CS 683 Emerging Technologies
Fall Semester, 2008
Doc 11 S3
Oct 9 2008

References

SOAP Tutorial, http://www.w3schools.com/soap/default.asp

Amazon Simple Storage Service Developer Guide API Version 2006-03-01, http://docs.amazonwebservices.com/AmazonS3/2006-03-01/

Amazon Simple Storage Service Getting Started Guide
http://docs.amazonwebservices.com/AmazonS3/2006-03-01/gsg/

REST, http://en.wikipedia.org/wiki/Representational_State_Transfer

erlaws, http://code.google.com/p/erlaws/

aws, http://timkay.com/aws/

Data Consistency Model

Updates to a single object at a key in a bucket are atomic

But a read after a write may return the old value

- Changes may take time to progate

No object locking

- If two writes to same object occur at the same time
- The one with later timestamp wins

Simple Storage System - S3

Store data on Amazon's servers

Objects
- Stored in buckets
- Accessed by key

Low level access
- SOAP
- REST

High level access
- Java
- C#
- Perl
- PHP
- Ruby

S3 Costs

Storage

$0.15 per GB-Month of storage used

Data Transfer

$0.100 per GB – all data transfer in

$0.170 per GB – first 10 TB / month data transfer out
$0.130 per GB – next 40 TB / month data transfer out
$0.110 per GB – next 100 TB / month data transfer out
$0.100 per GB – data transfer out / month over 150 TB

Requests

$0.01 per 1,000 PUT, POST, or LIST requests
$0.01 per 10,000 GET and all other requests

Objects

Objects contain
- Object data
- Metadata

Size
- 1 byte to 5 gigabytes per object

Object data
- Just bytes
- No meaning associated with bytes

Metadata
- Name-value pairs to describe the object
- Some http headers used
  - Content-Type

Buckets

Namespace for objects

No limitation on number of object per bucket

Only 100 buckets per account

Each bucket has a name

- Up to 255 bytes long
- Cannot be same as existing bucket name by any S3 user

Bucket Names

Bucket names must

- Contain lowercase letters, numbers, periods (.), underscores (_), and dashes (-)
- Start with a number or letter
- Be between 3 and 255 characters long
- Not be in an IP address style (e.g., "192.168.5.4")

To conform with DNS requirements, Amazon recommends

- Bucket names should not contain underscores (_)
- Bucket names should be between 3 and 63 characters long
- Bucket names should not end with a dash
- Bucket names cannot contain dashes next to periods (e.g., "my-.bucket.com" and "my.-bucket" are invalid

The java libraries from Amazon enforce the recommendations, so if you are using that library the recommendations are required.

Key

Unique identifier for an object within a bucket

Object Url

http://buckerName.s3.amazonaws.com/Key

http://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl

Bucket = doc
Key = 2006-03-01/AmazonS3.wsdl

Keys, Prefix & Delimiters

Keys can have prefixes

Separated by a delimiter
(/ default delimiter)

Can search for keys using prefix

UTF-8 encoding

Limited to 1024 bytes

Example

Bucket	Key	Object
cs683	lectures/Introduction	Introduction.pdf
cs683	lectures/cloudcomputing/introduction	CloudComputing.pdf
cs683	lectures/erlang/Sequential	ErlangSequential.pdf
cs683	lectures/erlang/genserver	genServer.pdf

Access Control Lists (ACL)

Each Bucket has an ACL

- Determines who has read/write access

Each Object can have an ACL

- Determines who has read/write access

ACL consists of a list of grants

Grant contains

- One grantee
- One permission

Grantees

Five Types

Owner

User by E-mail

- Email of S3 account
- They use their keys to access the bucket/object

User by Canonical Representation

- Each S3 account has a Canonical ID

AWS User Group

- Any S3 account

Anonymous Group

- Anyone

Permissions

Permission	Bucket	Object
READ	List the bucket	Read object data & metadata
WRITE	Create, overwrite & delete any object in bucket	Does not apply
READ_ACP	Read ACL of bucket	Read ACL of object
WRITE_ACP	Overwrite ACP of bucket	Overwrite ACP of object
FULL_CONTROL	All of the above	All of the above

