# Promise Test Cases

# Test Case 0: Pre-conditions

**Action:**

Step 0: Create an OpenStack project called “Promise”.

Step 1: Create a quota for “Promise” limiting the amount of VCPUs to 5  
Note: In this release the created quota will be taken as the total existing capacity by Promise (to be confirmed).

Step 2: Create a block storage quota for “Promise” limiting the amount of block storage to 20 GBytes.

# Test Case 1: Reservation of a VM for immediate use followed by allocation

**Action:**

Step 1: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Omit start time and end time.

**Result:**

Step 2: Verify successful response.

Step 3: Store the received reservationId.

**Action:**

Step 4: Create a compute resource using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and the compute resource’s attributes matching the ones used in the reservation.

**Result:**

Step 5: Verify successful response.

Step 6: Store received computeId.

**Action:**

Step 7: Terminate the compute resource using the received computeId.

**Result:**

Step 8: Verify successful response.

**Action:**

Step 9: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 10: Verify successful response.

# Test Case 2: Reservation of a VM followed by denial of service to another user and by allocation of reserved VM

**Action:**

Step 1: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 5 VCPUs. Omit start time and end time.

**Result:**

Step 2: Verify successful response.

Step 3: Store the received reservationId.

**Action:**

Step 4: Create a compute resource without using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation not containing the received reservationId.   
Note: the operation shall be sent as a user of the created “Promise” project in order to be constrained by the quota. This is a limitation of Promise in this release.

**Result:**

Step 5: Verify unsuccessful response.

**Action:**

Step 6: Create a compute resource using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and the compute resource’s attributes matching the ones used in the reservation.

**Result:**

Step 7: Verify successful response.

Step 8: Store received computeId.

**Action:**

Step 9: Terminate the compute resource using the received computeId.

**Result:**

Step 10: Verify successful response.

**Action:**

Step 11: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 12: Verify successful response.

# Test Case 3: Reservation of a VM for future use

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+200 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Store Current Time ‘T1’.

Step 6: Wait T0+200+1-T1 seconds.

Step 7: Create a compute resource using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and the compute resource’s attributes matching the ones used in the reservation.

**Result:**

Step 8: Verify successful response.

Step 9: Store received computeId.

**Action:**

Step 10: terminate the compute resource using the received computeId.

**Result:**

Step 11: Verify successful response.

**Action:**

Step 12: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 13: Verify successful response.

# Test Case 4: Update of an outstanding reservation, increase capacity

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+300 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Update the reservation adding one more VCPU.  
Description: Use the Update Compute Resource Reservation operation to modify the number of VCPUs to 4. Use the received reservationID.

**Result:**

Step 6: Verify successful response.

**Action:**

Step 7: Store Current Time ‘T1’.

Step 8: Wait T0+300+1-T1 seconds.

Step 9: Create a compute resource using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and the compute resource’s attributes matching the ones used in the updated reservation, i.e. 4 VCPUs.

**Result:**

Step 10: Verify successful response.

Step 11: Store received computeId.

**Action:**

Step 12: terminate the compute resource using the received computeId.

**Result:**

Step 13: Verify successful response.

**Action:**

Step 14: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 15: Verify successful response.

# Test Case 5: Update of an outstanding reservation, decrease capacity

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Update the reservation reducing the number of VCPUs to 2.  
Description: Use the Update Compute Resource Reservation operation to modify the number of VCPUs to 2. Use the received ReservationID.

**Result:**

Step 6: Verify successful response.

**Action:**

Step 7: Wait T0+400+1-T1 seconds.

Step 8: Create a compute resource without using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation not containing the received reservationId to create a VM with 3 VCPUs.  
Note: the operation shall be sent as a user of the created “Promise” project in order to be constrained by the quota. This is a limitation of Promise in this release.

**Result:**

Step 9: Verify successful response.

Step 10: Store received computeId.

**Action:**

Step 11: Store Current Time ‘T1’.

Step 12: Wait T0+400+1-T1 seconds.

Step 13: Create a compute resource using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and the compute resource’s attributes matching the ones used in the updated reservation, i.e. 2 VCPUs.

**Result:**

Step 14: Verify successful response.

Step 15: Store received computeId’.

**Action:**

Step 16: terminate the compute resource using the received computeId.

**Result:**

Step 17: Verify successful response.

