Lecture: Shared Responsibility Model

Volume 90%

In this video, we'll explain the AWS Shared Responsibility Model. AWS is responsible for securing the underlying infrastructure that supports its cloud platform. You, the customer, are responsible for any resources you operate on, place in, or connect to AWS. This shared responsibility model can reduce your total cost of ownership and help improve your security posture without the undifferentiated heavy lifting involved in an on-premises infrastructure.

Lesson Resources:

* [Shared Responsibility Model](https://aws.amazon.com/compliance/shared-responsibility-model/)
* [Penetration Testing](https://aws.amazon.com/security/penetration-testing/)

## Lecture: IAM: Users and Groups

This lesson covers IAM users, which you create to provide authentication for people and processes in your AWS account. You will also learn about IAM groups, which are collections of IAM users you can manage as a unit.

## Lecture: IAM: Policies

Knowing how to create policies is at the very center of IAM management. We use these policies to enable permissions for users, groups, and roles. This lesson discusses both the AWS managed policies as well as custom policies and how to create them.

## Lecture: IAM: Roles

Roles are a way for us to grant our AWS resources permission to interact with each other. We can also grant temporary permissions to users outside of our AWS environment by using roles with delegation and federation. This lesson discusses roles and how we use them.

## Lecture: IAM: Multi-Factor Authentication (MFA)

MFA adds extra security because it requires users to provide unique authentication from an AWS-supported MFA mechanism in addition to their regular sign-in credentials when they access AWS websites or services.

Read more about MFA form factors and virtual MFA applications in the [MFA documentation](https://aws.amazon.com/iam/details/mfa/).

## Lecture: Amazon S3: Bucket Policies

S3 bucket policies allow us to have fine, granular control over the access of objects in our S3 buckets. Using these policies, we can implement additional layers of security and access control for objects. This lesson discusses what these policies are and how to implement them in S3.

This sample bucket policy allows for both SSE-S3 and SSE-KMS based encrypted objects while denying everything else:

{

"Version": "2012-10-17",

"Id": "PutObjPolicy",

"Statement": [

{

"Sid": "DenyUnEncryptedObjectUploads",

"Effect": "Deny",

"Principal": {

"AWS": "\*"

},

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::your-bucket/\*",

"Condition": {

"StringNotEquals": {

"s3:x-amz-server-side-encryption": [

"AES256",

"aws:kms"

]

}

}

},

{

"Sid": "DenyUnEncryptedObjectUploads",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::your-bucket/\*",

"Condition": {

"Null": {

"s3:x-amz-server-side-encryption": "true"

}

}

}

]

}

You can read more about specifying conditions in a policy [here](https://docs.aws.amazon.com/AmazonS3/latest/dev/amazon-s3-policy-keys.html#object-keys-in-amazon-s3-policies).

## Lecture: S3: Data Integrity

S3 offers several features that can assist with data integrity in addition to its built-in availability and durability features. We also need methods to protect our objects from human error. This lesson discusses versioning, replication, and multi-factor authentication delete.

Also, in the AWS documention, there's a table [comparing the Amazon S3 storage classes](https://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html#sc-compare).

## Lecture: Amazon VPC: Security Groups and NACLs

Security groups and Network Access Control Lists (NACLs) are extremely important when looking to lock down the security of our applications. This lesson discusses a strategy to make them work together more efficiently. Misconfiguring either one of these tools can lead to a lot of time troubleshooting.

Here is the [example CloudFormation template](https://github.com/linuxacademy/content-aws-soa-c01/blob/master/wordpress.yaml) used in this lesson.

## Lecture: AWS STS: Federation

Identity federation needs to be understood for us to administer our applications on AWS. STS affects both customers and employees. For users, they often need to authenticate into our applications with third-party providers. For employees, they can use their domain credentials from on-premises services such as Active Directory to authenticate into and use AWS services. This lesson discusses these scenarios in further detail.

### **Helpful Links**

* [Web Identity Federation Playground](https://web-identity-federation-playground.s3.amazonaws.com/index.html)
* [JSON Web Tokens](https://jwt.io/)

## Lecture: Amazon Inspector

Inspector gives us the ability to evaluate our EC2 instances against a built-in library of best practices, common compliance standards, and public libraries of known vulnerabilities. AWS has compiled libraries using several focus areas for evaluating your instances. This lesson shows how to install the Inspector agent on EC2 instances and how to configure targets, templates, and runs from within Inspector.

Here's the AWS documentation on [installing Amazon Inspector agents](https://docs.aws.amazon.com/inspector/latest/userguide/inspector_installing-uninstalling-agents.html).

## Lecture: AWS KMS Essentials

AWS Key Management Service (KMS) is a managed service that makes it easy to create and control the encryption keys used to encrypt data. It's a FIPS 140-2 level 2 compliant service, and in this lesson, we walk through its architecture and key points as they relate to real-world usage and the exam.

## Lecture: AWS Certificate Manager

ACM handles the complexity of creating and managing SSL certificates for your web applications. You can also use public certificates provided by ACM for free.

You can create a listener on your load balancer that uses encrypted connections (also known as SSL offload). This feature enables traffic encryption between your load balancer and the clients that initiate SSL sessions.

To use an HTTPS listener, you must deploy a certificate on your load balancer. The load balancer uses this certificate to terminate the connection and then decrypt requests from clients before sending them to the targets.

In this lesson, we'll cover how to create a certificate in ACM, bind that to the load balancer, and browse to it using HTTPS with the domain name provided in the Route 53 hosted zone.

Course GitHub repo: <https://github.com/linuxacademy/content-aws-soa-c01>

## Lecture: AWS Web Application Firewall (WAF)

The AWS WAF service protects your web applications from common exploits that could affect availability, compromise security, or consume excessive resources. WAF monitors HTTP requests directed at Amazon CloudFront, API Gateway, or an Application Load Balancer.

### **Helpful Links**

* [CloudFormation template used in this lesson](https://github.com/linuxacademy/content-aws-soa-c01/blob/master/waf.yaml)
* [Linux Academy Instant Terminal](https://ssh.instantterminal.linuxacademy.com/)

## Lecture: AWS Trusted Advisor

AWS Trusted Advisor can help you reduce costs, increase performance, and improve the security of your AWS environments. It provides real-time guidance to help provision resources following AWS best practices.

# **Auditing Resource Compliance with AWS Config**

Learning Objectives

check\_circle**Enable AWS Config**

1. Navigate to the AWS Config Console.
2. Create S3 bucket.
3. Create AWS Config.

check\_circle**Configure Rules for Resources**

1. Select the cloudtrail-enabled card.
2. Add the rules.
3. Select the desired-instance-type card.
4. Add the rules.
5. Select the ec2-instances-in-vpc card.
6. Add the rules.
7. Select the s3-bucket-versioning-enabled card.
8. Add the rules.
9. Save configuration settings.

check\_circle**Configure the Non-Compliant Resources to Comply**

1. Open S3.
2. Edit settings and save.
3. Open CloudTrail and create a new trail named ConfigTrail.
4. Create a new S3 bucket and give it a unique name.

# Auditing Resource Compliance with AWS Config

## Introduction

In this hands-on lab, we'll implement AWS Config rules and use AWS Config for compliance auditing and remediation. We will configure compliance rules for evaluating EC2 instance type, if S3 versioning is enabled, EC2 instances in a VPC, and if CloudTrail is enabled. These rules will give you firsthand knowledge about how the AWS Config service works. We will then explore the configuration management aspect of AWS Config.

## Solution

Log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 region.

### **Enable AWS Config**

1. In the top-left corner of the page click **Services**, and under Management & Governance, click **Config**.
2. Click **Get Started**.
3. Leave the general settings as default and click **Next**.
4. Click **Next** and click **Confirm**.
5. Once enabled, click **View dashboard**.

### **Configure Rules for Resources**

1. In the left-hand menu click **Rules**.
2. Click **Add rule**.
3. In the search box, type "cloudtrail", select **cloudtrail-enabled**, and click **Next**.
4. Leave the configure rule settings as default and click **Next**.
5. Review the settings and click **Add rule**.
6. Click **Add rule**.
7. In the search box, type "desired", select **desired-instance-type**, and click **Next**.
8. Under Parameters, for the InstanceType Key Value, enter "t2.micro" and click **Next**.
9. Review the settings and click **Add rule**.
10. Click **Add rule**.
11. In the search box, type "ec2-instances", select **ec2-instances-in-vpc**, and click **Next**.
12. Under Parameters, for the vpcid Key Value, enter the VPC ID for the VPC provided with the lab.

Note: This can be found by navigating to the VPC section of the console under **Your VPCs**, and opening the details for the SysOpsVPC.

1. Review the settings and click **Add rule**.
2. Click **Add rule**.
3. In the search box, type "versioning", select **s3-bucket-versioning-enabled**, and click **Next**.
4. Leave the configure rule settings as default and click **Next**.
5. Review the settings and click **Add rule**.
6. Review the non-compliant rules.

### **Configure the Non-Compliant Resources to Comply**

1. In the top-left corner of the page, open the **Services** menu, and click **CloudTrail**.
2. In th left-hand menu, click **Trails**, and click **Create trail**.
3. For the Trail name, enter "ConfigTrail".
4. Disable the **Log file SSE-KMS encrpytion** setting. Keep other settings as default and click **Next**.
5. Under Event Type, make sure that **Management Events** is selected, and that all read and write activity is logged.
6. Click **Next**.
7. Review the trail and click **Create trail** (please see Steps 1-3 in the last objective for the exact next steps in the video).
8. In the top-left corner of the page, open the **Services** menu, and click **S3**.
9. In the Buckets list, open the config-bucket S3 bucket.
10. Click the **Properties** tab, and under Bucket Versioning, click **Edit**.
11. Enable Bucket Versioning and click **Save changes**.
12. Back in the Buckets list, open the cloudtrail-logs bucket.
13. Click the **Properties** tab, and under Bucket Versioning, click **Edit**.
14. Enable Bucket Versioning and click **Save changes**.

### **Re-Evaluate the Non-Compliant Rules in AWS Config**

1. Navigate to the AWS Config Console, click **Rules**.
2. Open the cloudtrail-enabled rule, and in the top-right corner, open the **Actions** menu.
3. Click **Re-evaluate** (this may take a few minutes).
4. Back in the AWS Config Console, click **Rules**.
5. Open the s3-bucket-versioning-enabled rule, and in the top-right corner, open the **Actions** menu.
6. Click **Re-evaluate**.
7. Review the rules and verify they are now compliant.

## Conclusion

Congratulations — you've completed this hands-on lab!

## Lecture: Amazon EC2 Status Checks

Status checks are made up of instance status checks and system status checks. Errors in these indicate different issues and should, therefore, be recovered differently. This lesson discusses AWS system and instance status checks and how to recover from an error in these checks.

## Lecture: EC2 Instance Types and Performance

There is a wide variety of EC2 instance types available. Some families are more suited for particular tasks. There is also a large difference in most families between the smallest and largest instance size that can affect performance. This lesson discusses the virtualization types, instance types, and instance sizes we can configure for our EC2 instances.

Note: For complete EC2 purchase options, see: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html>

## Lecture: EC2: Scale Out or Scale Up?

In AWS, scaling out means horizontal scaling by increasing the number of instances in an Auto Scaling group. Scaling up refers to vertical scaling by increasing the instance size or family. This lesson discusses decisions administrators need to make regarding Auto Scaling versus increasing instance size.

## Lecture: EC2: NAT Gateways and Bastion Hosts

In the event of an Availability Zone (AZ) failure, instances will still need access to the internet for updates. If multiple NAT gateways are deployed, we can allow for this. Bastion hosts also need to be available when an AZ fails. This lesson discusses how to make bastion hosts and NAT gateways fault tolerant.

## Lecture: EC2: Reserved Instances

EC2 reserved instances can be an effective method of saving money if long-term compute capacity is needed. They can also reserve us capacity in case of an Availability Zone or region shortage of on-demand instances. In this lesson, we discuss reserved instances in greater detail, as well as provide scenarios that show the benefits of using them.

For more information on reserved instances, see the ["On-Demand Capacity Reservations"](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-capacity-reservations.html) documentation.

