AWS Lambda & Elastic BeanStalk

AWS Lambda

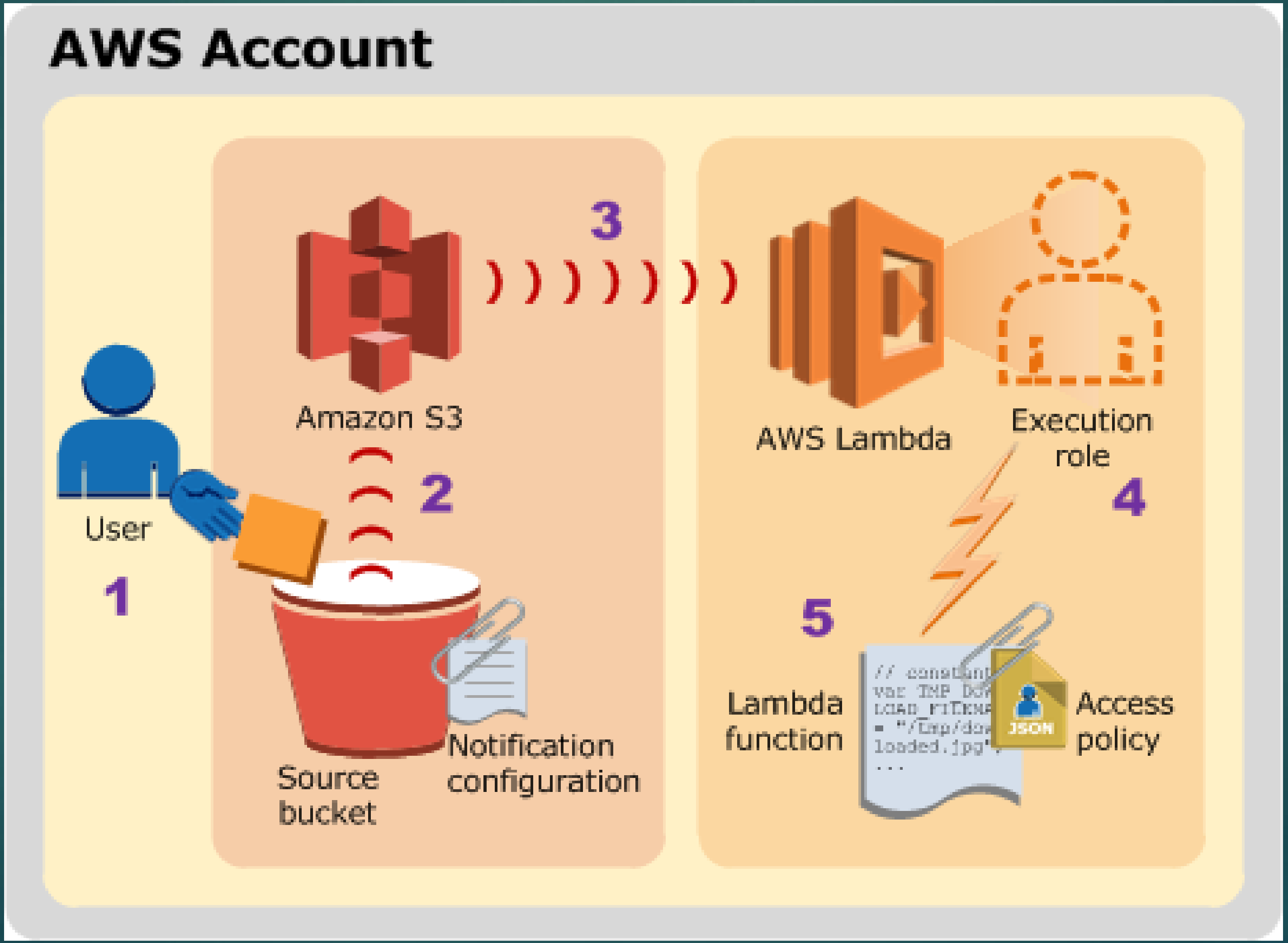
- Compute service that lets you run code without provisioning or managing servers.
- AWS Lambda executes code only when needed and scales automatically, from a few requests per day to thousands per second.
- Pay only for the compute time you consume - there is no charge when your code is not running.
- With AWS Lambda, Run code for virtually any type of application or backend service - all with zero administration.
- AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging.

How does Lambda Work

- Supply your code(function) in one of the languages that AWS Lambda supports (currently Node.js, Java, C#, Go and Python).

- Use AWS Lambda to run your code in response to events, such as changes to data in an Amazon S3 bucket or an Amazon DynamoDB table; to run your code in response to HTTP requests using Amazon API Gateway; or invoke your code using API calls made using AWS SDKs.

Use Case


AWS Account
3
Amazon S3
AWS Lambda
Execution role
User
2
4
1
5
Lambda function
// constant
var TMP_DOW
LOAD_FILENA
= "/tmp/dow
loaded.jpg"
...
JSON
Access policy
Source bucket
Notification configuration

Limitations

Resource	Limits
Memory allocation range	Minimum = 128 MB / Maximum = 3008 MB (with 64 MB increments). If the maximum memory use is exceeded, function invocation will be terminated.
Ephemeral disk capacity ("/tmp" space)	512 MB
Number of file descriptors	1,024
Number of processes and threads (combined total)	1,024
Maximum execution duration per request	300 seconds
Invoke request body payload size (RequestResponse/synchronous invocation)	6 MB
Invoke request body payload size (Event/asynchronous invocation)	128 K

