Instantiating application quickly

EC2 Instances:

- **Use a Golden AMI:** install your applications, O.S, dependencies etc beforehand and launch your EC2 instance from the Golden AMI.
- **Bootstrap Using User Data :** for dynamic configuration, use user data script.

- **RDS Databases:**
  - **Restore from a snapshot:** the database will have schemas and data ready!
  - **EBS Volumes:**
    - **Restore from a snapshot :** the disk already be formatted and have data

Developer problems on AWS

- Managing infrastructure
- Deploying code
- Configuring all the database, 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

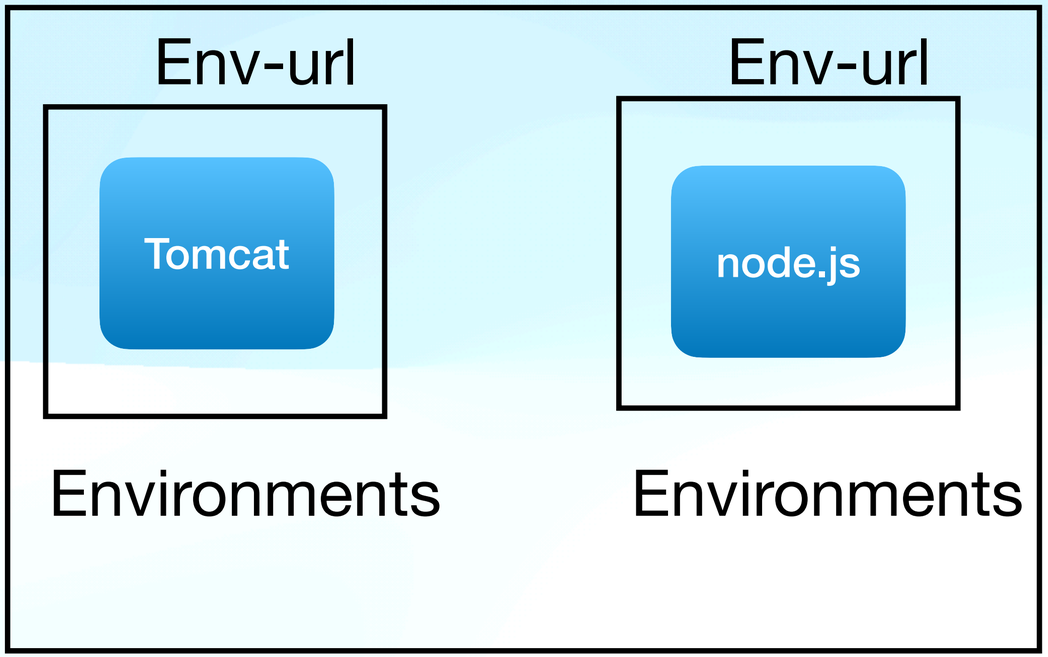
ElasticBeanStalk vs Iron Man

- ElasticBeanStalk is used for easy and quick deployment of applications (PAAS)
- **Iron Man**: The suit manages flight, weapons, defense systems, and even medical care. Tony Stark doesn't have to manually control every aspect; the suit handles it.
- **Elastic Beanstalk**: It manages the infrastructure and deployment of your application. You don't have to worry about provisioning servers, load balancing, scaling, or monitoring. Elastic Beanstalk handles these aspects for you.