## Lecture: EC2: Initializing Volumes

When restoring a volume from a snapshot, maximum volume performance is not achieved until all blocks on the device have been read. This lesson discusses initializing EBS volumes and when we should use it. The commands from the lesson are:

sudo dd if=/dev/nvme2n1 of=/dev/null bs=1M

sudo yum install -y fio

sudo fio --filename=/dev/nvme2n1 --rw=read --bs=128k --iodepth=32 --ioengine=libaio --direct=1 --name=volume-initialize

For more information on initializing volumes, see the ["Initializing Amazon EBS Volumes"](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-initialize.html) documentation.

## Lecture: EC2: Troubleshooting Auto Scaling Issues

Auto Scaling issues can be difficult to troubleshoot. There are many different configuration steps and items that can lead to problems. This lesson discusses many of the common issues when Auto Scaling is not working.

## Lecture: Amazon Lightsail and AWS Batch

Lightsail is a virtual private server (VPS) solution from AWS. It allows for a monthly "rental" of an instance. Batch is a fully managed AWS service for handling batch computing jobs. These compute services are mentioned in AWS documentation, and they are new in terms of the exam. This lesson is a quick overview of what Lightsail and Batch are and what they do.

# **Troubleshooting Amazon EC2 Network Connectivity**

Additional Information and Resources

Please make sure you are working in the us-east-1 (N. Virginia) region.

Goal: Fix the connectivity issue in the AWS environment so that you can update the yum package installer (from the command line) on the provided EC2 instance (named "web server"). This environment has been created with security in mind, so the "web server" EC2 instance has been provisioned in a private subnet and placed behind a bastion host and NAT gateway.

The Linux command to update the yum package installer is sudo yum update.

Learning Objectives

check\_circle**Fix SSH Ingress to Bastion Host**

#### **The Issue**

SSH traffic is being denied by the security group associated with the bastion host.

#### **How to Fix the Issue**

Add an SSH (port 22) allow rule to the security group associated with the bastion host.

check\_circle**Fix Egress from Web Server to Internet**

#### **The Issue**

The NACL protecting the web server only allows return traffic to the public subnet, not the internet.

#### **How to Fix the Issue**

Add an outbound "all traffic" allow rule to 0.0.0.0/0 to the NACL.

check\_circle**Fix Web Server Route to Internet**

#### **The Issue**

The web server does not have a route to the NAT gateway.

#### **How to Fix the Issue**

Add a route to the NAT gateway on the route table associated with the private subnet the web server is located in.

# Troubleshooting Amazon EC2 Network Connectivity

## Introduction

The goal of this hands-on lab is to fix the connectivity issue in the AWS environment so we can update the yum package installer (from the command line) on the provided EC2 instance (named "web server"). Here, we'll go step-by-step through the scenario and offer detailed instructions on how to solve the connectivity issue.

## Solution

Log in to the live AWS environment using the credentials provided. Make sure you're in the N. Virginia (us-east-1) region throughout the lab.

### **Fix SSH Ingress to Bastion Host**

#### **The Issue**

SSH traffic is being denied by the security group associated with the bastion host.

#### **How to Fix the Issue**

Add an SSH (port 22) allow rule to the security group associated with the bastion host

1. Locate the public IP address of the bastion host on the lab page and copy it. Alternately, in the AWS console, navigate to **EC2** > **Instances** and copy it from there.
2. Open a terminal session, and log in via SSH:

ssh cloud\_user@<PUBLIC IP ADDRESS>

1. It doesn't seem to connect. Head back to the AWS console to look at the bastion host.
2. Click the listed security group associated with the bastion host.
3. In the Inbound rules tab, we'll see the only allow rule is port 80, which is for HTTP traffic and not SSH traffic.
4. Click **Edit**.
5. Delete the existing rule.
6. Click **Add Rule**, and set the following values:
   * Type: **SSH**
   * Protocol: **TCP**
   * Port Range: **22**
   * Source: **Anywhere**
7. Click **Save**.
8. Back in the terminal, we should see the prompt to continue connecting. Enter yes, and then enter the password for the instance (provided on the lab page). We've now successfully logged in to the bastion host.
9. Now, we need to log in to the "web server" instance. Copy the private IP address from the lab credentials page (or in the AWS console). In the terminal, enter:

ssh cloud\_user@<PRIVATE IP ADDRESS>

Enter yes at the prompt, and then enter the password provided on the lab page for the web server instance.

1. Now, run the YUM package installer:

sudo yum update

1. Enter the password again.

There seems to be a hangup. Why is the EC2 instance not able to connect to the open internet in order to successfully update the YUM package installer?

### **Fix Egress from Web Server to Internet**

#### **The Issue**

The NACL protecting the web server only allows return traffic to the public subnet, not the internet.

#### **How to Fix the Issue**

Add an outbound "All Traffic" allow rule to 0.0.0.0/0 to the NACL.

1. In the AWS console, navigate to **VPC** > **Network ACLs**.
2. Click the **Private Network** NACL listed.
3. In the **Outbound Rules** tab, click **Edit outbound rules**.
4. Change the Destination to **0.0.0.0/0**.
5. Click **Save**.
6. Back in the terminal, run:

sudo yum update

It still won't connect.

### **Fix Web Server Route to Internet**

#### **The Issue**

The web server does not have a route to the NAT gateway.

#### **How to Fix the Issue**

Add a route to the NAT gateway on the route table associated with the private subnet the web server is located in.

1. In the AWS console, navigate to the **VPC** > **Route Tables**.
2. Select the **Private** route table, and click the **Routes** tab. We'll see there isn't a route to the NAT gateway.
3. Click **Edit routes**.
4. Click **Add route**, and set the following values:
   * Destination: 0.0.0.0/0
   * Target: Type "nat", and select the pre-populated NAT gateway listed in the dropdown
5. Click **Save routes**.
6. Back in the terminal, run:

sudo yum update

Note: If you get a lock message, kill -9 PID (replace PID with the process number), then run the yum command again.

It should work this time.

## Conclusion

Congratulations on completing this hands-on lab!

## Lecture: S3: Cross-Region Replication

It may seem strange that we are discussing backups for S3 because it has some impressive numbers for durability and availability. The issue with S3 buckets is that, even though the names are global, they exist in a particular region. If that region were to fail, we would not be able to access our objects. This lesson demonstrates how S3 cross-region replication works and what limitations are involved with this replication.

## Lecture: S3: Storage Classes

This lesson walks through the storage tiers available in S3 in addition to evaluating Intelligent-Tiering and lifecycle policies.

#### **Helpful Links**

* [Amazon S3 Storage Classes](https://aws.amazon.com/s3/storage-classes/)

## Lecture: AWS Storage Gateway

For hybrid environments, ones that include some sort of on-premises infrastructure, AWS provides services to assist with data durability. Storage Gateway provides us a way to back up and even migrate to the cloud. It has three main types, and they all include some sort of on-premises component. This lesson discusses the three types of Storage Gateway offers and how they are used.

## Lecture: AWS Snowball

For massive data transfers, transporting data over the ground can sometimes be faster and cheaper than over the internet. In this lesson, we'll cover AWS Snowball, Snowball Edge, and Snowmobile.

#### **Helpful Links**

* [AWS Snowball](https://aws.amazon.com/snowball/)
* [AWS Snowmobile](https://aws.amazon.com/snowmobile/)

## Lecture: Amazon EBS Essentials

It is important to know the basics of EBS volumes to make informed decisions in our environment. This lesson discusses some of the facts that are necessary to know for administering EBS and for the exam.

## Lecture: EBS: Performance

The performance of your EBS volumes should be a priority in your environment. Storage size, bursting, throughput, and IOPS can all cause issues. Also, if we haven't "tuned" our EBS usage carefully, we could be spending more than we need to. This lesson discusses the performance options a user has when provisioning EBS volumes. For the latest performance information (some has changed), see: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html>

## Lecture: EBS: Metrics

The metrics we need to monitor are varied and greatly depend on the volume type. This lesson discusses EBS metrics in CloudWatch and how we can use them to make performance decisions with our volumes.

## Lecture: EBS: Resizing or Changing Root Volumes

Resizing a root volume is a necessary skill as a Systems Operator. At some point, you will either run out of storage or need better IOPS performance. This video will show a couple of different techniques for resizing or changing a root EBS volume.

For information on extending a Linux file system after resizing, check out: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/recognize-expanded-volume-linux.html>

## Lecture: EBS: Ensuring Data Durability

Knowing how to preserve data is an important tool for any administrator. EBS volumes can behave differently depending on how they are being used. This lesson discusses methods of preserving data when our instances need to be terminated. There is also a short discussion on instance store-backed instances.

## Lecture: Amazon EFS: Deployment and Provisioning

FS is a highly scalable managed file system that can be shared by multiple instances. These attributes make it perfect for a web server data store. We can have many instances running and only have to launch and update our websites in one place. This lesson shows the process.

## Lecture: EFS: Monitoring for Performance and Availability

EFS is a scalable, highly available block storage file system we can use with our EC2 instances and on-premises servers. EFS is becoming a bigger part of all the Associate exams. In this lesson, we discuss a brief overview of what it is. Then, we move on to how to monitor it through CloudWatch.

#### **Helpful Links**

* [Amazon CloudWatch Metrics for Amazon EFS](https://docs.aws.amazon.com/efs/latest/ug/monitoring-cloudwatch.html)

## Lecture: RDS: Scaling for Performance

Read replicas allow us to offload database resources to another instance to improve read performance. Read replicas are also a useful tool for disaster recovery and migrations. In this lesson, we will discuss RDS Read Replicas and how they can help the performance of an application.

#### **Helpful Links**

* [Working with Read Replicas](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ReadRepl.html)

# **Automating EBS Snapshot Creation with CloudWatch Events and SNS**

Additional Information and Resources

Please log in to the AWS environment by using the cloud\_user credentials provided.

Once inside the AWS account, make sure you are using us-east-1 (N. Virginia) as the selected region.

**Note: Please delete your email in the SNS section before closing the lab.**

Learning Objectives

check\_circle**Create One (1) SNS Topic with One (1) SNS Subscription**

We will create a new SNS topic and SNS email subscription that we will use with our CloudWatch Events rule.

check\_circle**Create One (1) CloudWatch Events Rule**

Create the CloudWatch Events rule that will once-daily snapshot the specified EBS volume and initiates an email via SNS.

check\_circle**Verify One (1) EBS Snapshot Has Been Created**

Confirm that your new CloudWatch Events rule is working as expected by checking that a new EBS snapshot was created.

# Automating EBS Snapshot Creation with CloudWatch Events and SNS

## Introduction

In this AWS hands-on lab, we will be creating a CloudWatch Events rule for the purpose of automating the creation of EBS snapshots (and notifying system admins).

Creating and storing EBS snapshots are essentials for any backup/disaster recovery plan. Automating the creation process allows for reliable backup planning. And by utilizing SNS in this process, we can inform an administrator when each snapshot creation job has begun.

## Solution

Log in to the live AWS environment using the credentials provided. Make sure you're in the N. Virginia (us-east-1) region throughout the lab.

**Note: Please delete your email in the SNS section before closing the lab.**

### **Create One (1) SNS Topic with One (1) SNS Subscription**

1. Navigate to SNS using the Services menu or the unified search bar.
2. On the SNS page, in the Create topic box, give it a topic name of EBSSnap.
3. Click **Next step**.
4. Accept the defaults on the Create topic page and click **Create topic**.
5. Click **Create Subscription**.
6. Set the Protocol to **Email**.
7. For Endpoint, enter your email address.
8. Click **Create subscription**.
9. Check your email inbox and confirm the subscription in the email you received.

### **Create One (1) CloudWatch Events Rule**

1. Navigate to EC2 and select **Volumes** in the sidebar menu.
2. Select any of the listed volumes.
3. In the Description section below, copy its volume ID to your clipboard.
4. Navigate to CloudWatch. Below Events in the sidebar menu, select **Rules**.
5. Click **Create rule** at the top of the console.
6. Set the Event Source to **Schedule**, and set the following value:
   * Fixed rate of: **1** **Days**
7. Click **Add target**.
8. Use the main dropdown to choose **EC2 CreateSnapshot API call**, then paste in the volume ID from your clipboard.
9. Select **Create a new role for this specific resource** with the predefined name.
10. Click **Add target**.
11. Use the main dropdown to choose **SNS topic**, and select our newly created **EBSSnap** SNS topic.
12. Click **Configure details**.
13. Give the rule a name of EBSSnap.
14. Make sure State is set to **Enabled**, and click **Create rule**.

