ACB

Failover Tests for Automation Controller

Prepared by Ashu Rana
On behalf of Rackspace

Table of Contents

1 Failover Test for Automation Controller Node

Objective: Simulate a failover scenario for an Automation Controller node to ensure that tasks execute seamlessly in a single, non-live environment.

Prerequisites:

1. A single environment configured with Ansible Automation Platform components.
2. Documentation of the failover plan specific to the Automation Controller.
3. Team members familiar with the failover plan.

Test Steps:

1. **Schedule the Test:**
   - Choose a suitable time for the test, ensuring that it won't interfere with other configuration tasks in your non-live environment.
   - Notify relevant team members and stakeholders about the testing schedule.
2. **Preparation:**
   - Ensure that your single environment is configured with the Automation Controller and Execution Nodes.
   - Confirm that both Automation Controller nodes are operational and healthy in this environment.
3. **Identify the Node to Stop:**
   - Decide which of the two Automation Controller nodes you will simulate the failure for. Make sure this choice aligns with your failover plan.
4. **Monitor the Environment:**
   - Begin monitoring the system, including the status of both Automation Controller nodes, tasks running, and any relevant metrics.
5. **Execution Node Configuration:**
   - Verify that the Execution Nodes are correctly configured to work with both Automation Controller nodes.
6. **Execute Tasks:**
   - Trigger the execution of tasks that are typically run in your production environment.
   - Verify that tasks are assigned and executed as expected.
7. **Simulate Node Failure:**
   - Simulate the failure of the selected Automation Controller node. You can do this by stopping the relevant service or VM in your non-live environment.
   - Observe the impact on task execution.
8. **Monitor Failover Simulation:**
   - Simulate the failover process by manually reassigning tasks or responsibilities from the failed node to the functioning node.

- Continuously monitor this simulation to ensure it progresses smoothly and without errors.

9. **Verify Task Continuity:**
   - Check if tasks continue to execute seamlessly without any noticeable delays or issues during the failover simulation.
   - Pay attention to any potential task failures or errors.
10. **Review Metrics:**
    - Analyze system metrics, logs, and any performance indicators to ensure that the failover simulation did not negatively impact the environment.
11. **Document Results:**
    - Document the results of the failover simulation, including any issues encountered, task completion times, and any observations during the test.
12. **Debrief and Analysis:**
    - Hold a debriefing session with the team to discuss the test results and gather feedback.
    - Analyze the test outcomes to identify areas for improvement.
13. **Update the Failover Plan:**
    - Based on the results and feedback from the failover simulation, update your Automation Controller failover plan to address any weaknesses or areas for improvement.
14. **Repeat Regularly:**
    - Schedule regular failover simulations for the Automation Controller in your non-live environment to ensure continued reliability.
    - Test different failure scenarios to cover a range of potential issues.
15. **Documentation and Reporting:**
    - Document the details of the failover simulation and its outcomes in a formal report.
    - Share the report with relevant stakeholders and keep it for reference in future tests and audits.

While simulating a failover in a non-live environment is not the same as testing in a production or staging environment, it still allows you to evaluate the failover plan and make necessary improvements before going live.

2 Load Balancer Failover Test

Objective: To test the failover capability of the Octavia load balancer that balances traffic between two Automation Controller nodes. Ensure that tasks continue to execute smoothly during the failover process.

Prerequisites:

1. Octavia load balancer configured to distribute traffic to Automation Controller nodes.
2. Documentation of the failover plan specific to the load balancer configuration.
3. Team members are familiar with the failover plan.

Test Steps:

1. **Schedule the Test:**
   - Choose a suitable time for the test, ideally during a maintenance window.
   - Notify relevant team members and stakeholders about the testing schedule.
2. **Preparation:**
   - Ensure that your Automation Controller nodes are operational and healthy.
   - Confirm that the Octavia load balancer is correctly configured and distributing traffic between the two Automation Controller nodes.
3. **Monitor the Environment:**
   - Begin monitoring the system, including the status of both Automation Controller nodes, tasks running, and any relevant metrics.
4. **Execution Node Configuration:**
   - Verify that the Execution Nodes are correctly configured to work with both Automation Controller nodes.
5. **Execute Tasks:**
   - Trigger the execution of tasks that are typically run in your production environment.
   - Confirm that tasks are assigned and executed as expected through the load balancer.
6. **Failover Simulation:**
   - Simulate the failover scenario by intentionally failing over the Octavia load balancer from one Automation Controller node to the other.
   - This may involve changing DNS records or modifying the load balancer configuration to shift traffic to the secondary node.
7. **Monitor the Failover:**
   - Continuously monitor the failover process to ensure that it progresses smoothly and without errors.
   - Verify that the load balancer redirects traffic to the healthy Automation Controller node.
8. **Verify Task Continuity:**
   - Check if tasks continue to execute seamlessly without any noticeable delays or issues during the failover.