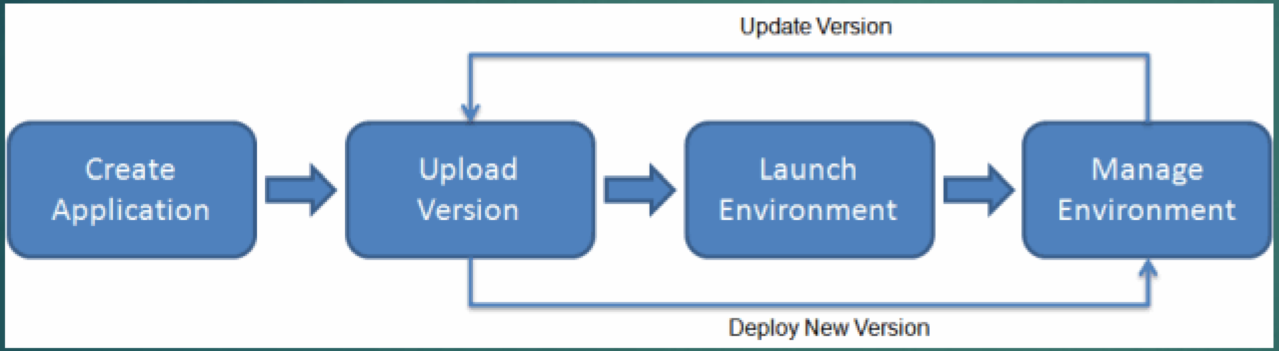
Hands – On- Lab

Overview of Elasticbeanstalk

- Quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications.
- AWS Elastic Beanstalk reduces management complexity without restricting choice or control.
- Simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.
- Elastic Beanstalk uses highly reliable and scalable services that are available in the AWS Free Tier.
- Elastic Beanstalk supports applications developed in Java, PHP, .NET, Node.js, Python, and Ruby, as well as different container types for each language.
- Interact with Elastic Beanstalk by using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or eb, a high-level CLI designed specifically for Elastic Beanstalk

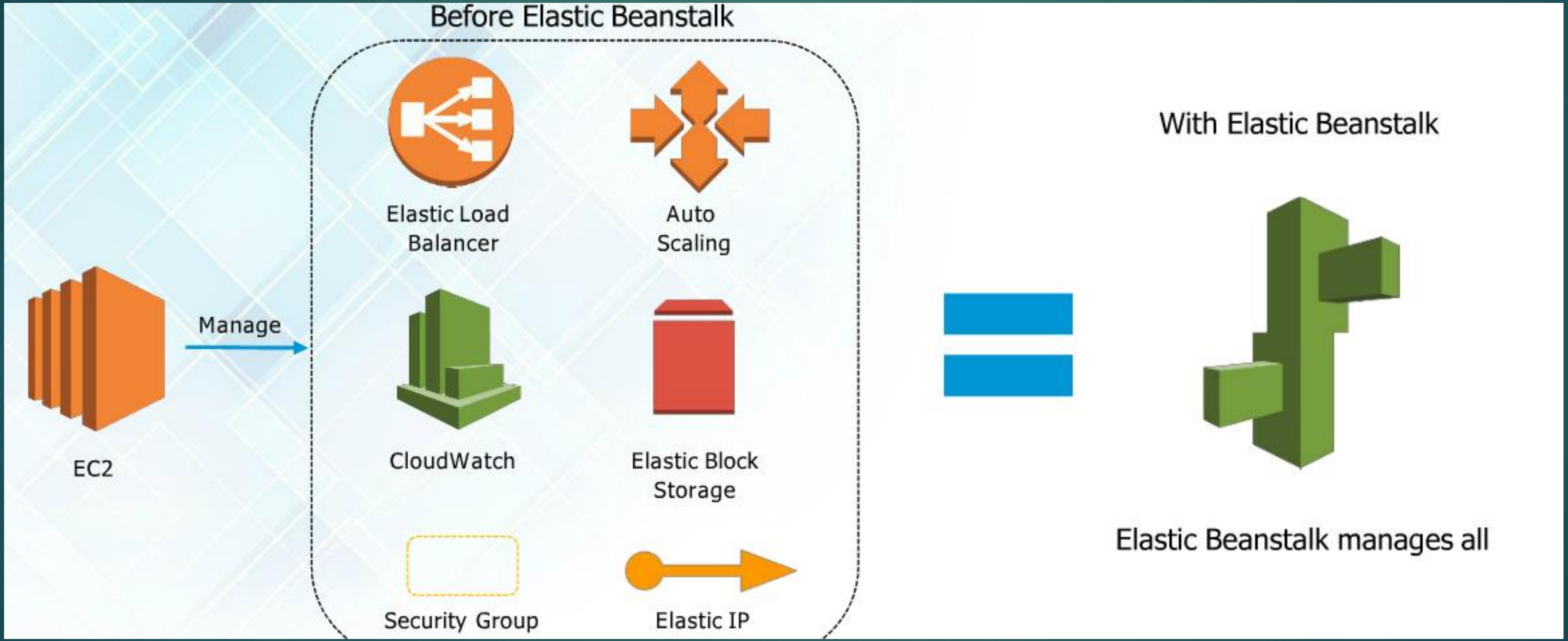
Using ElasticBeanstalk

- Create an application, upload an application version in the form of an application source bundle (for example, a Java .war file) to Elastic Beanstalk, and then provide some information about the application.
- Elastic Beanstalk automatically launches an environment and creates and configures the AWS resources needed to run your code.
- After your environment is launched, you can then manage your environment and deploy new application versions. The following diagram illustrates the workflow of Elastic Beanstalk.



- After you create and deploy your application, information about the application—including metrics, events, and environment status—is available through the AWS Management Console, APIs, or Command Line Interfaces, including the unified AWS CLI

Before and after using ElasticbeanStalk



Hands – On- Lab

THANK YOU