Scenario 1: Testing knowledge on EFS and EC2 and File system configuration

We have a requirement to create n no of instances with common shared location.

Shared location should be attached automatically even after restart of EC2. Please provide steps to how you can achieve this.

Solution

- Create a EFS volume with Security Group
- Create a Security Group for EC2 and allow outbound to EFS security Group
- Under EFS security group add inbound to EC2 Security Group
- Create an EC2 instances and under Configure Instance Details step Under File system add the EFS created in the Step1 and under Advanced Details add User Data script to mount the EFS

Scenario 2: Security and ACL configuration

We are getting huge traffic from an IP address (may be some kind of attack) and we want to restrict any traffic from a particular IP address to our application.

How we can achieve this.

Solution

- Create a New Network ACL rule on the VPC where the Application is hosted with high priority denying the access to the IP which is been suspected for the attack

Scenario 3: Testing understanding of LB and configuration of TG

Why we need to use ALB instead of NLB for any web application. Even though NLB will give better performance.

Solution

- ALB works on Network Layer7 so can intercept Http Header and route the traffic based on URL path
- ALB can also perform SSL hash for all the request, offloading it from Application and reducing the Resource consumption

Scenario 4: Testing subnet configuration

Currently we have all EC2 instances are available on public subnet. We understood from security team we should move all instances to private subnet. How you can do it in best way.

Solution

Convert the public subnet to private subnet

- Create a new route table
- Go the Subnet which has to be made private and remove the old Route table and attach the new route table
- Go to the new route table , under Routes edit route and add a new route with following details
  - Destination "0.0.0.0/0"
  - Target "NAT gateway available "
- Remove any Public IP attached to the EC2 instances

Scenario 5: (only for AWS cert guys): Testing Firewall rules knowledge

We have an application running with 2 different context paths.

1) /apps/admin
2) /apps/user/login

We have a requirement to block /apps/admin contact path to be accessed over network. How we can achieve this one.

Solution

- In Application Load balancer go to listener Select Listener and View/Edit rules
- Create new rule with following details
  - Rule ID: 1
  - If Condition: path value "/apps/admin"
  - Then: Retune fixed responses

Scenario 6: Testing design knowledge

We have a client who want to use AWS to host their application.

If you are asked to provide a best architecture diagram what kind of tools and practices you will follow to host a 3 tier architecture.

Solution 1

- Tools
  1) VPC
  2) Private and Public Subnets
  3) Security Groups
  4) Routes
  5) EC2 Instances
  6) RDS Instances
  7) WAF

8) Application Load Balancer
9) Classic Load Balancer
10) NAT Gateway
11) Internet Gateway
12) Backup / Snapshots
13) S3 Bucket
14) Cloud Watch
15) Cloud Trail
16) SNS Service
17) Route 53
18) AWS Certificate Manager (ACM)

Overview:

Network Layout: One VPC with One Public and two Private Subnet will be created in AZ1 and AZ2, Each Subnet will be assigned with a Security Group to Control the Network traffic.

Tier1: The Frontend application will be hosted in 2 EC2 instances each placed in different AZ for High availability, Traffic to the Frontend application will be load balanced by Application load balancer which will also provide SSL hash function and WAF will be enabled to protect the Application

Tier2: The middleware application will be hosted in 2 EC2 instances each placed in different AZ for High availability, Traffic to the Frontend application will be load balanced by Network load balancer

Tier3: Amazon RDS Multi-AZ instances will be used has Data tier to store the Application/user information and can only be accessed by middleware application

Additional components like cloud watch, SNS, S3 will be used to monitor the infrastructure

Solution 2

- Tools
  1. VPC
  2. Private and Public Subnets
  3. Security Groups
  4. Routes
  5. Elastic Kubernetes Service
  6. EC2 Instances
  7. RDS Instances
  8. WAF
  9. Application Load Balancer
  10. S3 Bucket
  11. Cloud Watch
  12. Cloud Trail
  13. SNS Service
  14. Route 53

15. AWS Certificate Manager (ACM)

Tier1 & 2: The both Frontend and middleware application will be deployed in Elastic Kubernetes Service will be created in a private subnet and Application load balancer will be used as ingress controller to route the traffic to Frontend

Tier3: Amazon RDS Multi-AZ instances will be used has Data tier to store the Application/user information

Scenario 7: Testing Kubernetes knowledge

In your K8S cluster one of node is not behaving properly so you want to remove it from cluster to debug the Infra issues. How you can do this best way.

Solution
Run the following command to Drain (move pods) from the node which has issue, this will tell K8s Scheduler to remove the pods from the node and place it in other node

```
kubectl drain <node_name> --ignore-daemonsets
```

When the issue is resolved we can run the below command to Join back the node

```
kubectl uncordon <node_name>
```

Scenario 8: Testing Kubernetes knowledge

In your K8S cluster you observed peak load so you want to add new node to the cluster. How you can move pods from current cluster nodes to new node.

Solution:
Add a taint with (effect: "NoSchedule") to the peak load node delete the Pod which are running in the peak load node one by one. This will ensure no down time for the application. And due to Taint on the node pods will not be schedule on the peak load node and will be move to new node Once the load is normal untint the node

Scenario 9: Testing Kubernetes knowledge

You are using Ingress Controller and you observed some of the services are not proxying properly in Ingress controller what kind of things you will do debug this issue.

Solution:

1. Check if right service and port is configured in Ingress rule.
2. Check if Ingress rule deployed in the right namespace where application is deployed
3. Check when using SSL, is the Certificate as secret is created and configured in the Ingress rule
4. Check the logs of ingress pod to see the request reaching the ingress controller
5. Check if the Network policy is blocking the communication from ingress controller pod to pod

Scenario 10: Testing LB knowledge

You have 4-5 pods/nodes configured under your LB. Currently out of 100 requests 80 requests are success and 20 are failing. How you can debug what pods/node can be culprit. What can be the configuration/solution you will suggest to avoid this kind of issues.

Solution:

1. Check pod health
2. Check application/pod logs if there are any errors are exceptions

To avoid this issue

We can configure HTTP liveness probe, K8s will restart the pod if the liveness probe fails, and this will ensure that the pod which is not working will be remove and replaced with new one.