Open-source CDMI proxy server Stoxy

Ilya Livenson
PDC KTH
Background & motivation

- Started with FP7 VENUS-C EU project
  - Partners included MS and regional data centers
  - Common data access layer was needed
    - Proprietary protocol was never considered
  - Initial version of CDMI was out
  - Went for that (CDMI-Proxy)
CDMI-Proxy

- Azure Storage
- S3 and SQS
- POSIX AMQP

Synchronized metadata layer

Metadata

Mobile device

Desktop

Cloud premises
Background & motivation (2)

- Currently working on a “v2”
  - From scratch
  - New name – Stoxy (STOrage proXY)

- Partially funded by EGI project, KTH university and OpenNode LLC

- Use cases
  - AWS S3-like storage
  - Data migration across clouds
  - Extension of missing backend functionality
  - Data source integration (CKAN-like)
Technology

- Python + Twisted + ZODB + ZCA
- Buildout packaging
- Multiplatform
Stozyy architecture

- CDMI endpoint
- SSH endpoint

Stozyy plugin
- OMS management framework
- Zope DB

Internal URI: protocol://logical/path/to/the/object

Value Backend A (local)
Value Backend B (aws)
Data streaming

- Typical goals – reduction of memory and disk footprint

Requirement for hashes of the full content in headers make streaming impossible (e.g. MS Azure Blob API)!
Data vs Metadata

- Sizes are not clear any more
  - But use cases are still there
- Stoxy uses internal references
  - protocol://username:password@server/path
- Internal resolver
  - Using named adapters in Twisted
  - Adapter has access to full information
  - Potentially replicating metadata to a value store (e.g. filename.metadata)
CDMI Security

- CDMI could be used on the application level (S3 use cases), but also for managing the underlying storage system
- Different credentials are used for those cases (in the setting of the academic infrastructure)
  - Username/passwords
  - client/server x509 + VOMS certificates
  - Token-based (OpenStack Keystone)
- CDMI unfortunately is not really specific about authN
Stoxy authentication

- HTTP Basic/Digest
- X509 client certificate based
  - Via Apache HTTPD
- Keystone authentication
  - Breaking a CDMI standard a bit…
Keystone token example

```
{
  "access": {
    "metadata": {
      ....metadata goes here....
    },
    "serviceCatalog": [
      ....endpoints goes here....
    ],
    "token": {
      "expires": "2013-05-26T08:52:53Z",
      "id": "placeholder",
      "issued_at": "2013-05-25T18:59:33.841811",
      "tenant": {
        "description": null,
        "enabled": true,
        "id": "925c23eafe1b4763933e08a4c4143f08",
        "name": "user"
      }
    },
    "user": {
      ....userdata goes here....
    }
  }
}
```
Explorative browsing with SSH

- Stoxy-specific functionality
- Comes with OMS framework
- Not CDMI interface
- Use putty for operating on the Stoxy objects/containers
- Very nice for debugging/browsing
Supported CDMI feature set

- Blob CRUD
- Container CRUD
  - Multiple sub levels
- ACL – planned to be full
  - At the moment requires admin privileges
Planned feature set

- Accent on server-side processing
  - To simplify the life for the client
  - Data lifecycle related operations
  - Data migration related operations
    - Not fully CDMI-compliant

- Accounting/Billing
Hope it will work 😊
CDMI issues

- Proxy
  - Checksum algorithms
    - Azure Blob schema expects a header with a hash, which includes derivatives of the full blob – proxy streaming is impossible (buffer == full blob)
- Executing actions
  - Direct – via HTTP verbs/queue mechanism
  - Expecting a side effect – via update of the object parameters
  - Really hard when writing clients
    - We have introduced another way
    - Custom ‘extensions’ PUT /path/to/action?args=…&…
Summary

- Open Source CDMI server (Apache2 license)
  - https://github.com/stoxy
  - Includes python client sdk/cli
- Accent on metadata manipulation and integration use cases
- Currently in active development, first production release is planned in 2-3 months (latest Dec 2014)
- Feedback/contributions are very welcome!
Thank you!

- And many thanks to
  - EGI Federated Cloud TF group
  - Andrei Sosnin, OpenNode
  - PDC KTH