### **Verify One (1) EBS Snapshot Has Been Created**

1. Check your inbox for the message from SNS.
2. After about a minute, check in **EC2** > **Snapshots** that a new snapshot has been created.

## Conclusion

Congratulations on completing this hands-on lab!

# **Managing Data in S3 with Versioning and Lifecycle Rules**

Additional Information and Resources

Please log in to the lab environment with the cloud\_user credentials provided.

Make sure you are using the us-east-1 region throughout the learning activity.

**NOTE**: For even more information, including the latest updates, see:

* [Amazon S3 Storage Classes](https://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html)
* [How Do I Create a Lifecycle Policy for an S3 Bucket?](https://docs.aws.amazon.com/AmazonS3/latest/user-guide/create-lifecycle.html)
* [Object Lifecycle Management](https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html)

Learning Objectives

check\_circle**Enable Versioning on the Provided S3 Bucket**

1. Navigate to **S3**.
2. Open the lab-provided S3 bucket.
3. Click the **Properties** tab.
4. Under *Bucket Versioning* click **Edit**, and **Enable**.
5. Click **Save changes**.

check\_circle**Assign a Lifecycle Rule to Objects in the Logs Folder**

1. In the *Management* tab of the bucket, click **Create lifecycle rule**.
2. Enter a name for the rule (e.g., "log-rule").
3. For *Prefix*, enter "Logs".
4. Under *Lifecycle rule actions*, select **Transition current versions of objects between storage classes**.
5. Under **Transition current versions of objects between storage classes**, set the following values:
   * *Storage class transitions*: **Standard-IA**
   * *Days after object creation*: **90**
6. Click **Create rule**.

check\_circle**Assign a Lifecycle Rule to Objects in the Images Folder**

1. Back in the *Management* tab, click **Create lifecycle rule** again.
2. Enter a name for the rule (e.g., "image-rule").
3. For the *Prefix*, enter "Images".
4. Under *Lifecycle rule actions*, select **Transition current versions of objects between storage classes**.
5. Under **Transition current versions of objects between storage classes**, set the following values:
   * *Storage class transitions*: **One Zone-IA**
   * *Days after object creation*: **30**
6. Click **Create rule**.

check\_circle**Add a Lifecycle Rule to Move the Older Log File to Glacier after 180 Days**

1. In the S3 bucket Logs/ folder and add a tag to the most recent log file.
2. Under *Lifecycle rule actions*, select **Transition current versions of objects between storage classes**.
3. Under **Transition current versions of objects between storage classes**, set the following values:
   * *Storage class transitions*: **Glacier**
   * *Days after object creation*: **180**

*Note:* You will be asked if you understand that the lifecycle rule will increase the one-time lifecycle request cost if it transitions small objects.

1. Click **Create rule**.

# Managing Data in S3 with Versioning and Lifecycle Rules

## Introduction

In this hands-on lab, we'll start with enabling versioning in an S3 bucket, then we will configure the lifecycle rules to automatically transition objects to lower-cost storage classes, and finally, we will change the image files to reduced redundancy storage as the organization has means to recreate them if they are lost.

Understanding and utilizing these S3 features are at the core of S3 data management. Common use cases include recovery from accidental deletions, and lifecycle policies allow for automated migration to [lower cost storage classes](https://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html) when appropriate.

Good luck, and enjoy the lab!

## Solution

Log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 region.

### **Enable Versioning on the Provided S3 Bucket**

1. Navigate to **S3**.
2. Open the lab-provided S3 bucket.
3. Click the **Properties** tab.
4. Under Bucket Versioning click **Edit**, and **Enable**.
5. Click **Save changes**.

### **Assign a Lifecycle Rule to Objects in the Log Folder**

1. In the Management tab of the bucket, click **Create lifecycle rule**.
2. Enter a name for the rule (e.g., "log-rule").
3. For Prefix, enter "Logs".
4. Under Lifecycle rule actions, select **Transition current versions of objects between storage classes**.
5. Under **Transition current versions of objects between storage classes**, set the following values:
   * Storage class transitions: **Standard-IA**
   * Days after object creation: **90**
6. Click **Create rule**.

### **Assign a Lifecycle Rule to Objects in the Images Folder**

1. Back in the Management tab, click **Create lifecycle rule** again.
2. Enter a name for the rule (e.g., "image-rule").
3. For the Prefix, enter "Images".
4. Under Lifecycle rule actions, select **Transition current versions of objects between storage classes**.
5. Under **Transition current versions of objects between storage classes**, set the following values:
   * Storage class transitions: **One Zone-IA**
   * Days after object creation: **30**
6. Click **Create rule**.

### **Add a Lifecycle Rule to Move the Older Log File to Glacier after 180 Days**

1. Navigate back to **S3** and open the bucket.
2. Click the Logs/ folder.
3. Open the file titled **2019.csv**.
4. In the top-right corner, click **Object actions**, and from the drop-down select **Edit tags**.
5. Click **Add tag**, and set the following values:
   * Key: **Type**
   * Value: **OldLogs**
6. Click **Save changes**.
7. Navigate back to the bucket, and click the Management tab.
8. Click **Create lifecycle rule**.
9. Enter a name for the rule (e.g., "tagged-logs").
10. Under Object tags, enter the following:
    * Key: **Type**
    * Value: **OldLogs**
11. Under Lifecycle rule actions, select **Transition current versions of objects between storage classes**.
12. Under **Transition current versions of objects between storage classes**, set the following values:
    * Storage class transitions: **Glacier**
    * Days after object creation: **180**

Note: You will be asked if you understand that the lifecycle rule will increase the one-time lifecycle request cost if it transitions small objects.

1. Click **Create rule**.

## Conclusion

Congratulations — you've completed this hands-on lab!

# **Querying Data in S3 with Amazon Athena**

Additional Information and Resources

Make sure you are in the us-east-1 region throughout this lab.

NOTE: This lab will be replaced soon. On the step to create a table: Click on "Connect data source" in the left navigation; on the right side of Data Source, then make sure to select "Query data in Amazon S3" for where the data is located. Also select "AWS Glue data catalog" for the metadata catalog and click "Next". Then we will select the "Add a table and enter schema info manually." Then the choice will update to "Continue to add table." After that add the db name, Table name, and s3 URL as in the video.

When prompted to bulk add column definitions, you may use this data to save time:

time timestamp, location string, bytes bigint, requestip string, method string, host string, uri string, status int, referrer string, useragent string, querystring string, cookie string, resulttype string, requestid string, hostheader string, requestprotocol string, requestbytes bigint, timetaken double, xforwardedfor string, sslprotocol string, sslcipher string, responseresulttype string, httpversion string

CloudFront raw access logs are stored in a CSV format called [Web Distribution Log File Format](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html#BasicDistributionFileFormat).

Learning Objectives

check\_circle**Create a Table from S3 Bucket Metadata**

1. Navigate to the **Amazon Athena** service:
   * Click **Get Started** if this is our first trip into Athena, otherwise continue to #2
2. First, add an S3 location for your queries by clicking on the '*Before you run your first query, you need to set up a query result location in Amazon S3.*' link
3. Paste in the S3 Bucket ARN we copied earlier, being sure to remove "arn:aws:s3:::" from the beginning of the data we paste in and including a trailling slash
4. Once the S3 location is properly configured you will notice the **Run query** button has been made active.
5. In the query editor paste the following query, then press Ctrl+Enter to run the query:

CREATE database aws\_service\_logs

1. Under *Tables*, select **Create Table** > **from S3 bucket data**.
2. Step 1: Name and Location:
   * Database: aws\_service\_logs
   * Table: cf\_access\_optimized
   * Location: s3://Name of the generated S3 bucket/ (including trailing slash)
3. Step 2: Data Format
   * Select **Parquet**
4. Step 3: Columns
   * Bulk add columns using this data:

time timestamp, location string, bytes bigint, requestip string, method string, host string, uri string, status int, referrer string, useragent string, querystring string, cookie string, resulttype string, requestid string, hostheader string, requestprotocol string, requestbytes bigint, timetaken double, xforwardedfor string, sslprotocol string, sslcipher string, responseresulttype string, httpversion string

1. Step 4: Partitions
   * Column Name: year, Column Type: string
   * Column Name: month, Column Type: string
   * Column Name: day, Column Type: string
   * Click **Create table**
2. Click **Run query** on the generated SQL statement. Ensure the S3 bucket location in the query matches the one generated in your lab environment.

check\_circle**Add Partition Metadata**

1. Open a new query tab
2. Run the following query: MSCK REPAIR TABLE aws\_service\_logs.cf\_access\_optimized
3. Verify the partitions were created with the following query: SELECT count(\*) AS rowcount FROM aws\_service\_logs.cf\_access\_optimized. You should see 207535 rows present in the table.
4. Run the following query: SELECT \* FROM aws\_service\_logs.cf\_access\_optimized LIMIT 10

check\_circle**Query the Total Bytes Served in a Date Range**

1. Perform the following query:
2. SELECT SUM(bytes) AS total\_bytes
3. FROM aws\_service\_logs.cf\_access\_optimized

WHERE time BETWEEN TIMESTAMP '2018-11-02' AND TIMESTAMP '2018-11-03'

1. Observe the value for total\_bytes equals 87310409.

# Querying Data in S3 with Amazon Athena

## Introduction

Welcome to this hands-on AWS lab for querying data in Amazon S3 with Amazon Athena. This lab allows you to practice analyzing data stored in S3 using SQL queries in Athena.

## Solution

Begin by logging in to the AWS Management Console using the credentials provided on the hands-on lab page.

### **Gather S3 Bucket's ARN for later use**

1. Navigate to the **S3** service
2. Check the box next to the bucket we've got in there
3. In the overlay that slides in (which contains details about the bucket) click **Copy Bucket ARN**

### **Create a Table from S3 Bucket Metadata**

1. Navigate to the **Amazon Athena** service:
   * Click **Get Started** if this is our first trip into Athena, otherwise continue to #2
2. First, add an S3 location for your queries by clicking on the 'Before you run your first query, you need to set up a query result location in Amazon S3.' link
3. Paste in the S3 Bucket ARN we copied earlier, being sure to remove "arn:aws:s3:::" from the beginning of the data we paste in and including a trailling slash
4. Once the S3 location is properly configured you will notice the **Run query** button has been made active.
5. In the query editor paste the following query, then press Ctrl+Enter to run the query:

CREATE database aws\_service\_logs

1. Under Tables, select **Create Table** > **from S3 bucket data**.
2. Step 1: Name and Location:
   * Database: aws\_service\_logs
   * Table: cf\_access\_optimized
   * Location: s3://Name of the generated S3 bucket/ (including trailing slash)
   * Paste in the S3 Bucket ARN we copied earlier, being sure to remove "arn:aws:s3:::" from the beginning of the data we paste in
3. Step 2: Data Format
   * Select **Parquet**
4. Step 3: Columns
   * Bulk add columns using this data:

time timestamp, location string, bytes bigint, requestip string, method string, host string, uri string, status int, referrer string, useragent string, querystring string, cookie string, resulttype string, requestid string, hostheader string, requestprotocol string, requestbytes bigint, timetaken double, xforwardedfor string, sslprotocol string, sslcipher string, responseresulttype string, httpversion string

1. Step 4: Partitions
   * Column Name: year, Column Type: string
   * Column Name: month, Column Type: string
   * Column Name: day, Column Type: string
   * Click **Create table**
2. Click **Run query** on the generated SQL statement. Ensure the S3 bucket location in the query matches the one generated in your lab environment.

### **Add Partition Metadata**

1. Open a new query tab
2. Run the following query: MSCK REPAIR TABLE aws\_service\_logs.cf\_access\_optimized
3. Verify the partitions were created with the following query:

SELECT count(\*) AS rowcount FROM aws\_service\_logs.cf\_access\_optimized

You should see 207535 rows present in the table.

