

Elastic Beanstalk

Elastic Beanstalk – overview

AWS Elastic Beanstalk is a fully managed service provided by Amazon Web Services (AWS) that simplifies the deployment, scaling, and management of web applications and services. It abstracts the underlying infrastructure complexity and allows developers to focus on writing code rather than managing servers and infrastructure configurations.

Key Features of AWS Elastic Beanstalk:

1. **Easy Application Deployment:** Allows developers to deploy web applications and services (web servers, APIs, microservices) written in various programming languages, such as Java, .NET, Node.js, Python, PHP, Ruby, Go, and Docker containers.
2. **Automatic Environment Provisioning:** Manages the underlying infrastructure, including provisioning of resources (like EC2 instances, load balancers, databases), auto-scaling, load balancing, and capacity provisioning based on application requirements.
3. **Application Health Monitoring:** Monitors application health and performance metrics, provides logging and metrics collection via integration with AWS CloudWatch, allowing developers to track and analyze application performance.
4. **Managed Updates and Rollbacks:** Simplifies application updates and versioning by handling deployment updates and allowing easy rollback to previous versions if needed.
5. **Multiple Deployment Options:** Supports various deployment options, including single-instance or multi-instance environments, blue-green deployments, and can integrate with Continuous Integration/Continuous Deployment (CI/CD) pipelines.
6. **Integration with AWS Services:** Easily integrates with other AWS services such as RDS (Relational Database Service), S3 (Simple Storage Service), IAM (Identity and Access Management), and more, providing flexibility and scalability.

Use Cases for AWS Elastic Beanstalk:

1. **Web Application Hosting:** Hosting web applications, websites, APIs, and microservices with ease, without managing underlying infrastructure.
2. **Development and Testing:** Facilitating development and testing by providing an environment for deploying and testing applications quickly.
3. **Scalability and Load Management:** Handling fluctuating traffic loads by automatically scaling resources up or down based on demand.
4. **Simplified Deployment:** Streamlining the deployment process and managing infrastructure configurations, allowing developers to focus on writing code and accelerating time-to-market.
5. **Cost Optimization:** Optimizing costs by automatically managing resources and scaling as needed, helping to avoid over-provisioning of infrastructure.

AWS Elastic Beanstalk is suitable for developers and teams looking for a managed and scalable platform to deploy and manage web applications without the complexity of managing infrastructure configurations. It allows quick deployment, scalability, and ease of management for various types of web applications and services.

Developer problems on AWS

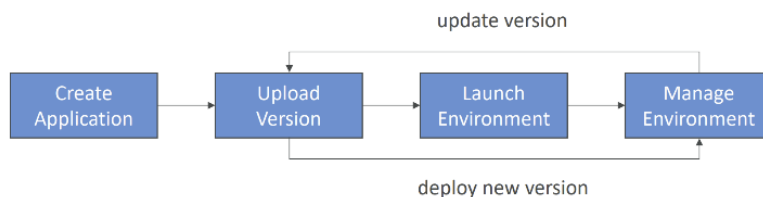
- Managing infrastructure
- Deploying Code
- Configuring all the databases, load balancers, etc
- Scaling concerns
- Most web apps have the same architecture (ALB + ASG)
- All the developers want is for their code to run!
- Possibly, consistently across different applications and environments

Elastic Beanstalk – Overview

- Elastic Beanstalk is a developer centric view of deploying an application on AWS
- It uses all the component's we've seen before: EC2, ASG, ELB, RDS, ...
- Managed service
 - Automatically handles capacity provisioning, load balancing, scaling, application health monitoring, instance configuration, ...
 - Just the application code is the responsibility of the developer
- We still have full control over the configuration
- Beanstalk is free but you pay for the underlying instances