**Action:**

Step 18: terminate the compute resource using the received computeId’.

**Result:**

Step 19: Verify successful response.

**Action:**

Step 20: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 21: Verify successful response.

# Test Case 6: Notification of reservation change

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 4 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Subscribe to notifications of changes affecting the reservation.  
Description: use the Subscribe operation with the received reservationId in order to get notified of changes that affect the reservation.

**Result:**

Step 6: Verify successful response.

**Action:**

Step 7: Reduce the quota for “Promise” limiting the amount of VCPUs to 3.  
Note: In this release the created quota will be taken as the total existing capacity by Promise (to be confirmed).

**Result:**

Step 8: Verify a Notification is received. The reservation is modified to 3 VCPUs.  
Note: alternatively, the notification could indicate that the reservation is no longer valid.

**Action:**

Step 9: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 10: Verify successful response.

**Action:**

Step 11: Restore the quota for “Promise” limiting the amount of VCPUs to 5.

# Test Case 7: Cancellation of a reservation

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Cancel the reservation.  
Description: use the Terminate Compute Resource Reservation operation with the receive reservationId.

**Result:**

Step 6: Verify successful response.

Step 7: Wait T0+400+1-T1 seconds.

**Action:**

Step 8: Create a compute resource without using the previous reservation.  
Description: use the Allocate Virtualised Compute Resource operation not containing the received reservationId to create a VM with 4 VCPUs.  
Note: the operation shall be sent as a user of the created “Promise” project in order to be constrained by the quota. This is a limitation of Promise in this release.

**Result:**

Step 9: Verify successful response.

Step 10: Store received computeId.

**Action:**

Step 11: Terminate the compute resource using the received computeId.

**Result:**

Step 12: Verify successful response.

# Test Case 8: Query of a reservation

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Query the reservation.  
Description: use the Query Compute Resource Reservation operation with the receive reservationId.

**Result:**

Step 6: Verify successful response and verify that attributes match the ones requested in Step 2.

**Action:**

Step 7: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 8: Verify successful response.

# Test Case 9: Create a bulk reservation of compute capacity

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a bulk compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a computePool with 5 VCPUs. Indicate ‘T0+200 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Wait T0+200+1-T1 seconds.

Step 6: Create a compute resource that consumes resources from the previous pool reservation  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and indicating 2 VCPUs.

**Result:**

Step 7: Verify successful response.

Step 8: Store received computeId.

**Action:**

Step 9: Create a second compute resource that consumes resources from the previous pool reservation  
Description: use the Allocate Virtualised Compute Resource operation containing the received reservationId and indicating 2 VCPUs.

**Result:**

Step 10: Verify successful response.

Step 11: Store received computeId’.

**Action:**

Step 12: Terminate the compute resource using the received computeId.

**Result:**

Step 13: Verify successful response.

**Action:**

Step 14: Terminate the compute resource using the received computeId’.

**Result:**

Step 15: Verify successful response.

**Action:**

Step 16: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 17: Verify successful response.

# Test Case 10: Rejection of a reservation due to lack of resources

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Create a second compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 4 VCPUs. Indicate ‘T0+400 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 6: Verify unsuccessful response.

**Action:**

Step 7: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 8: Verify successful response.

# Test Case 11: Reservation of block storage for future use

**Action:**

Step 1: Store Current Time ‘T0’.

Step 2: Create a storage resource reservation.  
Description: Use the Create Storage Resource Reservation operation to reserve a storagePool of 15 GBytes. Indicate ‘T0+200 s’ as start time and ‘T0+3800 s’ as end time.

**Result:**

Step 3: Verify successful response.

Step 4: Store the received reservationId.

**Action:**

Step 5: Store Current Time ‘T1’.

Step 6: Wait T0+200+1-T1 seconds.

Step 7: Create a block storage resource using the previous reservation.  
Description: use the Allocate Virtualised Storage Resource operation containing the received reservationId and indicating 15 GBytes as storage size.

**Result:**

Step 8: Verify successful response.

Step 9: Store received storageId.

**Action:**

Step 10: terminate the storage resource using the received storageId.

**Result:**

Step 11: Verify successful response.

**Action:**

Step 12: Terminate the storage resource reservation using the received reservationId.

**Result:**

Step 13: Verify successful response.