1. To look at a bit of actual data from this table (just ten rows' worth) we can run this query:

SELECT \* FROM aws\_service\_logs.cf\_access\_optimized LIMIT 10

### **Query the Total Bytes Served in a Date Range**

1. Add another query tab, then let's look at the timestamps on our newest and oldest data. Run these two queries:
2. SELECT \* FROM aws\_service\_logs.cf\_access\_optimized ORDER BY time DESC LIMIT 10

SELECT \* FROM aws\_service\_logs.cf\_access\_optimized ORDER BY time ASC LIMIT 10

Our newest timestamp is from **2018-11-07**, and the oldest is from **2018-11-02**.

1. Now let's look at a sum of the bytes column for data between **11-02** and **11-03**:
2. SELECT SUM(bytes) AS total\_bytes
3. FROM aws\_service\_logs.cf\_access\_optimized

WHERE time BETWEEN TIMESTAMP '2018-11-02' AND TIMESTAMP '2018-11-03'

1. Observe the value for total\_bytes equals 87310409.

## Conclusion

Congratulations — you've completed this hands-on lab!

# **Performing a Backup and Restore Using AMI and EBS**

Additional Information and Resources

Please log in to the live environment with the cloud\_user credentials provided.

Make sure you are using the us-east-1 region throughout the lab.

Learning Objectives

check\_circle**Create an EBS Snapshot**

Create a snapshot from a web application server's EBS volume.

1. Navigate to **EC2** > **Instances**.
2. Check the box beside one of the "webserver-instance" instances.
3. Click the link to the **Root device**, and then the EBS volume link.
4. On the Volumes page, click **Actions**, and then **Create Snapshot**.
5. Enter a description.
6. Click **Create Snapshot**.

check\_circle**Create a New EBS Volume from a Snapshot**

Create a new EBS volume from the snapshot you just created in the previous step. This is how you restore an EBS volume from a snapshot.

1. Navigate to the **Snapshots** page.
2. Check the box beside the snapshot you just created.
3. Click **Actions**, and then **Create Volume**.
4. Change the volume size to 10 GiB.
5. Click **Create Volume**.
6. Click **Close**.

check\_circle**Create Two EC2 AMIs**

Amazon Machine Images can be created via several different methods. Create two AMIs following the methods detailed in the video.

### **Method 1**

1. Navigate to the **Instances** page.
2. Check the box beside one of the **webserver-instance** instances.
3. Click **Actions**, then **Image**, and then **Create Image**.
4. Add a name and description.
5. Click **Create Image**, and then **Close**.
6. Choose **AMIs** from the left menu to see the image you created.
7. Once the image is available, navigate to the instances page.
8. Click **Launch Instance**.
9. Choose **My AMIs** in the left column.
10. Click **Select**, and then you can configure all the other options for a new instance.
11. For now, click **Cancel**.

### **Method 2**

1. Navigate to **Snapshots**.
2. Check the box beside the snapshot you made earlier.
3. Click **Actions**, and select **Create Image**.
4. Enter a name and description.
5. Change Virtualization type to **Hardware-assisted virtualization**.
6. Click **Create**.
7. Click **Close**.
8. Choose **AMIs** from the left menu to see the image you created.

# Performing a Backup and Restore Using AMI and EBS

## Introduction

In this hands-on lab, we will perform a backup and restore using AMIs and EBS. This activity will explore several common backup and restore methods for the Amazon EC2 service.

## Solution

Log in to the live AWS environment using the credentials provided. Make sure you're in the N. Virginia (us-east-1) region throughout the lab.

### **Create an EBS Snapshot**

1. Navigate to **EC2** > **Instances**.
2. Check the box beside one of the "webserver-instance" instances.
3. Click the Root device link in the Description section below, and then click the EBS ID link.
4. On the Volumes page, make sure the listed volume is selected, and then click **Actions** and select **Create Snapshot**.
5. Enter a description (e.g., "WordPressSnap").
6. Click **Create Snapshot**.

### **Create a New EBS Volume from a Snapshot**

1. Navigate to **Snapshots**.
2. Check the box beside the snapshot you just created.
3. Click **Actions**, and select **Create Volume**.
4. Change the volume size to 10 GiB.
5. Click **Create Volume**.
6. Click **Close**.
7. Select the newly created volume, and then click **Actions** and select **Attach Volume**.
8. In the Instance dropdown, select the **webserver-instance** and click **Attach**.

### **Create Two EC2 AMIs**

#### **Method 1**

1. Navigate to **Instances**.
2. Check the box beside one of the "webserver-instance" instances.
3. Click **Actions**, then **Image**, and then **Create Image**.
4. Add a name and description (e.g., both could be "AMI1").
5. Click **Create Image**, and then **Close**.
6. Choose **AMIs** from the left menu to see the image you created.
7. Once the image is available, navigate to the instances page.
8. Click **Launch Instance**.
9. Choose **My AMIs** in the left column.
10. Click **Select**.
11. Leave t2.micro selected, and click **Next: Configure Instance Details**.
12. On the Configure Instance Details page:
    * Network: Leave default
    * Subnet: **AppLayer1private**
    * Auto-assign Public IP: **Disable**
13. Click **Next: Add Storage**, then **Next: Add Tags**, and then **Next: Configure Security Group**.
14. Click to **Select an existing security group**.
15. Select the **WebServerSecurityGroup** one from the table.
16. Click **Review and Launch**, and then **Launch**.
17. In the key pair dialog, select **Proceed without a key pair**.
18. Click **Launch Instances**, and then **View Instances**.

#### **Method 2**

1. Navigate to **Snapshots**.
2. Check the box beside the snapshot you made earlier.
3. Click **Actions**, and select **Create Image**.
4. Enter a name and description (e.g., both could be "AMI2").
5. Change Virtualization type to **Hardware-assisted virtualization**.
6. Click **Create**, and then **Close**.
7. Choose **AMIs** from the left menu to see the image you created.

## Conclusion

Congratulations on completing this hands-on lab!

## Lecture: VPC Essentials

Virtual Private Cloud allows us to create networks for our applications to run on. We can customize many different features, such as IP address range, how many layers our application needs, routing, security, and many more. This lesson discusses the basics of VPCs, the building blocks, and the attributes of a default VPC.

## Lecture: VPC Flow Logs

VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data can be published to Amazon CloudWatch Logs and Amazon S3. After you've created a flow log, you can retrieve and view its data in the chosen destination.

## Lecture: VPC Peering

VPC peering is a tool we can use when we want our VPCs to communicate using private IP addresses as if they are on the same network. We also have the newly released ability to peer VPCs across regions.

**UPDATE**: Amazon Virtual Private Cloud (VPC) now supports IPv6 traffic over Inter-Region VPC Peering. With this launch, your resources in different AWS regions, can communicate with each other using IPv6 addresses without requiring gateways, VPN connections, or separate physical hardware. Inter-Region VPC Peering encrypts inter-region traffic with no single point of failure or bandwidth bottleneck. Traffic using Inter-Region VPC Peering always stays on the AWS global network and never traverses the public internet. Read more: <https://aws.amazon.com/about-aws/whats-new/2019/11/inter-region-vpc-peering-supports-ipv6-traffic/>

## Lecture: AWS VPN

AWS Virtual Private Network (VPN) is a way to get secured communications to AWS from an on-premises environment using the public internet. This lesson discusses the basics and components of an AWS VPN connection.

## Lecture: AWS Direct Connect

AWS Direct Connect is a great solution for organizations needing a private connection to AWS with dedicated throughput. It also gives us virtual interfaces that can route to AWS services in different ways. This lesson discusses the components and details a SysOps Administrator needs to know about Direct Connect.

Direct Connect Partners: <https://aws.amazon.com/directconnect/partners/>

## Lecture: EC2: Elastic IP (EIP) and Elastic Network Interfaces (ENI)

Elastic IP (EIP) and Elastic Network Interfaces (ENI) give us a flexible way to retain the same IP address on an instance. This ability is useful in many application scenarios. With ENI, settings such as IP addresses and security groups migrate with the interface. Instances must still be in public subnets to be accessible with either. This lesson shows how to use each and some of the behaviors we need to be aware of.

## Lecture: ELB: Monitoring for Performance and Availability

There are three types of load balancers in AWS. The use cases and behaviors need to be known for each to properly deploy and administer load balancers in an AWS environment. This lesson discusses the differences and shows configuration examples for each type.

## Lecture: Amazon ELB: High Availability

Load balancers are responsible for serving traffic to multiple instances in an application. In addition, they can also prevent poor application performance by evaluating the health of the instances it serves. Traffic can then be directed to only those instances that are "healthy." This lesson discusses how Elastic Load Balancing can assist in scalable, highly available applications.

## Lecture: ELB: SSL Offloading

Using SSL for secure web communications can increase the processing your application servers are required to do. It might even start to affect your application's performance. Offloading the SSL handshake and decryption duties to a load balancer is a great way to alleviate this. We can also use AWS Certificate Manager to create and renew our SSL certificates. This lesson discusses the process of offloading the SSL workload.

#### **Helpful Links**

* [Create an HTTPS Listener for Your Application Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/create-https-listener.html)
* [Using SNI with ALB](https://aws.amazon.com/blogs/aws/new-application-load-balancer-sni/#ALB)

## Lecture: Network Bottlenecks

Network performance is highly important to any application. It is also the first culprit to be blamed when an application is not performing efficiently. Systems Operators need to know the most common causes of network bottlenecks in AWS. This lesson discusses several of those causes.

## Lecture: Amazon CloudFront

CloudFront is the AWS content delivery network (CDN). It allows us to cache our web applications around the world to provide lower latency and a better experience for our end users. It is important to know how our content is distributed to the edge locations and what happens when the edge location does not have a cached version of our content. This lesson discusses the basics of how to configure a CloudFront distribution and processes we need to be aware of.

#### **Helpful Links**

* [AWS Global Cloud Infrastructure](https://infrastructure.aws/)
* [How CloudFront Delivers Content](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/HowCloudFrontWorks.html)

# **Troubleshooting Elastic Load Balancing Connectivity in AWS**

Additional Information and Resources

Please make sure you are working in the us-east-1 (N. Virginia) region.

Goal: Fix the connectivity issue in the AWS environment so that you can view the Linux 2/Apache test page of the provisioned EC2 instances via the ELB's DNS name.

Learning Objectives

check\_circle**Hint #1**

#### **The Issue**

The ELB's security group does not allow HTTP traffic.

#### **How to Fix the Issue**

Add an allow rule for HTTP traffic (port 80) to the ELB's security group.

check\_circle**Hint #2**

#### **The Issue**

EC2 instance health checks are not passing.

#### **How to Fix the Issue**

Change health check "ping port" on the ELB to port 80.

# Troubleshooting Elastic Load Balancing Connectivity in AWS

## Introduction

In this hands-on lab, we're going to fix the connectivity issue in the AWS environment so we can view the Linux 2/Apache test page of the provisioned EC2 instances via the ELB's DNS name.

## Solution

Log in to the live AWS environment using the credentials provided. Make sure you're in the N. Virginia (us-east-1) region throughout the lab.

### **Hint #1**

#### **The Issue**

The ELB's security group does not allow HTTP traffic.

#### **How to Fix the Issue**

Add an allow rule for HTTP traffic (port 80) to the ELB's security group.

1. Navigate to **EC2** > **Load Balancers**.
2. With the load balancer selected, scroll to the Security section in the Description section, and click the listed security group.
3. With the security group selected, scroll to the Inbound section, and click **Edit**.
4. Change the Type to **HTTP**, and click **Save**.
5. Click **Load Balancers** in the left-hand menu.
6. Copy the load balancer's DNS name in the Description section, and paste it into a new browser tab. The page won't be able to load.

### **Hint #2**

#### **The Issue**

EC2 instance health checks are not passing.

#### **How to Fix the Issue**

Change health check "ping port" on the ELB to port 80.

1. Back in the AWS console, click the Instances tab on the load balancers page, where we'll see the instances associated with our load balancer are marked as OutOfService.
2. In the Health check tab, click **Edit Health Check**.
3. In the dialog, change Ping Port to **80**, and click **Save**.
4. Back in the Instances tab, we should see they're now listed as InService.
5. Reload the load balancer DNS in the browser, which should now display the Linux 2/Apache test page.

## Conclusion

Congratulations on completing this hands-on lab!

