00

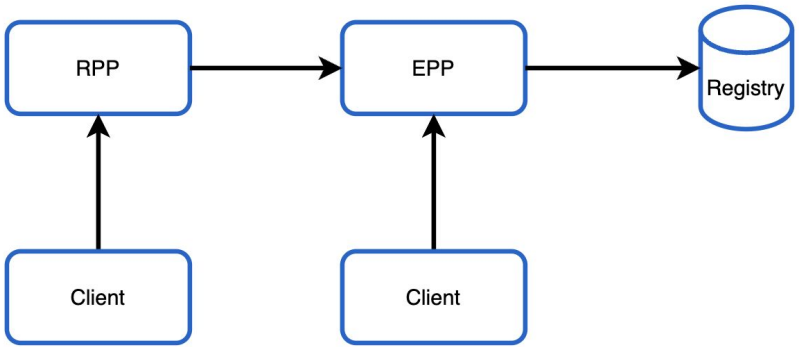
Deployment - Parallel / Native



Working implementations:

- DENIC
  - Basic Flow was compatible with existing endpoints
  - Transfer @DENIC is custom but worked with small additions
  - New OAuth2 Login Flow instead of the proprietary DENIC login flow
- Afnic
  - Very simple “RPP” server in Python (and curl + a Zig program as the client)

Deployment - RPP/EPP Proxy



Working implementations:

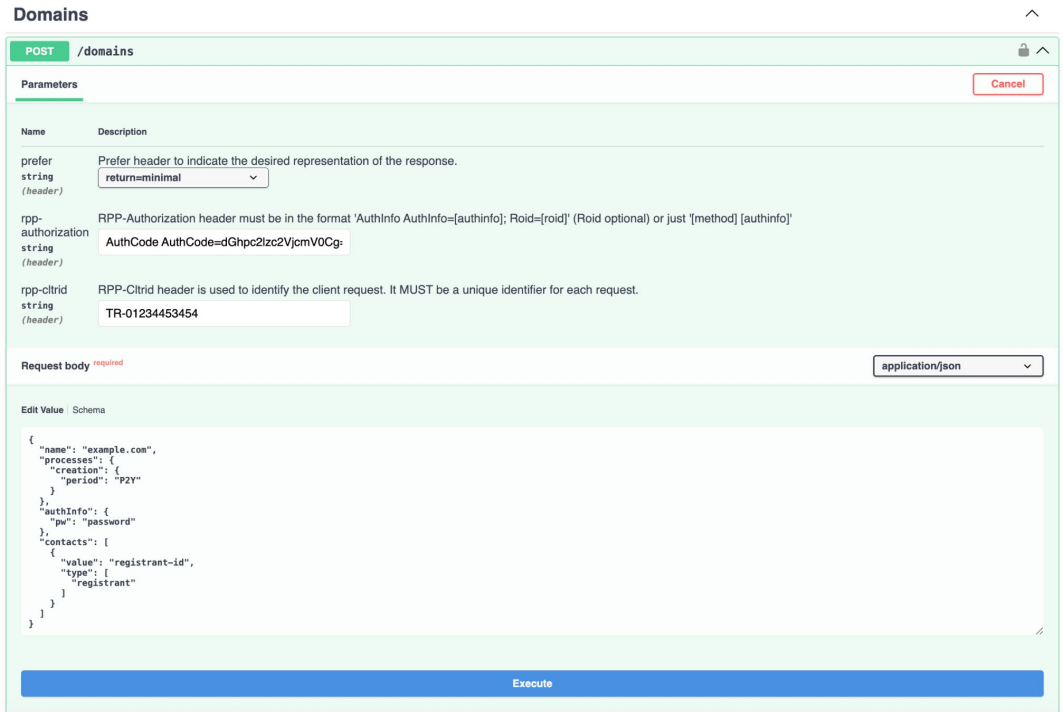
- SIDN
  - Py-rpp: RPP to EPP adapter for compatibility testing, Most EPP commands working.
- RPP Testbed
  - Working CREATE/INFO/DELETE for Domains and Contacts with xpanel/epp open source server
- Q
  - Added RPP frontend to epp-proxy
  - Check, create, info, and delete implemented for domains, contacts, and hosts
  - Tested against the real Identity Digital EPP server