# Test Case 12: Capacity Management, query

**Action:**

Step 1: Discover the total compute capacity.  
Description: Use the Query Compute Capacity operation to find out what is the total capacity. As resource type use a compute resource with one VCPU. Specify that you are interested in the total capacity using the filter attribute.

**Result:**

Step 2: Verify the response indicates capacity of 5 .

**Action:**

Step 3: Create a compute resource.  
Description: use the Allocate Virtualised Compute Resource operation. Indicate that the compute resource has one VCPU.

**Result:**

Step 4: Verify successful response.

Step 5: Store received computeId.

**Action:**

Step 6: Discover the available compute capacity.  
Description: Use the Query Compute Capacity operation to find out what is the available capacity. As resource type use a compute resource with one VCPU. Specify that you are interested in the available capacity using the filter attribute.

**Result:**

Step 7: Verify the response indicates capacity of 4.

**Action:**

Step 8: Discover the used compute capacity.  
Description: Use the Query Compute Capacity operation to find out what is the used capacity. As resource type use a compute resource with one VCPU. Specify that you are interested in the used capacity using the filter attribute.

**Result:**

Step 9: Verify the response indicates capacity of 1.

**Action:**

Step 10: terminate the compute resource using the received computeId.

**Result:**

Step 11: Verify successful response.

**Action:**

Step 12: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 3 VCPUs. Omit start time and end time.

**Result:**

Step 13: Verify successful response.

Step 14: Store the received reservationId.

**Action:**

Step 15: Discover the reserved compute capacity.  
Description: Use the Query Compute Capacity operation to find out what is the reserved capacity. As computeResourceType use a compute resource with one VCPU. Specify that you are interested in the reserved capacity using the filter attribute.

**Result:**

Step 16: Verify the response indicates capacity of 3.

**Action:**

Step 17: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 18: Verify successful response.

# Test Case 13: Capacity Management, notification

**Action:**

Step 1: Subscribe to notifications of capacity changes.  
Description: use the Subscribe operation to get notifications of changes in capacity. Use the filter attribute to specify that changes in total, available, reserved and used capacity should be notified. Specified as resource type a computeResource with 1 VCPU.

**Result:**

Step 2: Verify successful response.

Step 3: Store the received subscriptionId.

**Action:**

Step 4: Reduce the quota for “Promise” limiting the amount of VCPUs to 3.  
Note: In this release the created quota will be taken as the total existing capacity by Promise (to be confirmed).

**Result:**

Step 5: Verify a Notification is received. Verify it indicates that the total capacity equals 3 and the available capacity equals 3.

**Action:**

Step 6: Create a compute resource.  
Description: use the Allocate Virtualised Compute Resource operation indicating 1 VCPU.

**Result:**

Step 7: Verify successful response.

Step 8: Store the received computeId.

Step 9: Verify a Notification is received. Verify it indicates that the available capacity equals 2 and the used capacity equals 1.

**Action:**

Step 10: Create a compute resource reservation.  
Description: Use the Create Compute Resource Reservation operation to reserve a virtualizationContainer with 1 VCPU. Omit start time and end time.

**Result:**

Step 11: Verify successful response.

Step 12: Store the received reservationId.

Step 13: Verify a Notification is received. Verify it indicates that the available capacity equals 1 and the reserved capacity equals 1.

**Action:**

Step 14: Terminate the compute resource reservation using the received reservationId.

**Result:**

Step 15: Verify successful response.

Step 16: Verify a Notification is received. Verify it indicates that the available capacity equals 2 and the reserved capacity equals 0.

**Action:**

Step 17: terminate the compute resource using the received computeId.

**Result:**

Step 18: Verify successful response.

Step 19: Verify a Notification is received. Verify it indicates that the available capacity equals 3 and the used capacity equals 0.

**Action:**

Step 4: Restore the quota for “Promise” to its original value limiting the amount of VCPUs to 5.  
Note: In this release the created quota will be taken as the total existing capacity by Promise (to be confirmed).

**Result:**

Step 5: Verify a Notification is received. Verify it indicates that the total capacity equals 5 and the available capacity equals 5.

# Test Case Last : Clean-up

**Action:**

Step 1: Remove CPU quota.

Step 2: Remove storage quota.

Step 3: Remove OpenStack project “Promise”.

# 