Bucket/Object owners always have READ_ACP and WRITE_ACP

Using ACL

Default ACL

Grants owner FULL_CONTROL

Overwriting an object overwrites the object's ACL

You can overwrite ACL without modifying an object

Authentication

AWS Access Key ID

- Key given to developer

AWS Secret Access Key

- Used to generate signature for request

Signature

- HMAC-SHA1 hash of request information
- Uses Secret Access key to generate hash

Accessing S3

Two interfaces to S3

REST
SOAP

SOAP

Simple Object Access Protocol

XML based protocol

Used to transfer data

Uses http for transport

WSDL used to describe SOAP interface

Two mode of use

- RPC
- Document

SOAP Request contains

http headers
http body
- XML
- Envelope
- Header (optional)
- Body

Example SOAP Request

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPrice>
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```


Example from http://www.w3schools.com/soap/soap_example.asp

SOAP Response

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPriceResponse>
      <m:Price>34.5</m:Price>
    </m:GetStockPriceResponse>
  </soap:Body>
</soap:Envelope>
```

Example - Create Bucket

Request without http headers

```
<CreateBucket xmlns="http://doc.s3.amazonaws.com/2006-03-01">
<Bucket>quotes</Bucket>
<AWSAccessKeyId>1D9FVRAYCP1VJEXAMPLE=</AWSAccessKeyId>
<Timestamp>2006-03-01T12:00:00.183Z</Timestamp>
<Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

Response with http headers

```
<CreateBucketResponse xmlns="http://doc.s3.amazonaws.com/2006-03-01">
<CreateBucketResponse>
<Bucket>quotes</Bucket>
</CreateBucketResponse>
</CreateBucketResponse>
```


The examples are from Amazon Simple Storage Service Developer Guide API Version 2006-03-01. They are not legal SOAP requests. I don't know if the example are complete or they are not using legal SOAP. In SOAP Envelopes and body are required. I suspect that they are leaving the envelope and body off the examples to keep the examples readable.

REST

REpresentational State Transfer

Definition 1

Application state and functionality are abstracted into resources

Every resource is uniquely addressable using a universal syntax

All resources share a uniform interface for the transfer of state

- A constrained set of well-defined operations
- A constrained set of content types, optionally supporting code on demand
- A protocol which is:
  - Client-server
  - Stateless
  - Cacheable
  - Layered

Source Wikipedia, http://en.wikipedia.org/wiki/Representational_State_Transfer

REST

Definition 2

Any simple interface which transmits domain-specific data over HTTP

Example - Create a Bucket

Request

```
PUT / HTTP/1.1
Host: colorpictures.s3.amazonaws.com
Content-Length: 0
Date: Wed, 01 Mar 2006 12:00:00 GMT
Authorization: AWS 15B4D3461F177624206A:xQE0diMbLRepdf3YB+FIEXAMPLE=
```

Response

```
HTTP/1.1 200 OK
x-amz-id-2: YgIPIfBiKa2bj0KMg95r/
0zo3emzU4dzsD4rcKCHQUAdQkf3ShJTOOpXUueF6QKo
x-amz-request-id: 236A8905248E5A01
Date: Wed, 01 Mar 2006 12:00:00 GMT
Location: /colorpictures
Content-Length: 0
Connection: close
```

Accessing S3

SOAP & REST are used to develop libraries to access S3

S3 libraries exist in many languages

We will look at:

- Java
- Perl
- Erlang

erlaws

Project home - http://code.google.com/p/erlaws/

Functions

list_buckets()
create_bucket(Bucket)
create_bucket(Bucket,eu)
delete_bucket(Bucket)
list_contents(Bucket)
list_contents(Bucket, Options)
put_object(Bucket, Key, Data, ContentType, Metadata)
get_object(Bucket, Key)
info_object (Bucket, Key)
delete_object (Bucket, Key)

Installing

Download

svn checkout http://erlaws.googlecode.com/svn/trunk/ erlaws

Compile (in erlaws director)

erl -make

Put in erlang path (in .erlang in home directory)

code:add_pathz("/Users/whitney/Courses/683/Fall08/erlangCode/erlaws/ebin").

