一、设计需求

Implement a storage array simulator as a web service or application. It provides SOAP or REST API to users. The following functions are supported:

1. Create one or multiple LUN’s.

2. Resize a LUN.

3. Remove a LUN.

4. Retrieve the information (size) of a LUN.

5. Concurrent requests.

6. Data persistence.

General requirements:

· Implement as many functions above as possible.

· Implement necessary unit tests.

· A complete set of documents required.

· Submit source code and all artifacts to GitHub.

二、对需求的分析

1. Storage array simulator
2. 相关的操作：

* Create one or multiple LUN’s.
* Resize a LUN.
* Remove a LUN.
* Retrieve the information (size) of a LUN.

1. 阵列的相关信息

* 阵列空间大小
* 已用空间大小
* 阵列中的LUN信息(ID和大小)

1. 阵列中LUN的相关信息

* LUN的ID
* LUN的大小

1. Web service or application. Provides SOAP or REST API to users.
2. Create one or multiple LUN’s.

* 查看阵列剩余空间大小
* 查看输入LUN的大小是否合法(大于0且小于阵列剩余空间大小)；
* 如果合法，则创建成功，返回LUN的相关信息(LUN的ID, LUN的大小，LUN的状态)；调整阵列的存储空间，更新阵列中该LUN的相关信息；
* 如果不合法，则返回具体失败信息

1. Resize a LUN.

* 查看阵列剩余空间大小
* 调整LUN的大小合法(大于0且小于阵列剩余空间大小)；
* 如果合法，则调整成功，返回新的LUN的相关信息(LUN的ID, LUN的大小，LUN的状态)；调整阵列的存储空间，更新阵列中该LUN的相关信息；
* 如果不合法，则返回具体失败信息

1. Remove a LUN.

* 查看阵列中以有的LUN
* 被删除的LUN是否在阵列已有LUN列表中
* 如果在，执行删除操用。如果操作执行成功，返回操成功消息，调整阵列的存储空间，，更新阵列中LUN的相关信息
* 如果不在，则返回具体失败信息

1. Retrieve the information (size) of a LUN.

* 查看阵列中以有的LUN
* 检索该LUN是否在阵列已有LUN列表中
* 如果在，根据LUN ID,返回LUN大小
* 如果不在，返回具体失败信息

1. Concurrent requests.

* 针对并发请求，只对第一个请求进行响应，响应时，创建文件锁，如果文件存在，其它操请求拒绝，并返回具体信息
* 第一个请求完成时，删除文件锁

1. Data persistence

将存储阵列的相关信息(阵列空间大小、剩余空间大小、阵例中LUN的相关信息(ID和大小))存储在文件中

三、 需求设计

1. 根据需求分析，web service可执行5个操作：
2. create \_lun(lunsize,lunnumber)
3. resize\_un(lunid, lunsize)
4. delelte\_un(lunid)
5. list\_Lun(lunid)
6. storage\_array\_persistence ()
7. web service 接口
8. 增

* 接口： /create\_lun
* 接口方法: put
* 接口参数:一个或多个 size
* 返回信息：[{lunId: size},…,{returnCode:message}]
* 返回信息说明：

lunId: 创建LUN时，阵列分配的ID

size： 创建的LUN的大小

returnCode: 操作返回代码

message: 操作返回的信息

0：操作成功

1：输入参数错误(不是整数)

2: 输入参数错误(size必须大于0)

3： 阵列剩余存储空间不足

4: 创建该LUN失败

1. 删

* 接口： /delele\_lun
* 接口方法: delete
* 接口参数:一个lunid
* 返回信息：[{lunId: size},{returnCode:message}]
* 返回信息说明：

lunId: 创建LUN时，阵列分配的ID

size： 创建的LUN的大小

returnCode: 操作返回代码

message: 操作返回的信息

0：操作成功

1： 输入参数错误(不是整数)

2：输入参数错误(lunid必须大于等于0)

3： 在阵列中无法找到该LUN

1. 改

* 接口： /resize\_lun
* 接口方法: post
* 接口参数:lunid, size
* 返回信息：[{lunId: size},{returnCode:message}]
* 返回信息说明：

lunId: 阵列创建该LUN时分配的ID

size： 创建的LUN的大小

returnCode: 操作返回代码

message: 操作返回的信息

0：操作成功

1： 输入参数错误(lunid不是整数或size不是整数)

2: 输入参数错误(lunid必须大于0,size必须大于0)

3: 在阵列中无法找到该LUN

4： 阵列剩余存储空间不足

1. 查

* 接口： /list\_lun
* 接口方法: get
* 接口参数:一个lunid
* 返回信息：[{lunId: size},{returnCode:message}]
* 返回信息说明：

lunId: 阵列创建该LUN时分配的ID

returnCode: 操作返回代码

message: 操作返回的信息

0：操作成功

1： 输入参数错误(不是整数)