Elastic Beanstalk – Components

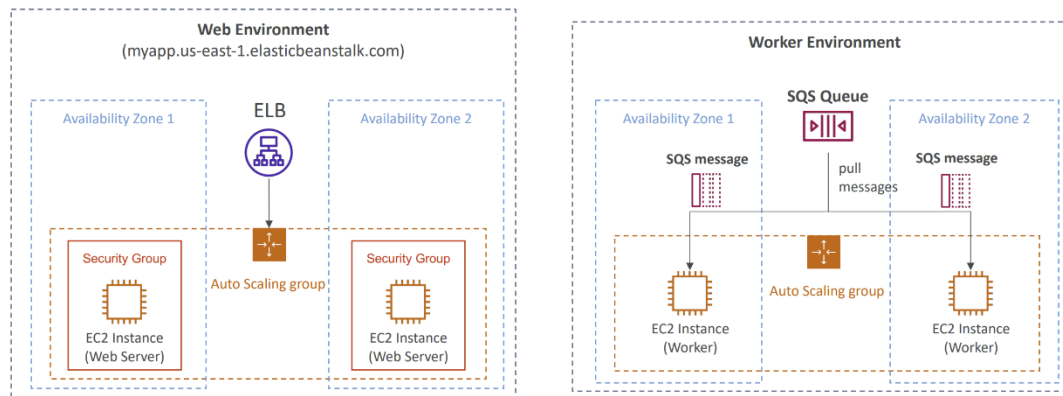
- **Application:** collection of Elastic Beanstalk components (environments, versions, configurations, ...)
- **Application Version:** an iteration of your application code
- **Environment**
 - Collection of AWS resources running an application version (only one application version at a time)
 - **Tiers:** Web Server Environment Tier & Worker Environment Tier
 - You can create multiple environments (dev, test, prod, ...)



Elastic Beanstalk- Supported Platforms

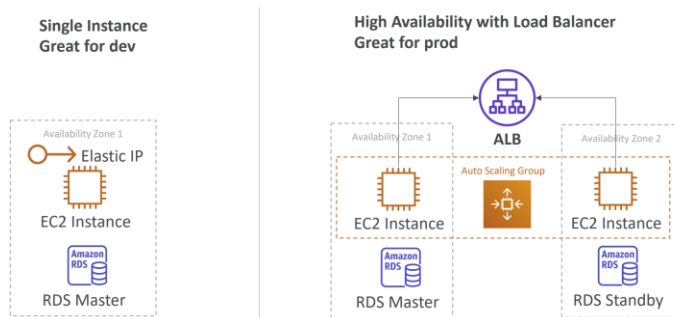
- Go
- Java SE
- Java with Tomcat
- .NET Core on Linux
- .NET on Windows Server
- Node.js
- PHP
- Python
- Ruby
- Packer Builder
- Single Container Docker
- Multi-container Docker
- Preconfigured Docker
- If not supported, you can write your custom platform (advanced)

Web Server Tier vs. Worker Tier

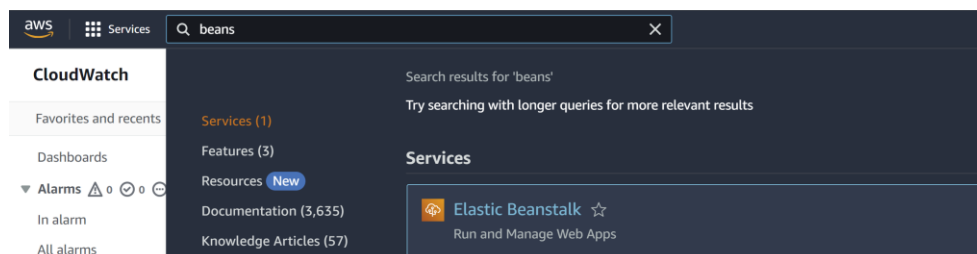


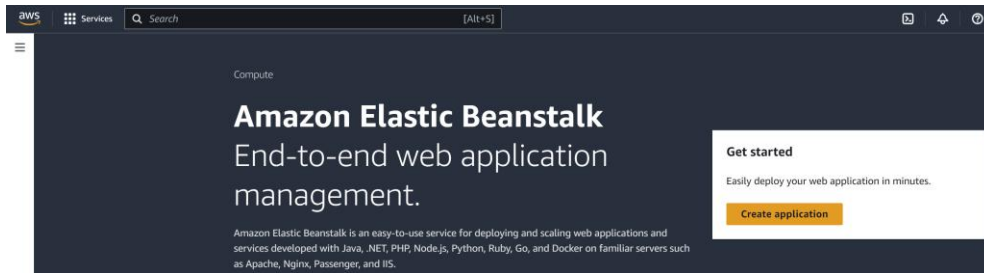
- Scale based on the number of SQS messages
- Can push messages to SQS queue from another Web Server Tier

Elastic Beanstalk Deployment Modes



Lab:





Step 1

Configure environment

Step 2

Configure service access

Step 3 - optional

Set up networking, database, and tags

Step 4 - optional

Configure instance traffic and scaling

Step 5 - optional

Configure updates, monitoring, and logging

Step 6

Review

Configure environment [Info](#)

Environment tier [Info](#)

Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

☒ Web server environment

Run a website, web application, or web API that serves HTTP requests. [Learn more](#)

☐ Worker environment

Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. [Learn more](#)

Application information [Info](#)

Application name

my-beanstalk-1

Maximum length of 100 characters.