## Lecture: Amazon RDS: Understanding Multi-AZ Deployments

Multi-AZ is a fault-tolerant feature in Relational Database Service (RDS). It prevents an Availability Zone failure from removing database access from an application. This lesson discusses RDS Multi-AZ deployments and how they handle fault tolerance. There is also a simulated failover executed.

## Lecture: RDS: Monitoring for Performance and Availability

The performance of your RDS instances can affect your application greatly. We can use these metrics to make decisions about instance types and read replicas. This lesson discusses the Relational Database Service (RDS) and how to monitor performance.

## Lecture: Amazon ElastiCache: Monitoring for Performance and Availability

Amazon ElastiCache is a web service that makes it easy to set up, manage, and scale a distributed in-memory cache environment in the cloud. It provides a high-performance, scalable, and cost-effective caching solution while removing the complexity associated with deploying and managing a distributed cache environment. ElastiCache is ideal for storing web application session state data. It can also offload the read traffic from a database to reduce latency caused by a read-heavy workload.

#### **Helpful Links**

* [Performance at Scale with ElastiCache](https://d0.awsstatic.com/whitepapers/performance-at-scale-with-amazon-elasticache.pdf)

## Lecture: Amazon DynamoDB Concepts

DynamoDB is a nonrelational database (often referred to as "NoSQL") that delivers performance at any scale. It's a fully managed, multi-region, multi-master database service that adapts to your throughput needs and offers built-in security, backups and restores, and in-memory caching. It's ideal for mobile, web, gaming, ad tech, IoT, and other applications that need consistent, low-latency data access.

#### **Helpful Links**

* [NoSQL Design for DynamoDB](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-general-nosql-design.html)
* [From SQL to NoSQL](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/SQLtoNoSQL.html)
* [Partitions and Data Distribution](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.Partitions.html)
* [DynamoDB Transactions](https://aws.amazon.com/blogs/aws/new-amazon-dynamodb-transactions/)
* [Dynamo: Amazon's Highly Available Key-value Store](http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf)
* [Sample movies data and scripts used in this lesson](https://github.com/linuxacademy/content-aws-soa-c01/tree/master/DynamoDB)

## Lecture: Amazon Redshift

The Amazon Redshift service manages all of the work of setting up, operating, and scaling a data warehouse. These tasks include provisioning capacity, monitoring and backing up the cluster, and applying patches and upgrades to the Redshift engine.

For more information on the Amazon Data Lake - Project Name "Andes":

[AWS re:Invent 2017: A Look Under the Hood – How Amazon.com Uses AWS Services for Analytics at Massive Scale (ABD329)](https://www.youtube.com/watch?v=lWK96BNx3Qk)

## Lecture: Amazon Aurora

Amazon Aurora is a fully managed database engine, compatible with MySQL and PostgreSQL. The code, tools, and applications you use today with your existing MySQL and PostgreSQL databases can be used with Aurora. With some workloads, Aurora can deliver up to 5X the throughput of MySQL and up to 3X the throughput of PostgreSQL without requiring changes to most of your existing applications.

#### **Helpful Links**

* [Amazon Aurora FAQs](https://aws.amazon.com/rds/aurora/faqs/)

# **Restoring an Amazon RDS Instance Using Snapshots**

Additional Information and Resources

In this scenario, you are a systems administrator for a company with a corporate website powered by WordPress. The developer has just notified you that some unspecified corruption in the database has occurred, and now the site is down and the admin panel cannot be reached.

Your task is to use the point-in-time restore capability of RDS automated snapshots to restore the database and bring the site back up.

Please log in to the lab environment with the cloud\_user credentials provided. Make sure you are using us-east-1 region throughout the lab.

Make a note of the cloud\_user password for connecting to the bastion instance.

**NOTE**: This environment will take up to 15 minutes to create.

Learning Objectives

check\_circle**Create an RDS Snapshot**

Create a manual snapshot of the existing RDS database in this environment.

1. In the RDS console, choose the running instance.
2. Select **Maintenance & backups** > **Take snapshot**.
3. Name the snapshot (e.g., "wordpress-YYYYMMDD").

check\_circle**Log in to the Bastion Host and Delete Database Data**

Log in to the bastion host via SSH, and, using the mysql command line, drop a critical table from the database.

1. In the EC2 console, find the public IP of the bastion host.
2. Log in to the bastion host via SSH as cloud\_user using the public IP.
3. Connect to MySQL:

mysql --user=wpuser --password=Password1 --host=<RDS ENDPOINT NAME>

1. At the MySQL prompt, switch to the WordPress database:

use wordpressdb;

1. Delete a critical table:

drop table wp\_posts;

check\_circle**Restore an RDS Database from a Snapshot**

Use the snapshot you created to create a new database with all the data from before it was corrupted.

1. In the RDS console, navigate to snapshots.
2. Note the snapshot creation time.
3. Navigate to *Databases*, and select **wordpress-database**
4. Under *Actions*, select **Restore to point in time**.
5. Select a custom restore time, and enter the point in time you want to restore from.
6. Name the recovered database instance "ordpress-recovery"
7. Select availability zone **-east-1b**
8. Launch the database instance.

check\_circle**Rename Database Instances**

Rename the wordpress-recovery instance to wordpress-database, and rename wordpress-database to something else.

1. In the RDS console, select the **wordpress-database** instance, and click **Modify**.
2. Change the DB instance identifier from "wordpress-database" to "wordpress-corrupt".
3. Click **Continue**.
4. Select **Apply Immediately**.
5. Click **Modify DB Instance**.
6. Follow the above steps to rename "wordpress-recovery" to "wordpress-database".
7. Select the database security group.
8. Click **Continue**.
9. Select **Apply Immediately**.
10. Click **Modify DB Instance**.
11. Visit the website and observe that the last known good configuration is present.

# Restoring an Amazon RDS Instance Using Snapshots

## Introduction

A critical part of database management is being able to recover your data after corruption or accidental deletion has occurred.

In this hands-on lab, we'll use the point-in-time restore capability of RDS automated snapshots to restore a database and bring a site back up.

## Solution

Please log in to the lab environment with the cloud\_user credentials provided. Make sure you are using us-east-1 region throughout the lab.

Make a note of the cloud\_user password for connecting to the bastion instance.

**NOTE**: This environment will take up to 15 minutes to create.

### **Open the WordPress Application to Monitor It**

1. Navigate to **EC2**.
2. Click the **Load Balancers** button.
3. Under Basic Configuration, copy the DNS name listed below, and paste it into a new browser tab.
4. Keep this tab open to monitor the application later.

### **Create an RDS Snapshot**

1. In the AWS console, navigate to **RDS**.
2. Click **DB Instances**
3. Select the running instance.
4. Select the **Maintenance & backups** tab.
5. Click **Take snapshot**.
6. Name the snapshot (e.g., "wordpress-YYYYMMDD").
7. Click **Take Snapshot**.

### **Log in to the Bastion Host and Delete Database Data**

1. Navigate to **EC2**.
2. Click **Running Instances**.
3. Select the **bastion-host** instance.
4. Copy the public IP listed in the Description section.
5. Open a terminal session, and log in to the bastion host via SSH as cloud\_user using the public IP:

ssh cloud\_user@<PUBLIC IP>

Use the password provided on the lab page — it will be different than the one you used to log in to the AWS console.

1. Install the MySQL command line:

sudo yum install mysql

1. When prompted, enter the cloud\_user password.
2. In a different tab, navigate back to the Amazon RDS console.
3. In the AWS console, navigate to **RDS**.
4. Click **DB Instances**
5. Select **wordpress-database**.
6. Select the **Connectivity & security** tab.
7. Copy the endpoint listed and paste it into a text file, as we'll need it for a later step to use for the RDS endpoint name.
8. Return to the terminal.
9. Connect to MySQL, pasting in the RDS endpoint you copied earlier:

mysql --user=wpuser --password=Password1 --host=<RDS ENDPOINT NAME>

1. At the MySQL prompt, switch to the WordPress database:

use wordpressdb;

1. Delete a critical table:

drop table wp\_posts;

1. Return to the browser.
2. Refresh the WordPress site, which should result in an error page.

### **Restore an RDS Database from a Snapshot**

1. In the RDS console, navigate to **Snapshots**.
2. Note the snapshot creation time.
3. Click **Databases** in the left-hand menu.
4. Select **wordpress-database**.
5. Click **Actions**, and select **Restore to point in time**.
6. Select a **Custom** restore time, and enter the point in time you want to restore from (selecting today's date and a time after the snapshot creation time).
7. Give it a DB instance identifier of "wordpress-recovery".
8. Set the Availability zone to **us-east-1b**.
9. Click **Launch DB Instance**. It will take a few minutes for it to become available.

### **Rename Database Instances**

1. Head back to the Databases dashboard.
2. Select **wordpress-database**, and click **Modify**.
3. Change the DB instance identifier from "wordpress-database" to "wordpress-corrupt".
4. Click **Continue**.
5. Select **Apply Immediately**, and click **Modify DB Instance**. It will take a minute for it to finish renaming.
6. Now, select **wordpress-recovery**, and click **Modify**.
7. Change the DB instance identifier from "wordpress-recovery" to "wordpress-database".
8. Under Security group, delete the default security group and select **DatabaseSecurityGroup** from the dropdown.
9. Click **Continue**.
10. Select **Apply Immediately**, and click **Modify DB Instance**.
11. Refresh the WordPress website, and observe that the last known good configuration is present.

## Conclusion

Congratulations on completing this hands-on lab!

# **Deploying an Amazon RDS Multi-AZ and Read Replica**

Additional Information and Resources

Please go ahead and log in to the live environment with the cloud\_user credentials provided.

Make sure you are using us-east-1 as your region throughout the lab.

Learning Objectives

check\_circle**Enable Multi-AZ Deployment**

1. Navigate to **EC2** > **Load Balancers**.
2. Copy the DNS name of the load balancer.
3. Open a new browser tab, and enter the DNS name. We will use this web page to test failovers and promotions in this lab.
4. Navigate to **RDS** > **Databases**.
5. Click on our database instance.
6. Click **Modify**.
7. Click **Yes** for Multi-AZ deployment.
8. Change Backup Retention to **1 day**, needed later for read replicas.
9. Click **Continue**.
10. Click **Apply immediately**, and then **Modify DB Instance**.
11. Once the instance shows Multi-AZ is enabled, reboot the primary.
12. Use the web page to monitor the outage (normally about 30 seconds).
13. The Multi-AZ standby is now the primary.

check\_circle**Create a Read Replica**

1. Check the circle beside the database instance.
2. Click **Actions**.
3. Choose **Create read replica**.
4. Choose **US East (N. Virginia)** for destination region.
5. Enter a name.
6. Choose **Create read replica**.

check\_circle**Promote the Read Replica and Change the CNAME Records Set in Route 53 to the New Endpoint**

1. Once the read replica is available, check the circle next to it.
2. Click **Actions**.
3. Choose **Promote**.
4. Click **Continue**, and then **Promote Read Replica**.
5. Use the web page to monitor for downtime.
6. Once the read replica is available, copy the endpoint.
7. Open Route 53 in a new tab.
8. Edit the database.sysopsdatabase.com record to use the read replica endpoint.
9. Monitor using the web page for downtime.

# Deploying an Amazon RDS Multi-AZ and Read Replica

## Introduction

Welcome to this live hands-on AWS lab, where we will be working with the Relational Database Service (RDS).

This lab will provide you with hands-on experience with:

* Enabling Multi-AZ and Backups
* Creating a read replica
* Promoting a read replica
* Updating the RDS endpoint in Route53

## Solution

Log in to the live AWS environment using the credentials provided. Make sure you're in the N. Virginia (us-east-1) region throughout the lab.

### **Enable Multi-AZ Deployment**

1. Navigate to **EC2** > **Load Balancers**.
2. Copy the DNS name of the load balancer.
3. Open a new browser tab, and enter the DNS name. We will use this web page to test failovers and promotions in this lab.
4. Back in the AWS console, navigate to **RDS** > **Databases**.
5. Click on our database instance.
6. Click **Modify**.
7. Under Multi-AZ deployment, click **Yes**.
8. Change Backup Retention to **1 day**, needed later for read replicas.
9. Click **Continue**.
10. Under Scheduling of modifications, select **Apply immediately**, and then click **Modify DB Instance**.
11. Once the instance shows Multi-AZ is enabled (it could take about 10 minutes), select the database instance.
12. Click **Actions**, and select **Reboot**.
13. On the reboot page, select **Reboot With Failover?**, and click **Reboot**.
14. Use the web page to monitor the outage (normally about 30 seconds).
15. The Multi-AZ standby is now the primary.