- Pay attention to any potential task failures or errors.

9. **Review Metrics:**
   - Analyze system metrics, logs, and any performance indicators to ensure that the failover process did not negatively impact the environment.
10. **Document Results:**
    - Document the results of the load balancer failover test, including any issues encountered, task completion times, and any observations during the test.
11. **Rollback (if necessary):**
    - If the failover did not work as expected, follow the documented rollback procedure to restore the original load balancer configuration.
12. **Debrief and Analysis:**
    - Hold a debriefing session with the team to discuss the test results and gather feedback.
    - Analyze the test outcomes to identify areas for improvement in the load balancer failover process.
13. **Update the Failover Plan:**
    - Based on the results and feedback from the test, update your load balancer failover plan to address any weaknesses or areas for improvement.
14. **Repeat Regularly:**
    - Schedule regular load balancer failover tests to ensure continued reliability.
    - Test different failure scenarios to cover a range of potential issues.
15. **Documentation and Reporting:**
    - Document the details of the load balancer failover test and its outcomes in a formal report.
    - Share the report with relevant stakeholders and keep it for reference in future tests and audits.

By performing this Load Balancer Failover Test, you can validate that your Octavia load balancer can seamlessly redirect traffic to a healthy Automation Controller node in the event of a node failure, ensuring uninterrupted service for your Ansible Automation Platform.

3 Resource Exhaustion Test

Objective: To test the failover capability of Automation Controller nodes when one node experiences resource exhaustion (e.g., CPU, memory, or disk space saturation). Verify that tasks automatically failover to the other node.

Prerequisites:

1. Two Automation Controller nodes in your cluster.
2. Documentation of the failover plan specific to resource exhaustion scenarios.
3. Team members familiar with the failover plan.

Test Steps:

1. **Schedule the Test:**
    - Choose a suitable time for the test, ideally during a maintenance window.
    - Notify relevant team members and stakeholders about the testing schedule.
2. **Preparation:**
    - Ensure that both Automation Controller nodes are operational and healthy before the test begins.
3. **Monitor the Environment:**
    - Start monitoring the resource utilization of the Automation Controller nodes, including CPU, memory, and disk space.
4. **Execution Node Configuration:**
    - Verify that the Execution Nodes are correctly configured to work with both Automation Controller nodes.
5. **Execute Tasks:**
    - Trigger the execution of a set of tasks that are typically run in your production environment.
    - Confirm that tasks are assigned and executed as expected by both Automation Controller nodes.
6. **Resource Exhaustion Simulation:**
    - Gradually increase the resource utilization on one of the Automation Controller nodes until it reaches a state of resource exhaustion.
    - You can achieve this by running resource-intensive tasks, filling up disk space, or consuming excessive memory or CPU.
7. **Monitor Resource Utilization:**
    - Continuously monitor the resource utilization of the overloaded Automation Controller node as it approaches resource exhaustion.
8. **Failover Observation:**
    - Observe the behavior of the Automation Controller nodes during resource exhaustion. The node under stress should experience issues or become unresponsive.
9. **Automated Failover Verification:**
    - Confirm that the failover mechanism automatically detects the resource exhaustion on the overloaded node and begins redirecting tasks to the healthy node.
10. **Verify Task Continuity:**
    - Check if tasks continue to execute seamlessly without any noticeable delays or issues on the healthy Automation Controller node.
    - Pay attention to any potential task failures or errors.
11. **Review Metrics:**
    - Analyze system metrics, logs, and any performance indicators to ensure that the failover process did not negatively impact the environment.
12. **Document Results:**

- Document the results of the resource exhaustion test, including resource utilization data, task completion times, and any observations during the test.