► Application tags (optional)

Environment information [Info](#)

Choose the name, subdomain and description for your environment. These cannot be changed later.

Environment name

My-beanstalk-1-env

Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen. This name must be unique within a region in your account.

Domain

Leave blank for autogenerated value

.us-east-1.elasticbeanstalk.com

Check availability

Environment description

Platform [Info](#)

Platform type

☒ Managed platform

Platforms published and maintained by Amazon Elastic Beanstalk. [Learn more](#)

☐ Custom platform

Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Node.js

Platform branch

Node.js 18 running on 64bit Amazon Linux 2023

Platform version

6.0.3 (Recommended)

Keep rest of things same now click next

Application code [Info](#)

☒ Sample application

☐ Existing version

Application versions that you have uploaded.

☐ Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

Presets [Info](#)

Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.

Configuration presets

☒ Single instance (free tier eligible)

☐ Single instance (using spot instance)

☐ High availability

☐ High availability (using spot and on-demand instances)

☐ Custom configuration

Cancel

Next

Now need to create and iam for with the following permissions role which is need in this next page

Q

beanstalk

X

All types

▼

14 matches

< 1 > ⚙

| <div><div></div></div> | Policy name <div></div> | Type | Description |
|--|---|-------------|---|
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AdministratorAccess-AWSElasticBeanstalk</div> | AWS managed | Grants account administrative permissi... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkCustomPlatformforEC2Role</div> | AWS managed | Provide the instance in your custom pl... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkEnhancedHealth</div> | AWS managed | AWS Elastic Beanstalk Service policy f... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkManagedUpdatesCustomerRole...</div> | AWS managed | This policy is for the AWS Elastic Beans... |
| <div><input checked="" type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkMulticontainerDocker</div> | AWS managed | Provide the instances in your multicon... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkReadOnly</div> | AWS managed | Grants read-only permissions. Explicitl... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleCore</div> | AWS managed | AWSElasticBeanstalkRoleCore (Elastic ... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleCWL</div> | AWS managed | (Elastic Beanstalk operations role) Allo... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleECS</div> | AWS managed | (Elastic Beanstalk operations role) Allo... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleRDS</div> | AWS managed | (Elastic Beanstalk operations role) Allo... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleSNS</div> | AWS managed | (Elastic Beanstalk operations role) Allo... |
| <div><input type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkRoleWorkerTier</div> | AWS managed | (Elastic Beanstalk operations role) Allo... |
| <div><input checked="" type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkWebTier</div> | AWS managed | Provide the instances in your web serv... |
| <div><input checked="" type="checkbox"/></div> | <div><div></div> <div></div> AWSElasticBeanstalkWorkerTier</div> | AWS managed | Provide the instances in your worker e... |

Now in screen chose the IAM role and click skip the review.

Step 1

Configure environment

Step 2

Configure service access

Step 3 - optional

Set up networking, database, and tags

Step 4 - optional

Configure instance traffic and scaling

Step 5 - optional

Configure updates, monitoring, and logging

Step 6

Review

Configure service access Info

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

Service role

☒ Create and use new service role
 ☐ Use an existing service role

Service role name

Enter the name for an IAM role that Elastic Beanstalk will create to assume as a service role. Beanstalk will attach the required managed policies to it.

View permission details

EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)

↻

EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

↻

View permission details

This ec2 role we just created now

Cancel

Skip to review

Previous

Next

we are now click skip to review because we dont use the optional things now

In the next page scroll down and submit

Updates

| | | |
|---------------------|-----------------------|----------------------------|
| Managed updates | Deployment batch size | Deployment batch size type |
| Activated | 100 | Percentage |
| Command timeout | Deployment policy | Health threshold |
| 600 | AllAtOnce | OK |
| Ignore health check | Instance replacement | |
| false | false | |

Platform software

| | | |
|----------------|---------------|--------------|
| Lifecycle | Log streaming | Proxy server |
| false | Deactivated | nginx |
| Logs retention | Rotate logs | Update level |
| 7 | Deactivated | minor |
| X-Ray enabled | | |
| Deactivated | | |

Environment properties

Key

Value

No environment properties

There are no environment properties defined

Cancel

Previous

Submit

It will take some time to create the beanstalk and in the backend it will use CloudFormation and create the resources we need. We the stapes by clicking on event.

