High Availability and Scalability: ELB and ASG

Uday Manchanda

August 8, 2021

1 High Availability and Scalability

- Scalability: application/system can handle greater loads by adapting
- Two types:
  - Vertical
  - Horizontal (aka elasticity)
- Linked to, but from different from, High Availability

1.1 Vertical Scalability

- increasing the size of the instances
- EX: Application runs on t2.micro, scaling vertically means running on t2.large
- Very common for non distributed systems like DBs
- hardware limits

1.2 Horizontal Scalability

- increasing the number of instances/systems
- Implies distributed systems
- Cloud makes it easier to scale horizontally
- Scale out/in
- ASGs/LBs

1.3 High Availability

- Goes hand in hand with horizontal scaling
- Running your application in multiple AZs

2 Elastic Load Balancing

- Load Balancers: servers that forward internet traffic to multiple servers downstream
- Users interface with the load balancer which serve traffic to the EC2 instances
- Why use it?
  - Spread load
  - Expose single point of access (DNS) to application
  - Seamlessly handle failures
  - Health checks, SSL termination
  - Separate public from private traffic
- ELB: managed load balancer from AWS
  - Classic Load Balancer v1 (HTTP/HTTPS/TCP)
  - Application Load Balancer v2 (HTTP/HTTPS/WebSocket)
  - Network Load Balancer v2 (TCP/TLS/UDP)
- Can scale but not instantaneously
- ELB access logs will access requests
- CW metrics gives aggregate metrics
- NLBs exposed a public static IP whereas an ALB/CLB exposes a static DNS

2.1 Health Checks

- enable the LB to know if instances it forwards traffic to are available to reply to requests
- Done on a port and route (usually $/health$). If not 200, traffic is unhealthy

2.2 Security Groups

- users $\rightarrow$ (HTTP/HTTPS) LB $\rightarrow$ (HTTP restricted to LB) EC2

2.3 Sticky Sessions

- aka session afinity
- it is possible to implement stickiness so that the same client is always redirected to the same instance behind a load balancer
- Works for CLB and ALBs
- The "cookie" is used for stickiness has an expiration date
- Use case: user doesn't lose his session data
- Cookie Names
  - Application based cookies
    - custom cookie - generated by the target, can include any custom attributes, must be specified for each TG
    - Application cookie - generated by the LB, named AWSALBAPP
  - Duration based cookies: generated by LB, always either AWSALB or AWSELB

2.4 Cross Zone Load Balancing

- Assume you have two load balancers in two different AZs with multiple EC2 instances
- Client is accessing these LBs
- with cross zone load balancing each LB instance distributes evenly across all registered instances in all AZs
- So the client traffic gets routed evenly to each AZ
- If AZ1 has 2 EC2 instances and AZ2 has 8 EC2 instances, each EC2 instance gets 10 percent of the traffic (total of 10 instances)
- without cross zone load balancing, the traffic is still routed evenly between each AZ
- But then each EC2 in AZ1 gets 25 percent of the traffic and AZ2 gets 6.25 percent
- Always on for ALBs but disabled by default in NLBs

2.5 SSL certificates

- SSL cert allows traffic between your clients and your LB to be encrypted in transit (in flight encryption)
- SSL = secure sockets layer
- TLS = transport layer security, newer version
- They have an expiration date
- HTTPS = HTTP + SSL
- LB does something called SSL certificate termination which allows it to talk to EC2 instances via HTTP
- ACM = AWS certificate manager
- Clients can use SNI (server name indication) to specify the hostname they reach
- SNI
  - solves the problem of loading multiple SSL certificates onto one web server to serve multiple websites
  - its a newer protocol and requires the client to indicate the hostname of the target server in the initial handshake
  - Server will then find the correct cert or return default one
  - Only works for ALB/NLB and CloudFront
  - EX: ALB with multiple TGs which point to different hostnames, load the SSL cert for each hostname, using SNI, the LB can point the right SSL cert to the right TG