13. **Resource Cleanup:**
    - After completing the test, perform resource cleanup to restore the overloaded Automation Controller node to its normal state.
14. **Debrief and Analysis:**
    - Hold a debriefing session with the team to discuss the test results and gather feedback.
    - Analyze the test outcomes to identify areas for improvement in resource management and failover procedures.
15. **Update the Failover Plan:**
    - Based on the results and feedback from the test, update your failover plan to address any weaknesses or areas for improvement related to resource exhaustion scenarios.
16. **Repeat Regularly:**
    - Schedule regular resource exhaustion tests for your Automation Controller nodes to ensure continued reliability.
    - Adjust the resource exhaustion scenarios to cover different resource types and levels of stress.
17. **Documentation and Reporting:**
    - Document the details of the resource exhaustion test and its outcomes in a formal report.
    - Share the report with relevant stakeholders and keep it for reference in future tests and audits.

Performing the Resource Exhaustion Test helps ensure that your Automation Controller nodes can handle resource saturation scenarios effectively, automatically failing over to a healthy node to maintain service continuity in production environments.

4 Recovery Teat After Failover

Objective: To simulate the recovery process after a failover by bringing the initially failed Automation Controller node back online. Verify that the node can rejoin the cluster, synchronize data, and resume task execution seamlessly.

Prerequisites:

1. A previously conducted failover test.
2. Two Automation Controller nodes in your cluster.
3. Documentation of the failover and recovery plan.

Test Steps:

1. **Schedule the Test:**
    - Schedule the recovery test after the previously conducted failover test.
    - Ensure it aligns with your maintenance window and notify relevant team members.
2. **Preparation:**
    - Confirm that the Automation Controller node, which was initially failed over, is still offline.
    - Ensure that the remaining Automation Controller node is operational and healthy.
3. **Verify Failover Success:**
    - Review the results of the failover test to confirm that tasks are executing smoothly on the healthy Automation Controller node.
4. **Recovery Simulation:**
    - Simulate the recovery process by bringing the initially failed Automation Controller node back online. This can be done by starting the relevant service or VM.
5. **Monitor Recovery:**
    - Continuously monitor the recovery process to ensure that the failed node can successfully rejoin the cluster and synchronize data.
    - Watch for any errors or issues that may occur during recovery.
6. **Data Synchronization:**
    - Confirm that the recovered node synchronizes data, configurations, and any changes made during the failover period from the healthy node.
7. **Cluster Reintegration:**
    - Verify that the recovered node seamlessly reintegrates with the cluster and participates in task assignment and execution.
8. **Task Execution:**
    - Trigger the execution of tasks after recovery to ensure that both Automation Controller nodes are capable of executing tasks without conflicts.
9. **Load Balancer Update (if applicable):**
    - If you use a load balancer, ensure that it is updated to distribute traffic to both Automation Controller nodes after the recovery.
10. **Verify Task Continuity:**
    - Check if tasks continue to execute seamlessly without any noticeable delays or issues on both Automation Controller nodes.
11. **Review Metrics:**
    - Analyze system metrics, logs, and any performance indicators to ensure that the recovery process did not negatively impact the environment.
12. **Document Results:**
    - Document the results of the recovery test, including the success of data synchronization, cluster reintegration, and task execution.
13. **Debrief and Analysis:**

- Hold a debriefing session with the team to discuss the test results and gather feedback.
- Analyze the test outcomes to identify areas for improvement in the recovery process.

14. **Update the Failover Plan:**
    - Based on the results and feedback from the recovery test, update your failover and recovery plans to address any weaknesses or areas for improvement.
15. **Repeat Regularly:**
    - Schedule regular recovery tests to ensure that your Automation Controller nodes can seamlessly recover and reintegrate into the cluster after a failover.
16. **Documentation and Reporting:**
    - Document the details of the recovery test and its outcomes in a formal report.
    - Share the report with relevant stakeholders and keep it for reference in future tests and audits.

Conducting a Recovery Test is crucial to ensure that the failover process is not only successful but also that the recovered node can seamlessly rejoin the cluster and resume normal operations, maintaining high availability and reliability.