Elastic Beanstalk

Applications

Environments

Change history

▼ Application: my-beanstalk-1

Application versions

Saved configurations

▼ Environment: My-beanstalk-1-env

Go to environment

Configuration

Events

Health

Logs

Monitoring

Alarms

Managed updates

Tags

▼ Recent environments

My-beanstalk-1-env

Elastic Beanstalk is launching your environment. This will take a few minutes.

Elastic Beanstalk

Environments

My-beanstalk-1-env

My-beanstalk-1-env

Actions

Upload and deploy

Environment overview

Health

Pending

Environment ID

e-wfkmmt3dq

Domain

My-beanstalk-1-env.eba-vymmkmw.us-east-1.elasticbeanstalk.com

Application name

my-beanstalk-1

Platform

Change version

Platform

Node.js 18 running on 64bit Amazon Linux 2023/6.0.3

Running version

-

Platform state

Supported

Events

Health

Logs

Monitoring

Alarms

Managed updates

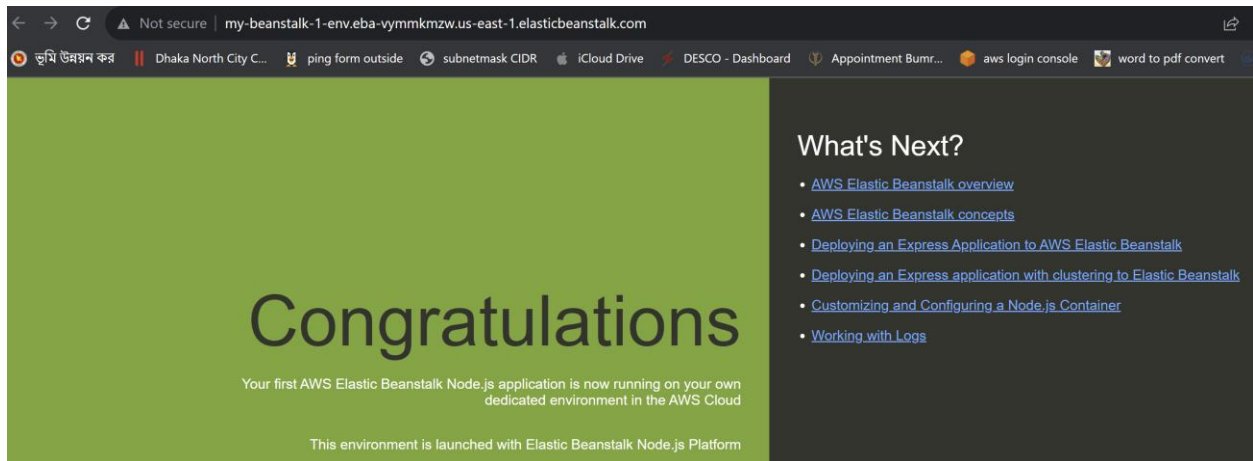
Tags

Events (8)

Filter events by text, property or value

| Time | Type | Details |
|------------------------------------|------|---|
| November 18, 2023 18:06:37 (UTC+6) | INFO | Application available at My-beanstalk-1-env.eba-vymmkmw.us-east-1.elasticbeanstalk.com. |
| November 18, 2023 18:06:22 (UTC+6) | INFO | Instance deployment completed successfully. |
| November 18, 2023 18:05:34 (UTC+6) | INFO | Waiting for EC2 instances to launch. This may take a few minutes. |
| November 18, 2023 18:05:18 (UTC+6) | INFO | Created EIP: 54.164.48.43 |

After completing we can browse the domain

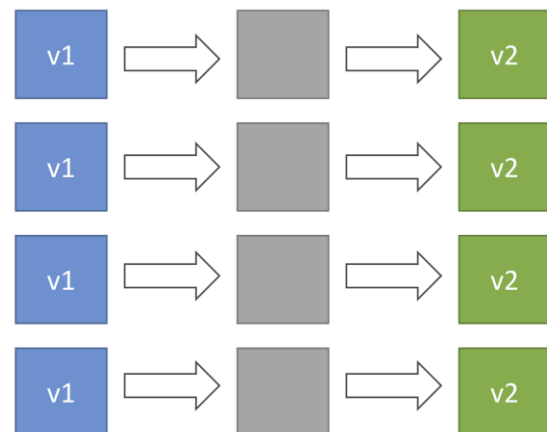


Beanstalk Deployment Options for Updates

- All at once (deploy all in one go) – fastest, but instances aren't available to serve traffic for a bit (downtime)
- **Rolling**: update a few instances at a time (bucket), and then move onto the next bucket once the first bucket is healthy
- **Rolling with additional batches**: like rolling, but spins up new instances to move the batch (so that the old application is still available)
- **Immutable**: spins up new instances in a new ASG, deploys version to these instances, and then swaps all the instances when everything is healthy
- **Blue Green**: create a new environment and switch over when ready
- **Traffic Splitting**: canary testing – send a small % of traffic to new deployment