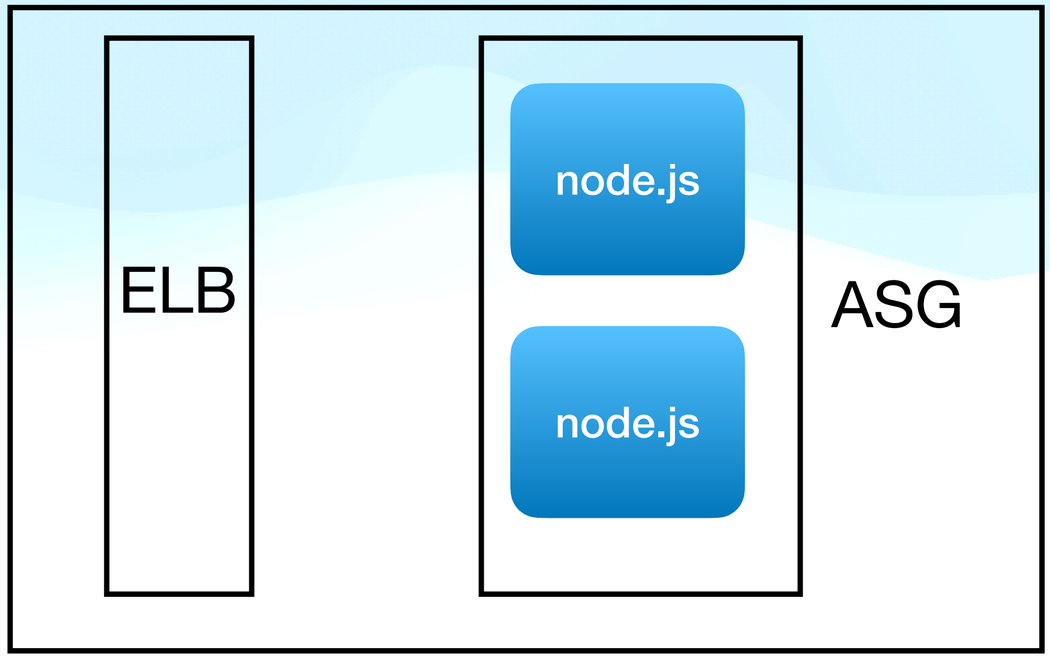
ElasticBeanStalk Overview

- Elastic Beanstalk is a developer centric view of deploying and 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 underling instances**

ElasticBeanStalk Architecture



Single instance deployment



HA instance deployment

ElasticBeanStalk Components

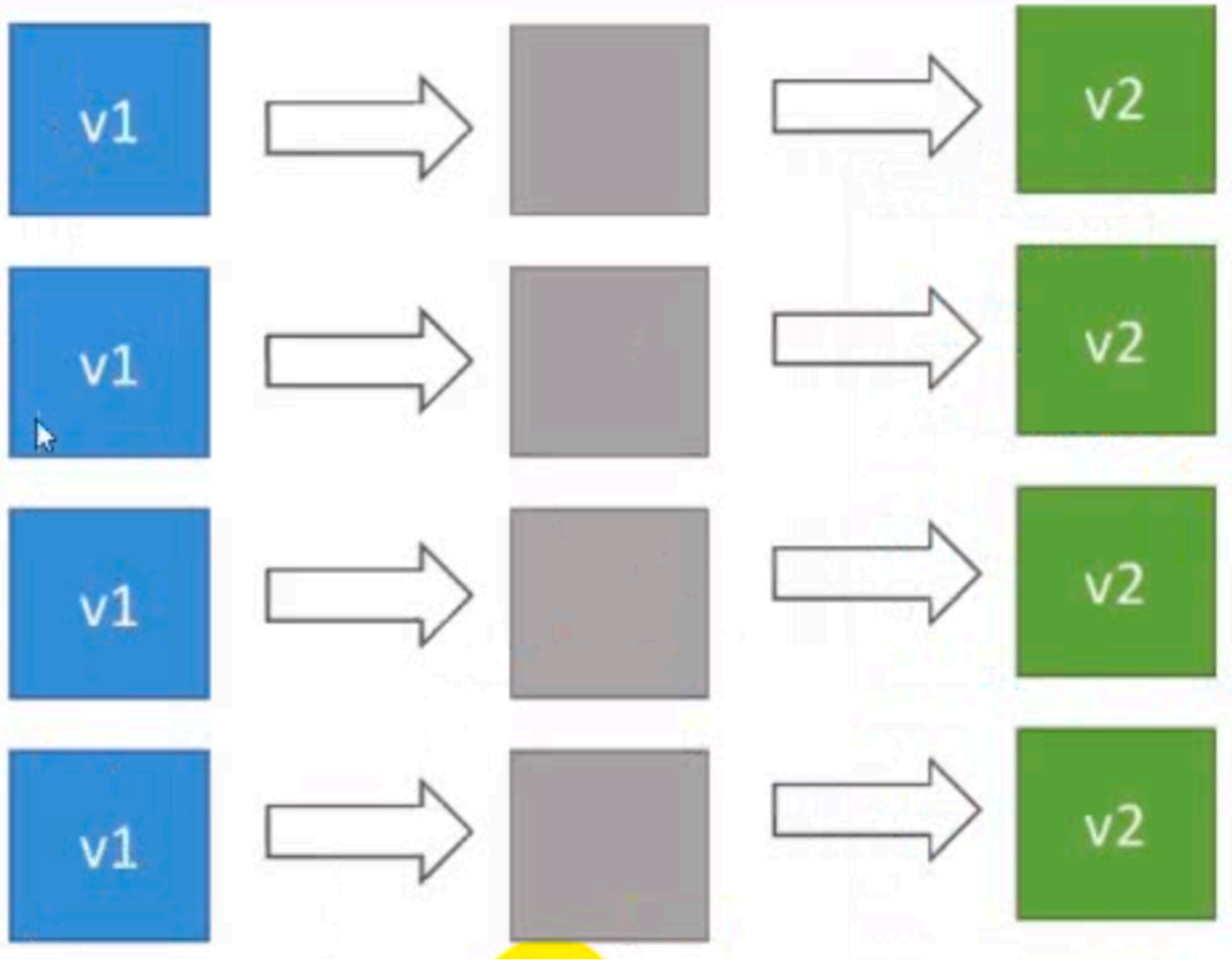
- Application: collection of Elastic Beanstalk components (environments, versions, configurations)
- Application version: and iteration of your application code
- Environment
  - Collection of AWS resources running an application version (only one application version at a time)
  - Tiers : web server Environment tier & worker environment tier.
  - You can create multiple environments, (dev, test, prod etc)

