What are some recommended best practices for performing Auto Scaling group rolling updates?

Issue

AWS CloudFormation allows you to control the behavior of updates to an Auto Scaling group resource by using the **UpdatePolicy** attribute. Performing a rolling update on an Auto Scaling group can result in unexpected behavior if you do not have the right settings configured.

Short Description

The AWS::AutoScaling::AutoScalingGroup resource supports an UpdatePolicy attribute. This is used to define how an Auto Scaling group resource is updated when an update to the CloudFormation stack occurs. A common approach to updating an Auto Scaling group is to perform a rolling update, which is done by specifying the AutoScalingRollingUpdate policy. This retains the same Auto Scaling group and replaces old instances with new ones, according to the parameters specified.
Note: This article focuses on a rolling update and not the AutoScalingReplacingUpdate policy.

Resolution

The AutoScalingRollingUpdate policy supports a number of configuration options; here is a sample template:

```
"UpdatePolicy": {
    "AutoScalingRollingUpdate": {
        "MaxBatchSize": Integer,
        "MinInstancesInService": Integer,
        "MinSuccessfulInstancesPercent": Integer
        "PauseTime": String,
        "SuspendProcesses": [ List of processes ],
        "WaitOnResourceSignals": Boolean
     }
}
```

SuspendProcesses must

During a rolling update, you must suspend Auto Scaling processes to avoid making unexpected changes to the group. If an unexpected scaling action changes the state of the group during a rolling update, this can cause CloudFormation to have an inconsistent view of the group and can cause the rolling update to fail. Processes to suspend include: HealthCheck, ReplaceUnhealthy, AZRebalance, AlarmNotification, and ScheduledActions.
Note: Do not suspend the following processes because they are required for the rolling update operation: Launch, Terminate, and AddToLoadBalancer (if the Auto Scaling group is being used with Elastic Load Balancing).

MinSuccessfulInstancesPercent 

If you are replacing a large number of instances during a rolling update and you are also waiting for a success signal from each instance, you might need to specify a MinSuccessfulInstancePercent value. This is useful to prevent CloudFormation from rolling back the entire stack if a single instance fails to launch. For example, if you perform a rolling update to change your AMI to replace 10 instances, and the first 9 are successful but the last one fails due to a transient issue, CloudFormation would not be able to update the entire stack and would attempt to roll back. If you set MinSuccessfulInstancePercent to 50%, CloudFormation would wait for 5 instances to signal success and then mark the overall AutoScaling Group resource as UPDATE_COMPLETE, because the new AMI is working correctly and there are enough instances in service to maintain traffic. If one of the subsequent instances fails to launch successfully, it would eventually be replaced by your Auto Scaling group health checks.

WaitOnResourceSignals + PauseTime 

Enabling WaitOnResourceSignals allows CloudFormation to wait until you have received a success signal before performing the next scaling action.

Note: It is important to note that when WaitOnResourceSignals is set to true, PauseTime becomes a timeout value. CloudFormation waits until the maximum time specified in the PauseTime value for a success signal. If a signal is not received, it cancels the update and attempts to roll back the stack. In order to roll back, CloudFormation performs a rolling update with the same settings, so the same PauseTime value is used upon rollback. It is recommended that you specify a long enough PauseTime for your instances to bootstrap and signal a success.