### **Create a Read Replica**

1. With the database instance still selected, click **Actions**, and select **Create read replica**.
2. For Destination region, select **US East (N. Virginia)**.
3. Enter a name under DB instance identifier (e.g., "wordpress-rr").
4. Leave the other defaults, and click **Create read replica**. It will take a few minutes for it to become available.

### **Promote the Read Replica and Change the CNAME Records Set in Route 53 to the New Endpoint**

1. Once the read replica is available, check the circle next to it.
2. Click **Actions**, and select **Promote**.
3. Leave the defaults, and click **Continue**, and then click **Promote Read Replica**.
4. Use the web page to monitor for downtime.
5. Once the read replica is available, click to open it.
6. In the Connectivity & security section, copy the endpoint under Endpoint & port.
7. Open Route 53 in a new tab.
8. Click **Hosted zones**, and select the **sysopsdatabase** hosted zone.
9. Click **Go to Record Sets**.
10. Click the **CNAME** row.
11. Replace what's currently in the Value box with the endpoint you copied.
12. Click **Save Record Set**.
13. Monitor using the web page for downtime.

## Conclusion

Congratulations on completing this hands-on lab!

## Lecture: AWS Elastic Beanstalk

AWS Elastic Beanstalk enables you to deploy and manage applications without worrying about the infrastructure that runs those applications. Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

In this video, you will learn about the AWS Elastic Beanstalk service. Specifically, you'll see how you can use Elastic Beanstalk to simplify the deployment of your applications on AWS by allowing Elastic Beanstalk to manage the deployment of many different AWS services for you. By the end of the video, you should understand the basic purpose of Elastic Beanstalk, as well as how to deploy a sample application using the service.

## Lecture: Amazon Elastic Container Service (ECS)

Amazon Elastic Container Service (ECS) is a highly scalable, high-performance container management service that makes it easy to run, stop, and manage Docker containers on a cluster of EC2 instances or using serverless technology with AWS Fargate.

## Lecture: AWS Systems Manager

AWS Systems Manager is a management service that helps you automatically collect software inventory, apply operating system patches, create system images, and configure Windows and Linux operating systems. Systems Manager can be used for both EC2 instances, on-premises servers, and VMs. These capabilities help you define and track system configurations, prevent drift, and maintain software compliance of your EC2 and on-premises configurations.

## Lecture: AWS Systems Manager

AWS Systems Manager is a management service that helps you automatically collect software inventory, apply operating system patches, create system images, and configure Windows and Linux operating systems. Systems Manager can be used for both EC2 instances, on-premises servers, and VMs. These capabilities help you define and track system configurations, prevent drift, and maintain software compliance of your EC2 and on-premises configurations.

## Lecture: AWS OpsWorks

OpsWorks is a service that uses Chef cookbooks developed in the Ruby language. It allows us to manage our application in layers. We can use recipes to affect our layers at various lifecycle events in an application's deployment. This video walks through the deployment of a sample Node.js application to further understand the OpsWorks infrastructure.

#### **Helpful Links**

* [OpsWorks FAQ](https://aws.amazon.com/opsworks/stacks/faqs/)

## Lecture: Disaster Recovery

There are many different decisions administrators must make when it comes to disaster recovery planning. A DR solution should be cost-effective so that it does not cost the organization more than it would lose during an outage. Cost, downtime, and frequent testing are important decisions in any company. This lesson discusses disaster recovery scenarios as well as the cost versus downtime planning decisions that need to be made.

# **Deploying a Highly Available Web Application and a Bastion Host in AWS**

Additional Information and Resources

Make sure you are using N. Virginia (us-east-1) as your region throughout the lab.

Database Snapshot ARN: arn:aws:rds:us-east-1:892710030684:snapshot:sysops-certification-la-course

* Database Name: wordpressdb (not wordpress)
* Username: wpuser
* Password: Password1
* Database Host: **Use the RDS endpoint name**
* Table prefix: wp\_

The user data script for the web server launch configuration is [here](https://github.com/linuxacademy/content-aws-soa-c01/blob/master/wordpress-userdata.sh).

NOTE: When selecting the AMI for your bastion host and instance for your web app, please use the **AMI ID** of the Amazon Linux 2 AMI instance.

Learning Objectives

check\_circle**Launch an RDS Database**

1. Launch an RDS database using the provided RDS backup image (containing the WordPress site data).

check\_circle**Create Security Groups**

1. Navigate to VPC.
2. Create four security groups with inbound rules.

check\_circle**Create Launch Configurations and Auto Scaling Groups**

1. Create two Auto Scaling groups - bastion host and application servers.

check\_circle**Modify Database Security Groups and Create an Application Load Balancer**

1. Create an application load balancer to distribute traffic to the application servers.

check\_circle**Browse Web Application**

1. Navigate to the RDS console.
2. Note the endpoint name.
3. Navigate to **Load Balancers** for the DNS name, and browse to it.
4. Configure WordPress:
   * Database Name: **wordpressdb** (not wordpress)
   * Username: **wpuser**
   * Password: **Password1**
   * Database Host: Use the RDS endpoint name
   * Table prefix: **wp\_**

# Deploy a Basic Infrastructure Using CloudFormation Templates

## Introduction

In this hands-on lab, we are going to build a highly available web application, along with a highly available bastion host architecture.

## Solution

Log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 region.

### **Launch an RDS Database**

1. From the AWS Management Console, navigate to the **VPC** dashboard.
2. In the left-hand menu, click **Your VPCs**. Review the lab-provided VPC.
3. In the left-hand menu, click **Subnets**. Review the lab-provided subnets.
4. In the top-left corner of the page, using the Services dropdown, navigate to the **RDS** dashboard.
5. In the left-hand menu, click **Subnet groups**.
6. Click **Create DB Subnet Group**.
7. Give the Subnet group a name and description, and select the lab-provided VPC from the list.
8. Under Add subnets, select the us-east-1a availability zone, and the subnet that uses the 10.99.21.0/24 CIDR block.
9. Under Add subnets, select the us-east-1b availability zone, and the subnet that uses the 10.99.22.0/24 CIDR block.
10. Click **Create**.
11. In the left-hand menu, click **Snapshots**.
12. Switch to the Public tab and search for the arn provided in the lab credentials.
13. Select the wordpress-database snapshot and in the Actions dropdown menu, select **Restore snapshot**.
14. Give the database an **instance identifier** and verify that the VPC and Subnet group have been pre-populated. If not, select the lab-provided VPC and the user-created Subnet group.
15. For the DB instance class, select a burstable db.t3.micro instance.
16. Under **Availability & durability**, select **Create a standy instance**.
17. Leave the remaining settings as default and click **Restore DB instance**.

### **Create Security Groups**

1. Using the Services dropdown menu, navigate to the **VPC** dashboard.
2. In the left-hand menu, under **Security**, click **Security Groups**.
3. Click **Create security group**.
4. For the Basic Details, enter the following:
   * Security group name: **BastionSG**
   * Description: **BHSG**
5. Under Inbound rules, click **Add rule**.
6. For the Type, choose **SSH**, and for the source, select **Anywhere**.
7. Click **Create security group**.
8. Navigate back to the Security Groups list and click **Create security group**.
9. For the Basic Details, enter the following:
   * Security group name: **ALBSG**
   * Description: **ALBSG**
10. Under Inbound rules, click **Add rule**.
11. For the Type, choose **HTTP**, and for the source, select **Anywhere**.
12. Under Inbound rules, click **Add rule**.
13. For the Type, choose **HTTPS**, and for the source, select **Anywhere**.
14. Click **Create security group**.
15. Navigate back to the Security Groups list and click **Create security group**.
16. For the Basic Details, enter the following:
    * Security group name: **WebServerSG**
    * Description: **WebServerSG**
17. Under Inbound rules, click **Add rule**.
18. For the Type, choose **SSH**, and for the source, select **BastionSG**.
19. Under Inbound rules, click **Add rule**.
20. For the Type, choose **HTTP**, and for the source, select **ALBSG**.
21. Under Inbound rules, click **Add rule**.
22. For the Type, choose **HTTPS**, and for the source, select **ALBSG**.
23. Click **Create security group**.
24. Navigate back to the Security Groups list and click **Create security group**.
25. For the Basic Details, enter the following:
    * Security group name: **DBSG**
    * Description: **DBSG**
26. Under Inbound rules, click **Add rule**.
27. For the Type, choose **MYSQL/Aurora**, and for the source, select **WebserverSG**.
28. Click **Create security group**.

### **Create Launch Configurations and Auto Scaling Groups**

1. Using the Services dropdown menu, navigate to the **EC2** dashboard.
2. In the left-hand menu, under Auto Scaling, click **Auto Scaling Groups**.
3. Click **Create Auto Scaling group**.
4. In the Launch template box, click **Create a launch template**.
5. Enter the following information:
   * Launch template name: **BastionLC**
   * Template version description: **BastionLC**
   * Amazon machine image: **Amazon Linux 2 AMI**
   * Instance type: **t2.micro**
   * Key pair: **Create new key pair**
   * Security groups: **BastionSG**
6. Under Network interfaces, set **Auto-assign public IP** to **Enable**.
7. Click **Create launch template**.
8. Click **View launch templates** to monitor the resource.
9. Once created, select the launch template, and in the **Actions** dropdown, click **Create Auto Scaling group**.
10. For the name, enter "BHASG" and click **Next**.
11. Under Network, select the lab-provided VPC, and choose the DMZ1public and DMZ2public subnets from the list. Click **Next**.
12. Leave the Load balancing options as default and click **Next**.
13. Leave the Group size settings as default and click **Next**.
14. Click **Next**, **Next** again, and click **Create Auto Scaling group**.
15. Back in the Auto Scaling groups list, click **Create an Auto Scaling group**.
16. In the Launch template box, click **Create a launch template**.
17. Enter the following information:
    * Launch template name: **WSLC**
    * Template version description: **WSLC**
    * Amazon machine image: **Amazon Linux 2 AMI**
    * Instance type: **t2.micro**
    * Key pair: **Use the key pair created earlier**
    * Security groups: **WebServerSG**
18. Open the Advanced details options and in the **User data** box, paste the script found at the [following link](https://github.com/linuxacademy/content-aws-soa-c01/blob/master/wordpress-userdata.sh).
19. Click **Create launch template**.
20. Click **View launch templates**.
21. Once created, select the Web Server launch template, and in the **Actions** dropdown, click **Create Auto Scaling group**.
22. For the name, enter "WSASG" and click **Next**.
23. Under Network, select the lab-provided VPC, and choose the AppLayer1 and AppLayer2 subnets from the list. Click **Next**.
24. Leave the Load balancing options as default and click **Next**.
25. For the Group size settings, set the following:
    * Desired capacity: **2**
    * Minimum capacity: **2**
    * Maximum capacity: **2**
26. Click **Next** through the remaining options and click **Create Auto Scaling group**.
27. Using the Services dropdown, navigate back to the **EC2** dashboard, and click **Instances**. Verify the instances were created successfully.

### **Modify Database Security Groups and Create an Application Load Balancer**

1. Using the Services dropdown, navigate back to the **RDS** dashboard.
2. In the left-hand menu, click **Databases**.
3. Select the wordpress-database and click **Modify**.
4. Under Connectivity, change the Security group setting to **DBSG**. Delete the **default** security group.
5. Click **Continue**.
6. Under Scheduling of modifications, select **Apply immediately** and click **Modify DB instance**.
7. Head back to the **EC2** dashboard, and in the left-hand menu under **Load Balancing**, click **Load Balancers**.
8. Click **Create Load Balancer**.
9. Under Application Load Balancer, click **Create**.
10. For the Name, enter "ALB1", and under Listeners, add an HTTP listener on port 80.
11. Under Availability Zones, set the following values:
    * us-east-1a: **DMZ1public**
    * us-east-1b: **DMZ2public**
12. Click **Next: Configure Security Settings**.
13. Ignore the warning and click **Next: Configure Security Groups**.
14. Select the ALBSG security group from the list and click **Next: Configure Routing**.
15. For the Target group settings, set the following:
    * Name: **TG1**
    * Target type: **Instance**
    * Protocol: **HTTP**
    * Port: **80**
16. For the Health checks settings, set the following:
    * Protocol: **HTTP**
    * Path: **/readme.html**
17. Click **Next: Register Targets**.
18. Under Instances, choose the **WebServerSG** instances, and click **Add to registered**.
19. Click **Next: Review** and click **Create**.
20. Click **Close** and review the load balancer configuration.