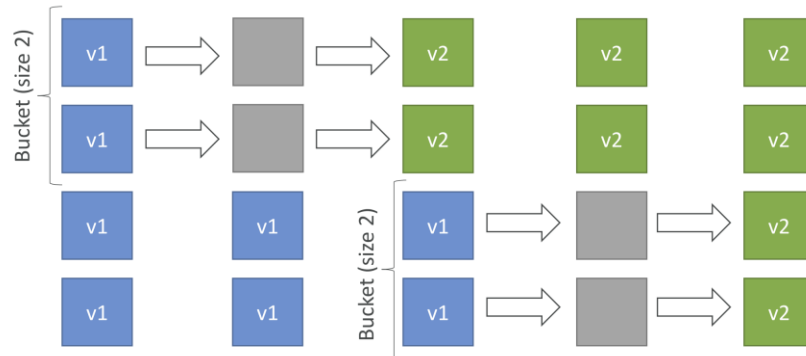
Elastic Beanstalk Deployment All at once

- Fastest deployment
- Application has downtime
- Great for quick iterations in development environment
- No additional cost



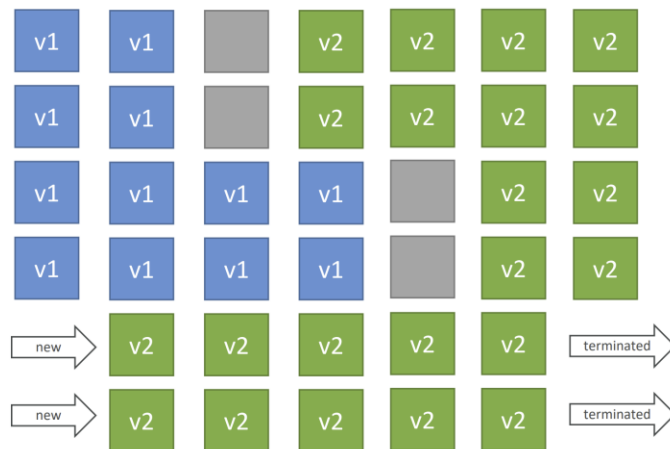
Elastic Beanstalk Deployment Rolling

- Application is running below capacity
- Can set the bucket size
- Application is running both versions simultaneously
- No additional cost
- Long deployment



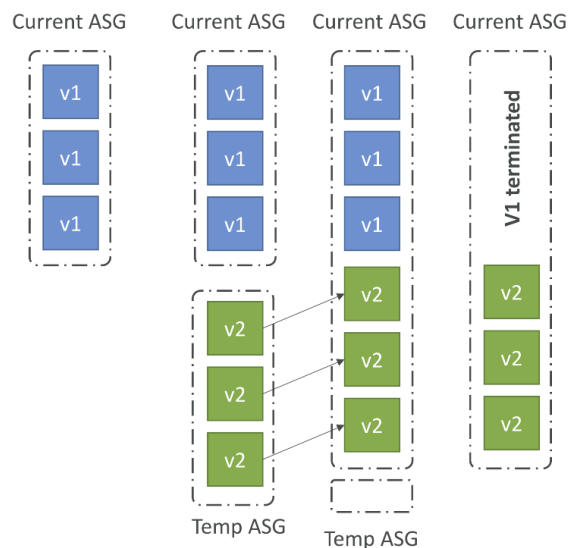
Elastic Beanstalk Deployment Rolling with additional batches

- Application is running at capacity
- Can set the bucket size
- Application is running both versions simultaneously
- Small additional cost
- Additional batch is removed at the end of the deployment
- Longer deployment
- Good for prod