ElasticBeanStalk Deployment options for update

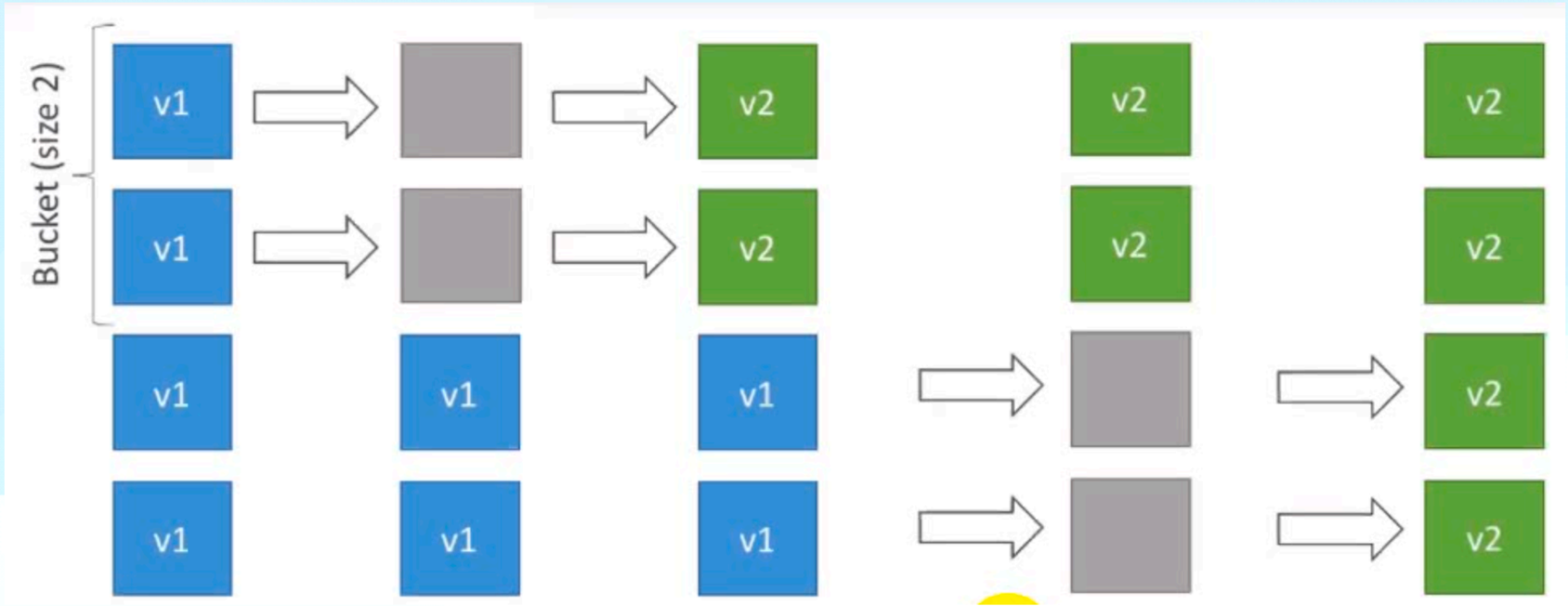
- **All at once (deploy all in one go)** - fastest, but instances aren't available to serve traffic for a bit (downtime)
- **Rolling :** update a few instances at a time (bucket) and then move onto the next bucket once the first bucket is healthy
- **Rolling with additional batches** : like rolling but spins up new instances to move the batch (so that the old application is still available)
- **Immutable :** spins up new instances in a new AST, deploys versions to these instances, and then swaps all the instance when everything is helathy