### **Browse Web Application**

1. Under Basic Configuration, copy the DNS name.
2. In a separate tab, paste the DNS name and navigate to the web application.
3. Click **Let's go!**
4. Enter the following values:
   * Database Name: **wordpress-db**
   * Username: **wpuser**
   * Password: **Password1**
   * Database Host:
   * Table Prefix: **wp\_**

Note: The wordpress-database endpoint can be found in the database's Connectivity & security tab.

1. Click **Submit**

## Conclusion

Congratulations — you've completed this hands-on lab!

# **Deploying a Basic Infrastructure Using CloudFormation Templates**

Additional Information and Resources

Log in to the live AWS environment using the credentials provided. Make sure you are in the us-east-1 (N. Virginia) Region throughout the lab.

[AWS CloudFormation templates](https://github.com/awslabs/aws-cloudformation-templates/tree/master/aws/services)

Learning Objectives

check\_circle**Using CloudFormation Designer, Configure the InstanceType Stack Parameter to T3.Micro**

1. Navigate to the CloudFormation dashboard and select **Designer** from the left navigational pane.
2. In CloudFormation Designer, modify the InstanceType Parameter defined at the top of the template. Change the T3.Small default size to T3.Micro.

check\_circle**Launch the Updated Stack and Verify the New EC2 Resource Is Reachable**

1. After making the necessary changes in CloudFormation Designer, launch the stack with a default InstanceType Parameter of T3.Micro.
2. Verify the stack updates successfully and that you're able to reach the newly allocated T3.Micro instance by navigating to the IPv4 Public DNS.
3. After verifying the resource is responsive, you have successfully modified the template in this lab.

# Deploy a Basic Infrastructure Using CloudFormation Templates

## Introduction

Your development team has been using this template for the intern-testing program. After performing an analysis, your team has determined that the t3.small instance was more compute power than they needed. Update the template so that the default instance defined in the stack is a t3.micro.

In this hands-on lab, we're going to jump into an environment that already has a CloudFormation stack deployed. We'll review the contents of the CloudFormation template, and then we'll perform direct updates to the stack itself.

## Solution

Log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 region.

### **Using CloudFormation Designer, Configure the InstanceType Stack Parameter to T3.Micro**

1. In the search bar at the top, enter "CloudFormation".
2. Select **CloudFormation** from the dropdown menu.
3. Select the stack already in the CloudFormation dashboard.
4. Review the Stack info tab.
5. Review the Events tab (the newest events will appear first).
6. Review the Resources tab.
7. Right-click and open the physical ID for the InternetGateway in a new tab.
8. Navigate back to the CloudFormation dashboard.
9. Right-click and open the physical ID for the WebServerInstance in a new tab.
10. Close out the InternetGateway and WebServerInstance tabs after reviewing.
11. Review the Outputs tab.
12. Right-click on the PublicDNS link and open it in a new tab.
13. Navigate back to the CloudFormation dashboard.
14. Click on the Parameters tab.
15. Click **Update** in the top-right corner.
16. Ensure **Use current template** is selected and click **Next**.
17. Under InstanceType, select **t3.micro**.

**Note**: This only changes the current deployment of the stack.

1. Click **Cancel**.
2. Navigate back to the CloudFormation dashboard and click on the existing stack.
3. Click **Update** in the top-right corner.
4. Select **Edit template in designer** and click **View in Designer**.
5. Ensure YAML is selected for the template.
6. Under Parameters, change the default size from t3.small to t3.micro by erasing "small" and typing "micro".

### **Launch the Updated Stack and Verify the New EC2 Resource Is Reachable**

1. Copy the template and click on the checkmarked box icon in the top-left corner to validate the template. You should see Template is valid.
2. Click on the cloud icon next to the checkmarked box to launch the stack.

**Note**: This is a direct update to the existing stack.

1. Click **Next**.
2. Under Parameters, select the InstanceType dropdown box and select **t3.micro**.
3. Click **Next** through Tags.
4. Scroll down and select the acknowledgement under Capabiities.
5. Click **Update stack**.
6. Click on the refresh icon in the top-right corner of the CloudFormation dashboard to ensure the update is complete (this may take a few minutes).
7. Navigate to the Resources tab and open the WebServerInstance physical ID in a new tab to verify that the InstanceType is t3.micro.
8. Navigate back to the CloudFormation dashboard and select the Outputs tab.
9. Open the PublicDNS value in a new tab to ensure the instance is reachable via the public web.

## Conclusion

Congratulations — you've completed this hands-on lab!

# **Simple Disaster Recovery with CloudFormation and Lambda**

Additional Information and Resources

You’ve just taken over responsibility for an AWS workload consisting of an application (WordPress) running on EC2. There is currently no Disaster Recovery plan for this environment. In this lab, you will evaluate and test the use of a CloudFormation template to recreate the needed infrastructure in the event of a disaster. You will also further automate the process by adding a Lambda function that will look up current AMI ID numbers and provide them to the CloudFormation template.

Click [here](https://github.com/ACloudGuru-Resources/content-simple-dr-cf-lambda) to download lab files. (These files are stored in a GitHub repository; clicking on the Download button will allow you to download a ZIP file containing the necessary lab files. You'll need to open the ZIP file to extract the individual files before uploading them to S3.)

Learning Objectives

check\_circle**Review the Disaster Recovery Plan**

1. Review the basics of Disaster Recovery in AWS including:
   * Disaster recovery strategies
   * Recovery time objective (RTO) and recovery point objective (RPO)
2. Examine a demo cross-region DR Plan using Backup and Restore with CloudFormation.

check\_circle**Implement and Test the Disaster Recovery Plan**

1. Verify the **CF\_WordPress\_Blog** CloudFormation template is in your S3 bucket.
2. Launch the template in the Production region (us-east-1).
3. Review the new stack and the resources created.
4. Review the **Outputs** tab of the CloudFormation stack.

check\_circle**Add and Test the Lambda AMI Lookup Function**

1. Copy the code for the Lamdba AMI Lookup function **(amilookup.zip)** to an S3 bucket.
2. Copy the **1-CF\_WordPress\_Blog\_with\_Lookup** CloudFormation template to an S3 bucket.
3. Copy the **2-Add\_Single\_Instance** CloudFormation template to an S3 bucket.
4. Examine both CloudFormation templates:
   * Note the **Cross Stack Reference**
5. Launch the **1-CF\_WordPressBlog\_with\_Lookup** template in the Disaster Recovery region:
   * For this lab, also use us-east-1 for DR.
   * Examine the Outputs tab of the new stack.
6. Launch the **2-Add\_Single\_Instance** template:
   * Use the same region as in the previous step.
7. Examine the newly created EC2 instance:
   * Note which VPC is in use.

## Introduction

Recovering AWS workloads after a disaster is a crucial skill for organizations using the AWS cloud. This requires creating strategies and testing **Disaster Recovery (DR)** plans and techniques.

**CloudFormation** offers a simple solution that can help with DR over the long term, and helps with working toward implementing more advanced strategies.

One problem, however, is that **AMI ID numbers** can change over time, and vary from region to region. Adding a simple **Lambda function** to the strategy can automate and fix the issue.

#### **In This Lab You Will:**

1. Explore **disaster recovery strategies** including:
   * Backup and Restore
   * Pilot Light
   * Warm Standby
   * Multi-site
2. Use a CloudFormation template to **restore a production AWS environment**.
3. Examine a **Lambda function** and a **CloudFormation Cross Stack Reference** that can **automate** looking up current AMI ID numbers for launching resources in the correct VPC.
4. Use a modified **CloudFormation template** with **Lambda** to launch an additional EC2 instance.

## Solution

### **Log In**

Use the credentials provided on the hands-on lab overview page, and log into the AWS console as cloud\_user.

### **Create a Key Pair**

1. In the AWS console, navigate to EC2, and then to **Key Pairs** (down a bit in the left hand menu).
2. Once in there, click **Create key pair**.
3. Give it a name in the next screen (**DR\_Lab\_Keys** should work), and leave the File format set to **pem**.
4. Click on the **Create key pair** button. The browser should automatically download the file, and we'll just leave that for later in our downloads folder.

### **Download CloudFormation Templates**

Now we can navigate to S3 in the AWS Console. If we click into the one that the lab environment created for us, we'll see a couple of files in there already. They're both CloudFormation templates that do the same thing, but one is written in JSON and the other in YAML.

### **Upload Files to the S3 Bucket**

Remember the lab files we grabbed from GitHub? ([If you don't have the files yet, get them here](https://github.com/ACloudGuru-Resources/content-simple-dr-cf-lambda)) They may came down as a zip, so once they're extracted on our local machine, we're going to upload those to this S3 bucket.

1. In the same S3 window where we saw the JSON and YAML files, click the **Upload** button.
2. Click **Add files** on the next window.
3. Then navigate to where we extracted the GitHub zip file.
4. We want to upload 1-CF\_WordPress\_Blog\_with\_Lookup.json, 2-Add\_Single\_Instance.json, and amilookup.zip. So for each, click on it, then the **Open** button.
5. Then repeat the process until all three are up in the bucket.

### **Set up the Environment**

1. Click on CF\_Wordpress\_Blog.json (found in your lab provided S3 Bucket) and copy the Object URL at the bottom of the next screen. This file is the CloudFormation template we'll use to create the environment.
2. In a new browser tab, because we'll be swapping back and forth quite a bit, navigate to CloudFormation in the AWS console.
3. There is one stack in there already, but we want to create another one. Click **Create stack** and choose **With new resources (standard)** from the dropdown.
4. Down in the **Specify template** section, paste in the S3 URL that we copied a little while ago.
5. Click the **Next** button, then click it again.
6. On the next screen give this a Stack name of **ProdEnv**.
7. Choose our **DR\_Lab\_Keys** from the WebServerKeyName dropdown below that.
8. Click **Next** again, and yet again on the following screen. On this last screen, click **Create stack**.

We'll land out on the Stack details page, and here we can hit the refresh button down in the Events pane. As we refresh, we'll see the different things that our template is doing.

Once everything is finished, we'll see a **CREATE\_COMPLETE** message for our **ProdEnv** stack.

### **Disaster Strikes**

We're going to simulate a disaster, so that we can practice recovering from it.

1. Looking at the stack we just created, hit the **Delete** button (near the top of the window).
2. Then click **Delete stack** in the pop-up that we get.
3. Press the same refresh button that we did before (in the **Events** section) to see things as they happen. Eventually, we'll see that the stack has ceased to be.

### **Relaunch the Stack**

1. Let's get back into the original S3 bucket, and click on 1-CF\_Wordpress\_Blog\_with\_Lookup.json.
2. Just like the last time we did this, copy the Object URL.
3. Navigate to CloudFormation.
4. Click **Create stack** and choose **With new resources (standard)** from the dropdown.
5. Down in the **Specify template** section, paste in the S3 URL we just copied.
6. Click **Next** twice.
7. On the next screen, give the stack a name on the next screen (let's call it **DR-Env**).
8. Choose our **DR\_Lab\_Keys** from the WebServerKeyName dropdown.
9. Then click **Next**.
10. Click **Next** on the following Configure stack options screen.
11. Then on the final screen, click **Create Stack**.

Just like last time, we can watch the Events tab (hitting the refresh button occasionally) and see the stack come up. Once we see a CREATE\_COMPLETE message for DR-Env, we can move on.