Sample Use

```
1> application:start(crypto).
ok
2> application:start(inets).
ok

3>  S3 = erlaws_s3:new("YourKey","YourSecretKey",false).
{erlaws_s3,"YourKey","YourSecretKey",false}

4> S3:list_buckets().
{ok,["cs683","roger-.whitney","roger_.whitney",
     "rogerwhitney","rogerwhitneyearlsdsu","sdsu"],
    {requestId,"8A5DA6197F370C4F"}}

5. Lecture = S3:get_object("cs683","lectures/Introduction").
{ok,<<"%PDF-1.3\n%?????????\n4 0 obj\n<< /Length 5 0 R /
Filter /FlateDecode >>\nstream\nx"...>>,
    {requestId,"8C58D0B262171331"}}
```


The application crypto and inets must be started before using erlaws_s3. erlaws_s3 is a parameterized module so you have to instantiate the module first. The third parameter to erlaws_s3:new indicates if one should use HTTPS or not (false meaning no - use HTTP). HTTPS did not work for me.

Perl - aws

Project home http://timkay.com/aws/

Command	Action
s3ls	List all buckets
s3ls [-X] BUCKET [PREFIX]	List objects in given bucket If PREFIX is given restrict to object whose keys start with PREFIX
s3put BUCKET	Create a bucket
s3put BUCKET/OBJECT FILE	put FILE in BUCKET with key OBJECT
s3get BUCKET/OBJECT [FILE]	get the object in Bucket with key OBJECT
s3delete BUCKET/OBJECT	Delete object in BUCKET with key OBJECT

See http://timkay.com/aws/ for more command & options, and installation instructions

Example Usage

Al pro 72->s3ls

```
+----------------------+--------------------------+
|         Name         |       CreationDate       |
+----------------------+--------------------------+
| cs683                | 2008-10-09T02:25:17.000Z |
| roger-.whitney       | 2008-10-09T01:39:18.000Z |
| roger_.whitney       | 2008-10-09T01:38:56.000Z |
| rogerwhitney         | 2008-10-09T01:41:04.000Z |
| rogerwhitneyearlsdsu | 2008-10-09T01:31:03.000Z |
| sdsu                 | 2008-10-09T02:25:03.000Z |
+----------------------+--------------------------+
```

Al pro 73->s3put foo-bar

Al pro 74->s3put foo-bar/sample CloudComputing.pdf

Al pro 75->s3ls foo-bar

```
+---------+--------+--------+---------+-------------+--------+--------------------------+--------+--------------+
|  Name   | Prefix | Marker | MaxKeys | IsTruncated |  Key   |       LastModified       |  Size  | StorageClass |
+---------+--------+--------+---------+-------------+--------+--------------------------+--------+--------------+
| foo-bar |        |        | 1000    | false       |        |                          |        |              |
|         |        |        |         |             | sample | 2008-10-09T20:11:19.000Z | 200584 | STANDARD     |
+---------+--------+--------+---------+-------------+--------+--------------------------+--------+--------------+
```

ACL - Reading

Al pro 18->s3get --xml 'cs683?acl'

```
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
        <Owner>
                <ID>80098d9063cdf686171a90db1828e1af49c1995942b5eae87a308bec845ecf54</ID>
                <DisplayName>whitney201</DisplayName>
        </Owner>
        <AccessControlList>
                <Grant>
                        <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
                                <ID>80098d9063cdf686171a90db1828e1af49c1995942b5eae87a308bec845ecf54</ID>
                                <DisplayName>whitney201</DisplayName>
                        </Grantee>
                        <Permission>FULL_CONTROL</Permission>
                </Grant>
        </AccessControlList>
</AccessControlPolicy>
```

Adding Second Grant

acl.xml

```
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Owner>
        <ID>80098d9063cdf686171a90db1828e1af49c1995942b5eae87a308bec845ecf54</ID>
        <DisplayName>whitney201</DisplayName>
    </Owner>
    <AccessControlList>
        <Grant>
            <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">
                <ID>80098d9063cdf686171a90db1828e1af49c1995942b5eae87a308bec845ecf54</ID>
                <DisplayName>whitney201</DisplayName>
            </Grantee>
            <Permission>FULL_CONTROL</Permission>
        </Grant>
        <Grant>
            <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
                <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
            </Grantee>
            <Permission>READ</Permission>
        </Grant>
    </AccessControlList>
</AccessControlPolicy>
```

Al pro 42->s3put 'cs683?acl' acl.xml

See Using Amazon S3, Authentication and Access Control Lists in Amazon Simple Storage Service Developer Guide API Version 2006-03-01 (page 31) for XML details on the various grants.

