Table of Contents

Amazon S3 compatible service

Public access S3 bucket

```
{
   "Version" : "2012-10-17",
   "Statement" : [
      {
         "Effect" : "Allow",
         "Principal" : {
            "AWS" : [ "*" ]
         },
         "Action" : [ "s3:GetBucketLocation", "s3:ListBucket" ],
         "Resource" : [ "arn:aws:s3:::public" ]
      },
      {
         "Effect" : "Allow",
         "Principal" : {
            "AWS" : [ "*" ]
         },
         "Action" : [ "s3:GetObject" ],
         "Resource" : [ "arn:aws:s3:::public/*" ]
      }
   ]
}
```

Access by username/password

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "allow-username-and-password-access",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::BUCKETNAME/*",
            "Condition": {
                "StringEquals": {
                    "aws:UserAgent": "USERNAME",
                    "aws:Referer": "PASSWORD"
                }
            }
        }
    ]
```

```
curl --user-agent USERNAME --referer PASSWORD --upload-file "FILENAME" --request PUT
"https://s3-AWS_REGION.amazonaws.com/BUCKETNAME/FILENAME"
```

```
curl --user-agent USERNAME --referer PASSWORD
"https://s3-AWS_REGION.amazonaws.com/BUCKETNAME/FILENAME" > FILENAME
```

Generate time expiration link

```
<?php

  if(!function_exists('el_crypto_hmacSHA1')){
    /**
    * Calculate the HMAC SHA1 hash of a string.
    *
    * @param string $key The key to hash against
    * @param string $data The data to hash
    * @param int $blocksize Optional blocksize
    * @return string HMAC SHA1
    */
    function el_crypto_hmacSHA1($key, $data, $blocksize = 64) {
        if (strlen($key) > $blocksize) $key = pack('H*', sha1($key));
        $key = str_pad($key, $blocksize, chr(0x00));
        $ipad = str_repeat(chr(0x36), $blocksize);
        $opad = str_repeat(chr(0x5c), $blocksize);
        $hmac = pack( 'H*', sha1(
        ($key ^ $opad) . pack( 'H*', sha1(
          ($key ^ $ipad) . $data
        ))
      ));
        return base64_encode($hmac);
    }
  }

  if(!function_exists('el_s3_getTemporaryLink')){
    /**
    * Create temporary URLs to your protected Amazon S3 files.
    *
    * @param string $accessKey Your Amazon S3 access key
    * @param string $secretKey Your Amazon S3 secret key
    * @param string $bucket The bucket (bucket.s3.amazonaws.com)
    * @param string $path The target file path
    * @param int $expires In minutes
    * @return string Temporary Amazon S3 URL
    * @see http://awsdocs.s3.amazonaws.com/S3/20060301/s3-dg-20060301.pdf
    */
    function el_s3_getTemporaryLink($accessKey, $secretKey, $bucket, $path, $expires = 5) {
      // Calculate expiry time
      $expires = time() + intval(floatval($expires) * 60);
      // Fix the path; encode and sanitize
      $path = str_replace('%2F', '/', rawurlencode($path = ltrim($path, '/')));
      // Path for signature starts with the bucket
      $signpath = '/'. $bucket .'/'. $path;
      // S3 friendly string to sign
      $signsz = implode("\n", $pieces = array('GET', null, null, $expires, $signpath));
      // Calculate the hash
      $signature = el_crypto_hmacSHA1($secretKey, $signsz);
      // Glue the URL ...
      $url = sprintf('http://%s.s3.amazonaws.com/%s', $bucket, $path);
      // ... to the query string ...
      $qs = http_build_query($pieces = array(
        'AWSAccessKeyId' => $accessKey,
        'Expires' => $expires,
        'Signature' => $signature,
      ));
      // ... and return the URL!
      return $url.'?'.$qs;
    }
```

```
  }

?>
```

```
<?php echo el_s3_getTemporaryLink('your-access-key', 'your-secret-key', 'bucket-name',
'/path/to/file.mov'); ?>
```

S3 access using CLI

```
aws configure
aws --endpoint-url=http://ceph-RGW-IP:7480 s3api list-buckets
```

S3 LifeCycle

```
{
    "Rules": [
        {
            "ID": "Removing all old versions",
            "Filter": {
                "Prefix": "/"
            },

            "Expiration": {
                "ExpiredObjectDeleteMarker": true
            },

            "DelMarkerObjectExpiration": {
                "Days": 10
            },

            "NoncurrentVersionExpiration": {
                "NewerNoncurrentVersions": 1,
                "NoncurrentDays": 1,
                "expiredObjectDeleteMarker": "true"
            },
            "Status": "Enabled"
        }
    ]
}
```

```
/mnt/sdc/mc mirror s3/public local/backup --overwrite --remove
/mnt/sdc/mc rm local/backup --non-current --recursive --versions --older-than 31d0hh0mm0ss --
force
```