Beanstalk Deployment - All at once

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

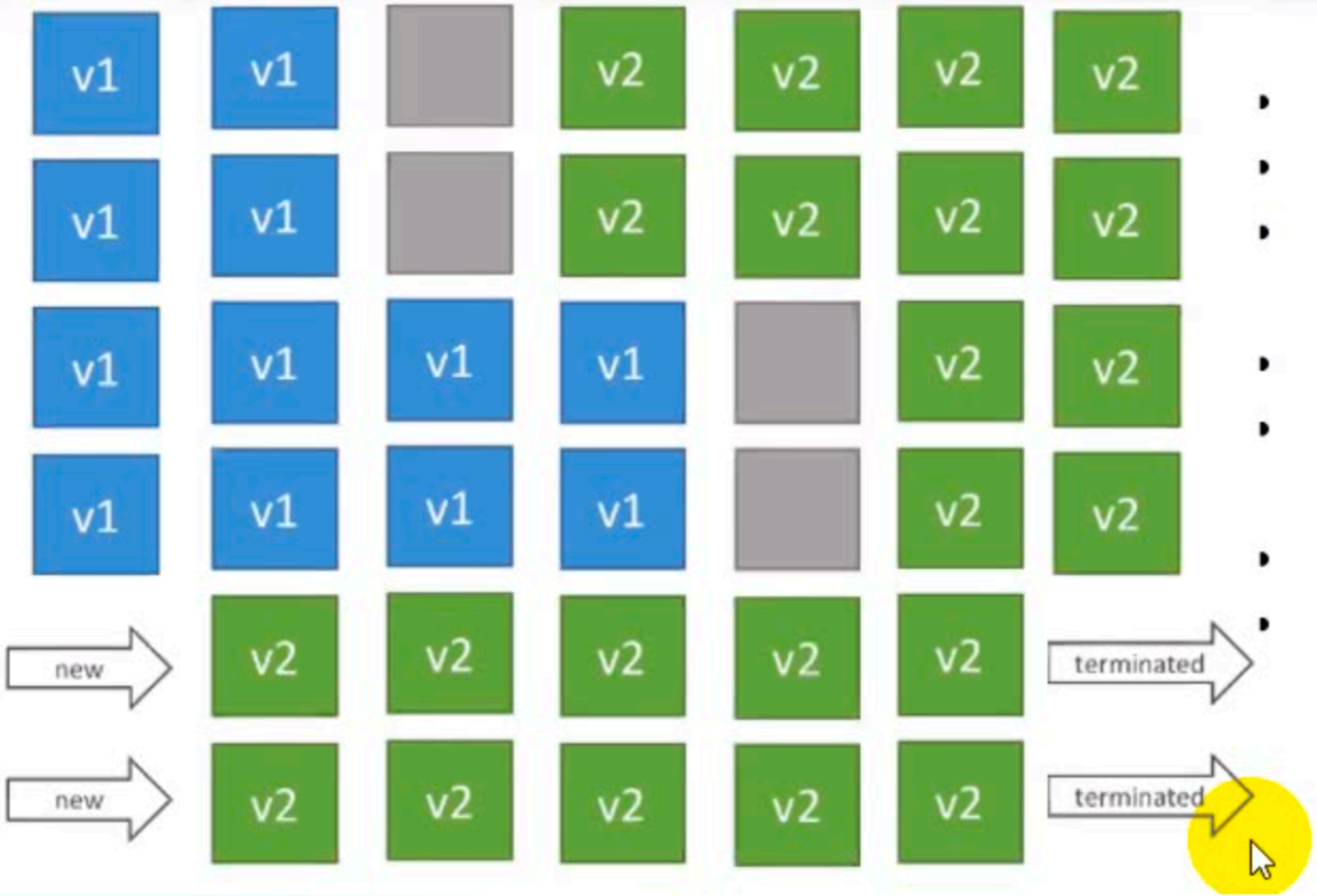


Beanstalk Deployment - Rolling



- Application is running below capacity
- Can set the bucket size
- Application at some point of time running both versions simultaneously
- No Additional cost

Beanstalk Deployment - Rolling with additional batches



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

Beanstalk Deployment - Immutable

- Zero downtime
- New code is deployed to new instances, on a temp ASG
- High cost, double capacity
- Longest deployment
- Quick rollback in case of failure (Just terminate new ASG)
- Great for Production