2.6 Connection Draining

- Time to complete in-flight requests while the instance is de-registering or unhealthy
- Stops sending new requests to the instance which is de-registering
- default is 300 seconds

3 Classic Load Balancer

- TCP, HTTP, and HTTPS
- v1
- Fixed hostname
- client $\rightarrow$ CLB $\rightarrow$ internal EC2

4 Application Load Balancer

- v2, HTTP
- Load balancing to multiple HTTP applications across machines (target groups)
- Load balancing to multiple applicants on the same machine (ex: containers)
- Routing tables to different target groups
  - Routing based on path in URL
  - Routing based on hostname in URL
  - Routing based on query string, headers
- great fit for micro-services and container based applications
- Port mapping feature to redirect to a dynamic port in ECS
- EX: ALB hosts multiple microservices which are routed via target groups
- Fixed hostname
- Application servers don't see the IP of the client directly
- The true IP of the client is inserted in the header **X-Forwarded-For**

4.1 Target Groups

- EC2 instances (can be managed by ASG) - HTTP
- ECS tasks (managed by ECS itself) - HTTP
- Lambda functions - HTTP request translated into a JSON event
- IP addresses - private IPs
- ALB can route to multiple target groups

5 Network Load Balancer

- v2, TCP/UDP
- Less latency, handles millions of requests
- Has one static IP per AZ and supports assigning Elastic IP (helpful for whitelisting specific IP)
- NLB are used for extreme performance, TCP or UDP traffic
- not in free tier

6 Auto Scaling Groups

- in real life, the load on your websites and application can change quickly
- the goal is to: scale out (add EC2 instances) to match increased load or scale in by removing EC2 instances
- Ensure we have a min and max number of machines running
- automatically register new EC2 instances to an LB
- ASGs have the following attributes
  - AMI + instance type
  - EC2 user data
  - EBS volumes
  - SGs
  - SSH key pair
  - Min/Max Size, initial capacity
  - Network and Subnets information
  - LB information
  - Scaling policies
- Auto scaling alarms: Can scale ASG based on CW alarms
- Auto scaling new rules: "define" better AS rules that are directly managed by the EC2 (ex: CPU usage, avg network in/out, number of requests)
- Can create with Launch Template

6.1 ASG Scaling Policies

- Dynamic Scaling Policies
  - Target Tracking Scaling
    - Easiest to set up
    - EX: i want the avg ASG CPU to stay around 40 percent
  - Simple/Step Scaling
    - EX: When a CW alarm is triggered (CPU $>$ 70 percent)then add 2 units
    - EX: when a CW alarm is triggered (CPU $<$ 30 percent) then remove 1 unit
  - Scheduled Actions
    - Anticipate a scaling based on known usage patterns

        * EX: increase min capacity to 10 at 5 pm friday

- Predictive Scaling
  - continuously forecast load and schedule scaling ahead
- Good metrics to scale on: CPU Utilization, RequestCountPerTarget, Avg Network in/out
- Cooldown: there is a cooldown period after a scaling happens. ASG will not launch or terminate additional instances
- Use a ready to use AMI to reduce config time in order to be serving requests faster

6.2 ASG for solution architects

- ASG default termination policy
  1. Find the AZ which has the most number of instances
  2. If there are multiple instances in the AZ to choose from, delete the one with the oldest launch configuration
- ASG tries to balance the number of instances across the AZs by default
- Lifecyle Hooks
  - By default as soon as an instance is launched it's in service
  - You have the ability to perform extra steps before the instances goes into service (pending state)
  - Also ability to perform actions before instance is terminated (terminating state)
- Launch Configuration vs Launch Template
  - Both: AMI, instance type, SGs, tags
  - Configuration: legacy, must be re-created every time
  - Template: newer, can have multiple versions, create parameter subsets, provision using both on-demand and spot instances (recommended by AWS)
- Unhealthy EC2 instances will get terminated