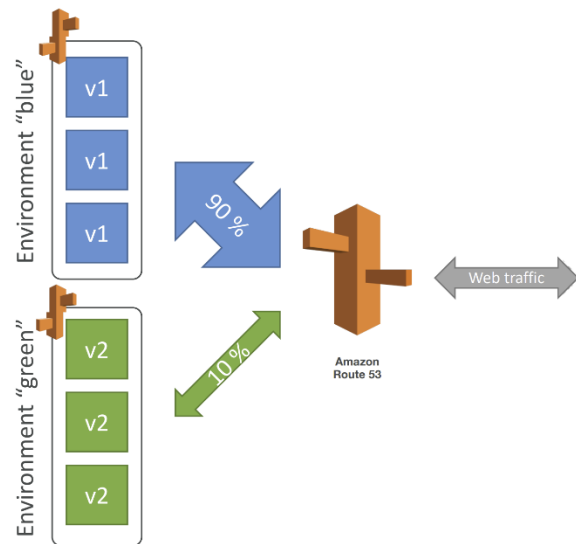
Elastic Beanstalk Deployment Immutable

- Zero downtime
- New Code is deployed to new instances on a temporary ASG
- High cost, double capacity
- Longest deployment
- Quick rollback in case of failures (just terminate new ASG)
- Great for prod



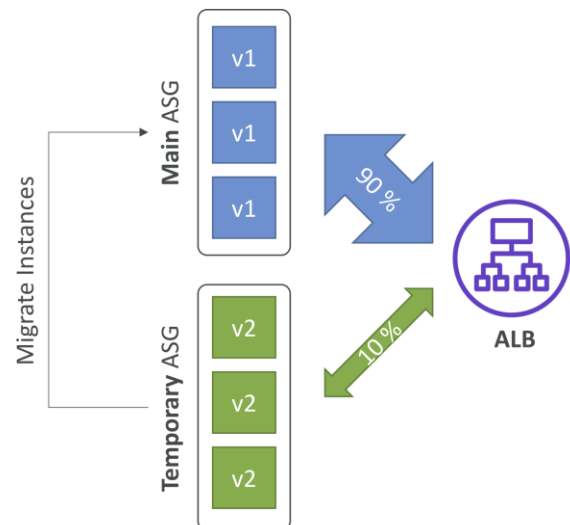
Elastic Beanstalk Deployment Blue / Green

- Not a “direct feature” of Elastic Beanstalk
- Zero downtime and release facility
- Create a new “stage” environment and deploy v2 there
- The new environment (green) can be validated independently and roll back if issues
- Route 53 can be setup using weighted policies to redirect a little bit of traffic to the stage environment
- Using Beanstalk, “swap URLs” when done with the environment test



Elastic Beanstalk -Traffic Splitting

- Canary Testing
- New application version is deployed to a temporary ASG with the same capacity
- A small % of traffic is sent to the temporary ASG for a configurable amount of time
- Deployment health is monitored
- If there's a deployment failure, this triggers an **automated rollback (very quick)**
- No application downtime
- New instances are migrated from the temporary to the original ASG
- Old application version is then terminated



Elastic Beanstalk Deployment Summary from AWS Doc

| Deployment methods | | | | | | |
|----------------------------------|--|---------------|---------------|---------------|---|----------------------------|
| Method | Impact of failed deployment | Deploy time | Zero downtime | No DNS change | Rollback process | Code deployed to |
| All at once | Downtime | ☹ | x | ✓ | Manual redeploy | Existing instances |
| Rolling | Single batch out of service; any successful batches before failure running new application version | ☹ ☹ † | ✓ | ✓ | Manual redeploy | Existing instances |
| Rolling with an additional batch | Minimal if first batch fails; otherwise, similar to Rolling | ☹ ☹ ☹ † | ✓ | ✓ | Manual redeploy | New and existing instances |
| Immutable | Minimal | ☹ ☹ ☹ ☹ | ✓ | ✓ | Terminate new instances | New instances |
| Traffic splitting | Percentage of client traffic routed to new version temporarily impacted | ☹ ☹ ☹ ☹ †† | ✓ | ✓ | Reroute traffic and terminate new instances | New instances |
| Blue/green | Minimal | ☹ ☹ ☹ ☹ | ✓ | x | Swap URL | New instances |

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.deploy-existing-version.html>

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 Deployment Process

- Describe dependencies
(requirements.txt for Python, package.json for Node.js)
- Package code as zip, and describe dependencies
 - Python: requirements.txt
 - Node.js: package.json
- **Console:** upload zip file (creates new app version), and then deploy
- **CLI:** create new app version using CLI (uploads zip), and then deploy
- Elastic Beanstalk will deploy the zip on each EC2 instance, resolve dependencies and start the application