2： 输入参数错误（lunid必须大于等于0）

3： 在阵列中无法找到该LUN

1. 数据持久性

* 接口： /storage\_array\_persistece
* 接口方法: get
* 接口参数:
* 返回信息：[{returnCode:message}]
* 返回信息说明：

lunId: 阵列创建该LUN时分配的ID

returnCode: 操作返回代码

message: 操作返回的信息

0：操作成功

四、web service实现方式

1. 使用Flask这个轻量级框架
2. 使用storageArrrayController.py实现模拟存储阵例可进行的5种接口操作
3. 命名storageArrrayServer.py实现模拟存储阵列和LUN
4. 使用storageArrayUnitTest.py进行接口测试
5. 使用storageArrayLock文件锁来判断是否并发
6. Unit\_test\_log\_file.txt 为4中测试的log

五、手动接口测试命令

1. 测试创建LUN

[root@n213-h54 Test3]# **curl -i -X PUT -d '{"lunsize":"3","lunnumber":"10"}' -H "Content-Type:application/json" http://127.0.0.1:5000/create\_lun**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 270

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:13:33 GMT

[{"2017032319133358": 3, "2017032319133369": 3, "2017032319133340": 3, "2017032319133350": 3, "2017032319133347": 3, "2017032319133397": 3, "2017032319133385": 3, "2017032319133308": 3, "2017032319133381": 3, "2017032319133391": 3}, {"0": "The LUN create successfully"}]You have new mail in /var/spool/mail/root

1. 测试查看LUN

[root@n213-h54 Test3]# **curl -i -X GET -d '{"lunid":"2017032319133358"}' -H "Content-Type:application/json" http://127.0.0.1:5000/list\_lun**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 69

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:14:04 GMT

[{"2017032319133358": 3}, {"0": "List LUN information seccussfully"}]

1. 测试resize LUN(改小)

[root@n213-h54 Test3]# **curl -i -X POST -d '{"lunid":"2017032319133358","lunsize":"2"}' -H "Content-Type:application/json" http://127.0.0.1:5000/resize\_lun**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 63

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:14:38 GMT

[{"2017032319133358": 2}, {"0": "Resize the LUN seccussfully"}]You have new mail in /var/spool/mail/root

1. 测试resize LUN(改大)

[root@n213-h54 Test3]# **curl -i -X POST -d '{"lunid":"2017032319133358","lunsize":"8"}' -H "Content-Type:application/json" http://127.0.0.1:5000/resize\_lun**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 63

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:15:05 GMT

[{"2017032319133358": 8}, {"0": "Resize the LUN seccussfully"}]

1. 测试删除LUN

[root@n213-h54 Test3]# **curl -i -X DELETE -d '{"lunid":"2017032319133358"}' -H "Content-Type:application/json" http://127.0.0.1:5000/delete\_lun**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 84

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:15:49 GMT

[{"2017032319133358": 8}, {"0": "Delete the LUN in the storage array seccussfully"}]You have new mail in /var/spool/mail/root

1. 测试persistence阵列和LUN信息

[root@n213-h54 Test3]# **curl -i -X GET -H "Content-Type:application/json" http://127.0.0.1:5000/storage\_array\_persistence**

HTTP/1.0 200 OK

Content-Type: text/html; charset=utf-8

Content-Length: 65

Server: Werkzeug/0.12.1 Python/2.7.5

Date: Thu, 23 Mar 2017 23:16:00 GMT

[{"0": "Persistence the storage array and LUN info seccuessfully"}]

1. 查看persistence后阵列和LUN的信息

[root@n213-h54 Test3]# **cat persistence\_file.txt**

all\_size: 100

unused\_size: 73

exist\_luns:

{'2017032319133369': 3, '2017032319133340': 3, '2017032319133350': 3, '2017032319150536': 6, '2017032319133347': 3, '2017032319143845': -1, '2017032319133397': 3, '2017032319133385': 3, '2017032319133308': 3, '2017032319133381': 3, '2017032319133391': 3}

六、遗留的问题

1. 在五的7中，查看阵列和LUN的永久性时，发现有一个LUN的大小竟然是-1,可能代码还存在问题
2. 关于并发处理的问题，占时想到的有创建文件锁，当有task执行阵列或LUN相关的操作时，创建一个storageArrayLock的文件，task结整时，删除这个文件；其它task检查到这个文件存在时，向前端返回失败。
3. 前端用AngularJS创建动态页面，模拟客户操作行为，进行功能和性能测试。