Elastic Beanstalk

If you are studying for AWS Developer Associate Exam, this guide will help you with quick revision before the exam. it can use as study notes for your preparation.

Dashboard Other Certification Notes

Elastic Beanstalk

***Elastic Beanstalk* is a developer centric view of deploying application on AWS.**

- A managed service
  - Instance configuration
  - OS is handled by Beanstalk
  - Deployment strategy is configurable but performed by Beanstalk
  - Application code configurable
- It will leverage all the AWS components that we have gone over thus far:
  - EC2
  - ASG
  - ELB
  - RDS
  - Etc..
- Elastic Beanstalk is free but you pay for the underlying instances
- Three architecture models:
  - Single instance deployment: good for developers
  - LB + ASG: great for production or staging web applications
  - ASG only: great for non-web apps in production
- Elastic Beanstalk has three different components:
  - Application
  - Application Version (Each deployment gets assigned a version)
  - Environment name (dev, staging, prod): free naming
- You deploy application versions to environments and can promote application versions to the next environment
- Rollback feature to previous application versions
- Full control over the lifecycle of environments
- Support for many platforms:
  - Go
  - Java
  - Python
  - Node.js
  - Ruby
  - Single Container Docker
  - Multi Container Docker
  - Pre-configure Docker
  - Write your own custom platforms (If the any of the above is not supported)

Elastic Beanstalk Deployment modes

- **Single Instance deployment**: good for development, we get 1 ec2 instance in 1 ASG and 1 Elastic IP. DNS names maps straight ot the Elastic IP
- **High Availability with Load Balancer**: great for production, we have an ASG across multiple AZs
- Deployment updates:
  - **All at once**: we deploy all our applications in on go. Fastest, but instances have a downtime. No additional cost is applied while deploying
  - **Rolling**: update e few instances at a time, move onto next set of instances (bucket) if the first set was healthy. The application will run bellow capacity for a given period (bucket size). At some point the application will run both versions. No additional cost is encountered during deployment
  - **Rolling with additional batches**: similar deployment procedure as the **rolling** approach with the difference of having an additional batch started at the beginning of deployment with the newer version of the application. Afterwards, the older instanced are gradually moved to newer version. While deploying, both application version will be running at the same time. Minor additional cost may be encountered since the additional batch will be

same time. Minor additional cost may be encountered since the additional batch will be present until the deployment is finished.
    - **Immutable**: new version of the application is deployed to an entirely new ASG. If the new version passes the initial validation, the old ASG is terminated. Deployment will cause 0 downtime. The additional cost encountered while deploying is the highest compared to other deployment types.

Blue / Green Deployment

- This is not a direct feature of Elastic Beanstalk
- Zero downtime and release facility
- Create a new staging environment and deploy your newest version there
- The new environment (green) can be validated independently and roll back if there's issues
- Route 53 can be setup using weighted policies to redirect a little bit of traffic to the staging environment
- Using the elastic beanstalk console, you can "swap URLs" when with the testing environment
- This is a manual feature, it's not directly embedded in EB

Elastic Beanstalk CLI

- We can install an additional CLI called the "EB cli" which makes working with Beanstalk from the CLI easier
- Basic commands are:
    - eb create
    - eb status
    - eb health
    - eb events
    - eb logs
    - eb open
    - eb deploy
    - eb config
    - eb terminate
- It's helpful for your automated deployment pipelines!

Elastic Beanstalk Lifecycle Policy

- Elastic Beanstalk can store at most 1000 application versions
- When this limit is reached, we wont be able to deploy a new version
- In order to be able to deploy again, we have to remove older versions
- To phase out old versions, we can use a lifecycle policy
- This policy can be based on:
    - Time (remove versions which are older than...)
    - Space (remove older versions if we have more versions than...)
- Currently used versions are not deleted
- There is an option to not delete source bundles from S3, only from beanstalk interface

Elastic Beanstalk Extensions

- A zip file containing our code must be deployed to Elastic Beanstalk
- All the parameters set in the UI can be configured with code using files
- Requirements:
    - in the .ebextensions/ directory in the root of source code
    - YAML / JSON format
    - .config extensions (example: logging.config)
    - Able to modify some default settings using: option_settings
    - Ability to add resources such as RDS, ElastiCache, DynamoDB, etc...
- Resources managed by .ebextensions get deleted if the environment goes away
- The .ebextensions folder goes to the root of your project

Elastic Beanstalk Under the Hood

- Elastic Beanstalk uses CloudFormation under the hood
- We can take advantage of this by provisioning other resources from beanstalk
- We can place config files in the .ebextensions to provision basically anything we want

Elastic Beanstalk Cloning and Migrations

- We can clone an environment with the exact same configurations
- Can be useful for deploying "test" versions for our applications
- All resources and configurations are preserved after cloning
- Settings can be changed for the new EB stack after cloning
- Migrate Load Balancer:
    - The LB type can not be changed after the EB environment was created
    - In order to be able to change the type of an LB, we need to do a migration:
        1. Create a new environment with the same configurations as the original except LB => **this means we can not do cloning !!!**

this means we can not do cloning !!!
    2. Deploy our application into the new environment
    3. Shift the traffic from the old environment to the new one (this can be done with a CNAME swap or DNS update)
- Decouple RDS from the EB stack:
  - RDS database can be added to the EB stack, although it is not recommended for production, because the DB lifecycle is tied to the EB lifecycle
  - Steps to decouple RDS:
    1. Create a snapshot form the DB for safety
    2. Protect the RDS DB from deletion
    3. Create a new EB environment without RDS and point the application to the existing RDS instance
    4. Perform a CNAME swap
    5. Terminate the old EB stack (RDS wont be deleted because of the protection)
    6. Delete CloudFormation stack (it will be in DELETE_FAILED state)

Elastic Beanstalk and Docker

- We can run docker container if we provide:
  - Dockerfile: EB will build an run the Docker container
  - Dockerrun.aws.json (v1): describe where the Docker image should be downloaded from (DockerHub, ECR, etc.)
- EB in single container mode wont use ECS!
- Multi Docker Container:
  - Will run multiple containers per EC2 instance
  - Will create:
    - ECS Cluster
    - EC2 instances in an ECS cluster
    - Load Balancer (high availability mode)
    - Task definition and execution
  - Requires a config of Dockerrun.aws.json (V2) in the root of the source code
    - Dockerrun.aws.json is used to generate the ECS task definition
    - Docker images should be pre-build and stored in ECR, DockerHub, etc.

Elastic Beanstalk and HTTPS

- SSL certificate can be loaded from the console (EB console, load balancer configuration) or from the config .ebextensions/securelistener-alb.config
- SSL certificates can be provisioned using ACM (AWS Certificate Manager) or CLI
- Must configure SG with allowing port 443
- Redirect HTTP to HTTPS:
  - Instances can be configured to redirect traffic
  - Or ALB can be configured with a rule as well
  - Health checks should ne be redirected from the ALB