Results

- 5 working implementations, 3 connected to real registry systems
- Aligned URL structure in the draft
- Quite a few different proposals to data representations of provisioning objects
- Added generic proposal for handling EPP security token
- Representation of error responses using Problem Detail documents

Sneak peek

From https://pawel-kow.github.io/ietf-rpp-api-typespec/



See also: https://sidn.github.io/py-rpp/swagger.html from py-rpp Project

Model - Domain

From https://pawel-kow.github.io/ietf-rpp-api-typespec/

```
{
  "name": "example.com",
  "processes": {
    "creation": {
      "period": "P2Y"
    }
  },
  "authInfo": {
    "pw": "password"
  },
  "contacts": [
    {
      "value": "registrant-id",
      "type": [
        "registrant"
      ]
    }
  ]
}
```

Model - Contact - alternative 1

From https://pawel-kow.github.io/ietf-rpp-api-typespec/


```
{
    "id": "FO-XXXXXX-03",
    "name": "John Doe",
    "organisationName": "ACME Inc.",
    "contactType": "ORG",
    "address": {
        "street": ["123 Main St", "Line 2", "Line 3"],
        "city": "Anytown",
        "postalCode": "12345",
        "stateProvince": "PA",
        "country": "US"
    },
    "email": ["foo@example.com"],
    "phone": ["+49.123123123x123"],
    "fax": ["+49.123123123x123"],
    "authInfo": {
        "pw": "Password1!@"
    }
}
```

Model - Contact - alternative 2 (JSContact)

From https://sidn.github.io/py-rpp/swagger.html


```
{
  "@type": "RPPContact",
  "card": {
    "@type": "Card",
    "version": "2.0",
    "rpp.ietf.org:id": "FO-FOOOOO-03",
    "name": {
      "full": "John Doe"
    },
    "organizations": {
      "o1": {
        "name": "ACME Inc."
      }
    },
    "addresses": {
      "a1": {
        "components": [
          { "kind": "name",
            "value": "123 Main StLine"
          },
          { "kind": "number",
            "value": "3"
          },
          { "kind": "postcode",
            "value": "12345"
          },
          { "kind": "region",
            "value": "PA"
          },
          { "kind": "locality",
            "value": "Anytown"
          }
        ],
        "countryCode": "US"
      }
    },
    "phones": {
      "voice": {
        "features": {
          "voice": true
        },
        "number": "+49.123123123x123"
      },
      "fax": {
        "features": {
          "fax": true
        },
        "number": "+49.123123123x123"
      }
    },
    "emails": {
      "e1": {
        "address": "foo@example.com"
      }
    }
  },
  "authInfo": {
    "value": "Password1!@"
  }
}
```

Future work

- Publicly available test environment for testing the latest RPP specification(s);
  - Online running sandbox (Cloud) -> DONE
  - Docker
- Finalize requirements document
- Design of API spec and data model
- Update existing prototypes

References

Drafts:

- https://datatracker.ietf.org/doc/draft-ietf-rpp-requirements/
- https://datatracker.ietf.org/doc/draft-kowalik-rpp-architecture/
- https://datatracker.ietf.org/doc/draft-wullink-rpp-core/
- https://datatracker.ietf.org/doc/draft-simmen-rpp-dns-data/

Open API Specifications (both WIP):

- https://pawel-kow.github.io/ietf-rpp-api-typespec/
- https://sidn.github.io/py-rpp/swagger.html

References

Implementations:

- RPP API in TypeSpec https://github.com/pawel-kow/ietf-rpp-api-typespec
- Q - RPP frontend to epp-proxy https://github.com/AS207960/epp-proxy/tree/root/src/rpp
- RPP/EPP Proxy Python: https://github.com/pawel-kow/ietf-rpp-api-epp-proxy-python
- PyRPP (RPP/EPP proxy, tested with SIDN and Nominet systems): https://github.com/SIDN/py-rpp
- Afnic server software https://github.com/bortzmeyer/RPP-Afnic

Instances:

- Afnic test server: https://wiki.ietf.org/en/group/rpp/hackathon/hackathon-123
- RPP Testbed: https://rpp.paulonet.eu/ui/

Thank you