## Add an EC2 Instance to the Stack

1. First, back in our original S3 bucket, get into the 2-Add\_Single\_Instance.json, and copy the Object URL.
2. Back in CloudFormation, we're going to create a new stack. Click **Create stack** and choose **With new resources (standard)** from the dropdown. D
3. own in the **Specify template** section, paste in this third S3 URL we copied.
4. Then click **Next**.
5. We'll name this one "SingleInstance" and set the InstanceType to **t2.micro**. The ModuleName should already be populated with **amilookup**.
6. Put the name of our original S3 bucket in the S3Bucket field (which we can grab by getting into that bucket and copying the name). The S3Key field should already be populated with amilookup.zip.
7. Once these are filled in, we can click **Next**.
8. We'll scroll down on the following page and click **Next**.
9. In this final screen, we have to check a box acknowledging that AWS CloudFormation might create IAM resources.
10. Then click **Create stack**.
11. Just like with the other stacks, we can keep refreshing in the Events tab to watch things spin up.

## Conclusion

We were able to simulate a disaster and then quickly recover using CloudFormation templates. Congratulations!

## Lecture: Amazon SQS and Amazon SNS: Scalability

Messaging services can be used to decouple applications. Because these services are scalable and highly available, they provide applications with the ability to grow automatically. This lesson discusses how we can use messaging services to make applications scalable.

## Lecture: AWS Lambda

AWS Lambda is a serverless compute service. It runs your code in response to events. Lambda automatically manages the underlying compute resources with no server configuration from you. There is no need to provision or manage servers. Lambda functions can be triggered by events generated by other AWS services.

## Lecture: Amazon CloudWatch Essentials

CloudWatch is a very powerful tool for monitoring and troubleshooting in AWS. In this lesson, we will discuss CloudWatch metrics and dashboards, the actions we can perform with them, and how to configure them.

## Lecture: CloudWatch Alarms

You can use CloudWatch alarms to automatically initiate actions on your behalf. An alarm watches a single metric over a specified timeframe and performs one or more specified actions, based on the value of the metric relative to a threshold over time. The action is a notification sent to an SNS topic or an Auto Scaling policy. You can also add alarms to dashboards.

## Lecture: CloudWatch Logs

CloudWatch Logs can be used to monitor, store, and access log files from EC2 instances, AWS CloudTrail, and servers running in an on-premises datacenter. It is then possible to retrieve and report on the associated log data from CloudWatch Logs.

## Lecture: CloudWatch Events

CloudWatch Events are a way to automatically take action with our AWS resources based on certain event triggers or schedules. In this lesson, we will discuss what CloudWatch Events are, how to configure them, and common uses for them.

## Lecture: AWS CloudTrail

CloudTrail is a service we can use to log all the API calls in our account. API Calls include interaction from the console, AWS CLI, and SDKs. We can also create trails that we can analyze with CloudWatch Logs or third-party tools. This lesson shows us how.

Additional Information and Resources

Log in using the credentials provided to you. You should have been given a user name of '**cloud\_user**' and a randomly generated password.

**When you log in, please make sure you are in the *us-east-1* Region.**

Learning Objectives

check\_circle**Create a CloudWatch Dashboard for the DMZ Layer**

Create a CloudWatch dashboard specific to the DMZ layer. After, set up a widget within the DMZ layer dashboard for the bastion host instance set to monitor the CPUUtilization metric.

1. In the AWS Management Console, navigate to CloudWatch.
2. Select *Dashboards*.
3. Choose *Create dashboard* (Name your dashboard DMZLayer).
4. Select a **Line** widget, then select **Metrics**.
5. Select **Configure**.
6. Under *All Metrics* > *EC2* > *Per-instance Metrics*, enter CPUUtilization into the search bar. Find the *bastion-host* server row, select it, and configure the timeline to **15** minutes.
7. Choose **Create widget**, then **Save dashboard**.

check\_circle**Create a CloudWatch Dashboard for the Application Layer**

Create a separate CloudWatch dashboard with widgets displaying metrics specific to the Application layer.

1. In the AWS Management Console, navigate to CloudWatch.
2. Select **Metrics** > **EC2**> **Per-instance Metrics**. Find **CPUUtilization**, click on it, then choose *Search for this only* in the dropdown.
3. Find the *instance-wordpress* rows and the *database* row, select them, configure the timeline to **15** minutes, and select **Stacked area**.
4. Select *Add to dashboard* and create a new dashboard. Name your dashboard AppLayer and save it.
5. Select > **Add widget** > **Number** > **ApplicationELB** > **Per AppELB Metrics**.
6. Find *RequestCount* under *Metric Name* and click the down arrow next to the name. Select **Search for this only** from the menu.
7. Select **EC2** from the CloudWatch Management Console in the new tab.
8. Select **Load Balancers** from the left-hand menu and click on the target group.
9. Verify the identity of the correct load balancer by viewing the last few digits of the *ARN*.
10. Select the corresponding load balancer in the CloudWatch Management Console tab.
11. Select **Create Widget** > **Save dashboard** > **Add widget**.
12. Repeat these steps that guided you in setting up the CPUUtilization metric for the **instance-wordpress** and **database** instances, but this time choose the NetworkIn metric.

check\_circle**Test the Widgets**

1. Copy the *DNS name* link and paste the link in a new tab.
2. Create the login credentials, then install the WordPress site.
3. Enter the credentials just created and log in to the site.
4. Navigate back to the CloudWatch Management Console tab.
5. Click on the refresh icon for the CPUUtilization dashboard.
6. Click on the three dots in the top-right corner of the CPUUtilization dashboard and select **Edit**.
7. Configure *Auto refresh* to **10 Seconds**.
8. Click **Update widget**.
9. Verify you see updated metrics using your newly created widgets.

# Using CloudWatch for Resource Monitoring

## Introduction

Welcome to this AWS hands-on lab for Using CloudWatch for Resource Monitoring!

This lab provides practical experience with creating and configuring multiple custom AWS CloudWatch dashboards and widgets.

The primary focus will be on the following features within CloudWatch:

1. CloudWatch Dashboards
2. Dashboard Widgets
3. CloudWatch Metrics

## Solution

1. Log in to the AWS Management Console using the credentials provided on the lab instructions page. Make sure you're using the us-east-1 region.

### **Create a CloudWatch Dashboard for the DMZ Layer**

1. In the AWS Management Console, start typing "CloudWatch" into the search box and click on **CloudWatch** when it appears in the list.
2. Click on **Dashboards** from the left-hand menu.
3. Click **Create dashboard**.
4. Under Dashboard name: enter "DMZLayer".
5. Click **Create dashboard**.
6. Select the **Line** option and click **Next**.
7. Select **Metrics** and click **Configure**.
8. Under All Metrics, select **EC2**.
9. Click **Per-Instance Metrics**.
10. In the filter box, enter "CPUUtilization".
11. Select the box next to **bastion-host**.
12. Click on **custom** at the top of the window and select **15 Minutes**.
13. Click **Create widget**.
14. Click **Save dashboard**.

### **Create a CloudWatch Dashboard for the Application Layer**

1. Click on **Metrics** from the left-hand menu.
2. Click **EC2**.
3. Click **Per-Instance Metrics**.
4. Find CPUUtilization under Metric Name and click the down arrow next to the name. Select **Search for this only** from the menu.
5. Select the **database** instance and both **instance-wordpress** instances by clicking the boxes next to their names.
6. Click on **custom** at the top of the window and select **15 Minutes**.
7. Click the dropdown at the top with the word **Line** displayed. Select **Stacked area** from the list of options.
8. Click **Actions** in the top-right corner and select **Add to dashboard**.
9. Click **Create new** under **Select a dashboard**. Enter "AppLayer" in the box that appears and then click the checkmark next to the box.
10. Click **Add to dashboard**.
11. Click **Add widget**.
12. Select **Number**.
13. Click **Next**.
14. Click **ApplicationELB**.
15. Click **Per AppELB Metrics**.
16. Find RequestCount under Metric Name and click the down arrow next to the name. Select **Search for this only** from the menu.
17. Duplicate the CloudWatch Management Console tab and search "EC2" in the new tab.
18. Select **EC2** from the dropdown menu options.
19. Select **Load Balancers** from the left-hand menu.
20. Click on the target group, and under Load Balancer, select **load-balancer**.
21. Under Basic Configuration, verify the identity of the correct load balancer by viewing the last few digits of the ARN.
22. Navigate back to the first CloudWatch Management Console tab and select the corresponding load balancer by clicking the checkbox (only this load balancer should be selected).
23. Click **Create widget**.
24. Click **Save dashboard**.
25. Click **Add widget**.
26. Select **Line**.
27. Select **Metrics**.
28. Click **Configure**.
29. Click **EC2**.
30. Click **Per-Instance Metrics**.
31. Find NetworkIn under Metric Name and click the down arrow next to the name. Select **Search for this only** from the menu.
32. Select the **database** instance and both **instance-wordpress** instances by clicking the boxes next to their names.
33. Click **Create widget**.

### **Test the Widgets**

1. Navigate back to the EC2 Management Console in the second tab.
2. Under Description > Basic Configuration, copy the DNS name.
3. Open a new tab and paste the load balancer.
4. Click **Continue**.
5. For Site Title, enter "Lab".
6. For Username, enter "wpuser".
7. For Password, enter "Password1".
8. Click the box for **Confirm use of weak password**.
9. For Your Email, enter "[test@acloud.guru](mailto:test@acloud.guru)".
10. Click **Install WordPress**.
11. Click **Log In**.
12. Enter the credentials just created and log in to the site.
13. Navigate back to the CloudWatch Management Console tab.
14. Click on the refresh icon for the CPUUtilization dashboard.
15. Click on the three dots in the top-right corner of the CPUUtilization dashboard and select **Edit**.
16. Click **custom** at the top-right corner and select **5 Minutes**, then select **15 Minutes**.
17. To configure Auto refresh, click on the down arrow next to the refresh icon and select **10 Seconds** as the Refresh interval.
18. Click **Update widget**.
19. Click on the refresh icon for the CPUUtilization dashboard (this may take a few minutes).

## Conclusion

Congratulations — you've completed this hands-on lab!

arn:aws:elasticloadbalancing:us-east-1:808685258119:loadbalancer/app/load-balancer/4dde51e921a32a14

## Lecture: AWS Config

AWS Config is a service we can use to evaluate the configurations of our resources. It records all the details, including relationships between resources. This can be very helpful in troubleshooting situations. We can also create a set of rules for evaluating our resources. When a resource is non-compliant with our set rules, AWS Config will let us know. This lesson shows how to configure and use the AWS Config service.

## Lecture: Health Dashboards

The AWS Service Health Dashboard provides access to the current status and historical data about every AWS Cloud service. If there’s a problem with a service, it is possible to expand the appropriate line in the details section to get more information. The AWS Personal Health Dashboard provides alerts and remediation guidance when AWS is experiencing events that impact customers.

## Lecture: AWS Billing and Organizations

Running our applications in the cloud can present large cost savings for our organizations. We must know how to monitor and optimize costs to take full advantage of these savings. AWS Billing and Cost Management hold several features we use to not only pay our bills but also monitor and optimize costs.

## Lecture: AWS Cost Explorer

AWS provides a way for us to investigate expenses in our account. We can tag resources by environments (dev, test, prod) and see cost reports for each. We can also filter costs by region, VPC, instance type, and many more. This lesson is a basic walkthrough of the AWS Cost Explorer service.

## Lecture: Cost Optimization

Costs can always increase unexpectedly. Some common mistakes contribute to these increases. In this lesson, we will discuss AWS recommendations to optimize costs and avoid some of the common causes of cost increases.

## Further reading

### **Cost Optimization:**[**https://aws.amazon.com/pricing/cost-optimization/**](https://aws.amazon.com/pricing/cost-optimization/)

## Lecture: How to Prepare for the Exam

This lesson goes over the steps you can take to ensure your success on the exam. We’ll outline a full strategy for preparing for the exam, including study tips, key terms you should know and helpful test-taking strategies.

## Lesson links

<https://aws.amazon.com/certification/certified-sysops-admin-associate/>

<https://d0.awsstatic.com/training-and-certification/docs/AWS_certified_sysops_associate_examsample.pdf>

<https://d1.awsstatic.com/training-and-certification/docs-sysops-associate/AWS_Certified_SysOps_Associate-Exam_Guide_EN_1.4.pdf>

<http://slack.linuxacademy.com/>