Java

Available in Amazon Simple Storage Service Getting Started Guide
http://docs.amazonwebservices.com/AmazonS3/2006-03-01/gsg/

"Preparing the Samples" section of "Working with Amazon S3"

Example

```
import com.amazon.s3.AWSAuthConnection;
import com.amazon.s3.CallingFormat;
import com.amazon.s3.QueryStringAuthGenerator;
import com.amazon.s3.S3Object;
import com.amazon.s3.Response;
import com.amazon.s3.ListBucketResponse;

public class BucketTests extends Object
{
    static final String awsAccessKeyId = "YourKey";
    static final String awsSecretAccessKey = "YourSecretKey";
```


Example modified from tutorial examples

Creating a Bucket

```
public static void main(String args[]) throws Exception {
    AWSAuthConnection conn =
      new AWSAuthConnection(awsAccessKeyId, awsSecretAccessKey);

    if (!conn.checkBucketExists("cs683"))
     {
        System.out.println("----- creating bucket -----");
        Response bucketCreated =
          conn.createBucket("cs683", AWSAuthConnection.LOCATION_DEFAULT, null);
    System.out.println(bucketCreated.connection.getResponseMessage());
}
```

Listing Bucket Contents

```
System.out.println("----- listing bucket -----");
ListBucketResponse allObjects = conn.listBucket("cs683", null, null, null, null);
System.out.println(allObjects.entries);


System.out.println("----- listing bucket with prefixes -----");
ListBucketResponse erlang = conn.listBucket("cs683", "lectures/erlang", null, null, null);
```

```
        System.out.println(erlang.entries);

        System.out.println("----- listing all my buckets -----");
        System.out.println(conn.listAllMyBuckets(null).entries);
    }
}
```

Output

```
----- listing bucket -----
[lectures/Introduction, lectures/cloudcomputing/introduction, lectures/erlang/Sequential,
lectures/erlang/genserver]
----- listing bucket with prefixes -----
[lectures/erlang/Sequential, lectures/erlang/genserver]
----- listing all my buckets -----
[cs683, foo-bar, roger-.whitney, roger_.whitney, rogerwhitney, rogerwhitneyearlsdsu, sdsu]

Program exited with status 0.
```

Creating a Bucket Using Java

```
AWSAuthConnection conn =
    new AWSAuthConnection("[aws-access-key-id]", "[aws-secret-access-key-id]");

Map headers = null;
Response response = conn.createBucket("[bucket-name]", headers);
if (response.connection.getResponseCode() == 200) {
    // bucket was created
} else {
    // something bad happened
```

Writing an Object

```
AWSAuthConnection conn =
    new AWSAuthConnection("[aws-access-key-id]", "[aws-secret-access-key-id]");

Map metadata = null;
3Object simpleObject = new S3Object("this is a test".getBytes(), metadata);
Map headers = null;
Response response = conn.put("[bucket-name]", "[key-name]", simpleObject, headers);
```

Writing a ACL

Methods in AWSAuthConnection class

public Response putACL(String bucket, String key, String aclXMLDoc, Map headers)

public Response putBucketACL(String bucket, String aclXMLDoc, Map headers)

Reading an Object

```
AWSAuthConnection conn =
    new AWSAuthConnection("[aws-access-key-id]", "[aws-secret-access-key-id]");

Map headers = null;
GetResponse response = conn.get("[bucket-name]", "[key-name]", headers);
String value = response.object.data;
Map metadata = response.object.metadata;
List values = (List)metadata.get("title");
String title = (String)values.get(0);
```