



Using AWS Resilience Hub To Monitor Cloud Architectures

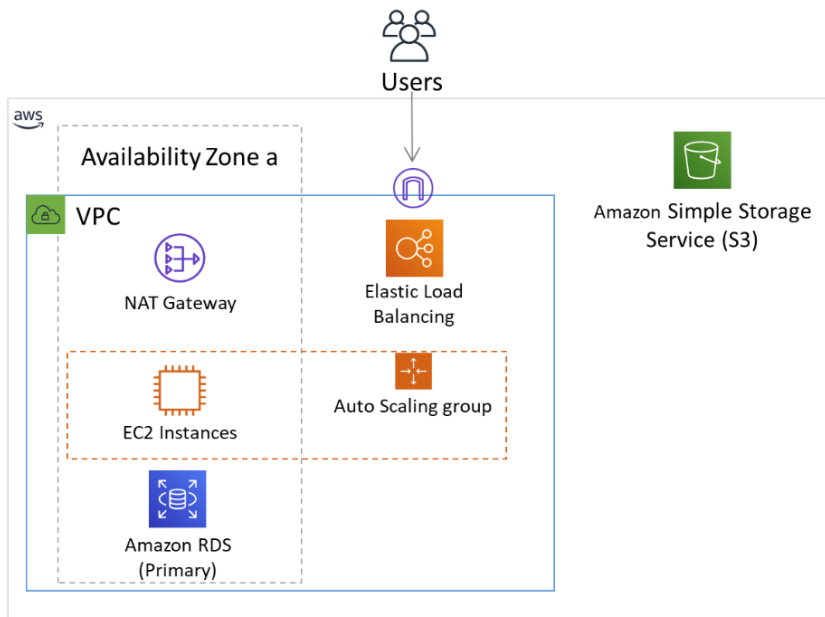
By Gerald Blackmon 08/26/2022

The objective of this lab is to deploy a workload and use AWS Resilience Hub to measure its resiliency. You will learn to use recommendations provided by Resilience Hub to make improvements to the workload and ensure it meets the resiliency targets you are looking for. You will also perform Chaos testing on the workload using AWS Fault Injection Simulator (FIS) to measure the effectiveness of workload improvements by intentionally inducing failure and seeing how the workload responds. By the end of the lab you should have an understanding of how Resilience Hub can be used to measure and improve your application resilience.

1. Deploy the workload.
2. Add and assess application using AWS Resilience Hub.
3. Resiliency findings and recommendations.
4. Operational recommendations.
5. Chaos testing with AWS Fault Injection Simulator.

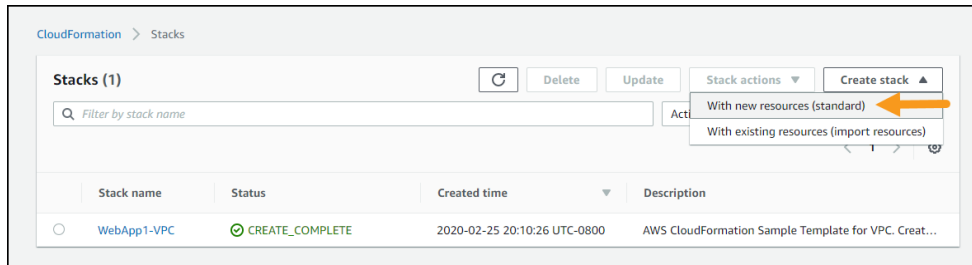
Wordpress Web application architecture

The architecture diagram below shows an overview of the WordPress application infrastructure we will deploy in phase 1. It is a 3-tier architecture consisting of an application load balancer, an autoscaled fleet of EC2 instances, and an RDS database in a single Availability Zone (AZ). The EC2 instances have outbound connectivity through a NAT Gateway and static assets for the application are stored in an S3 bucket.



Deploy The Template

1. Select **us-east-1** for the region.
2. Download the `wordpress_singleaz.yaml` CloudFormation template to your local machine.
3. Navigate to the CloudFormation console and click **Create Stack > With new resources (standard)**.

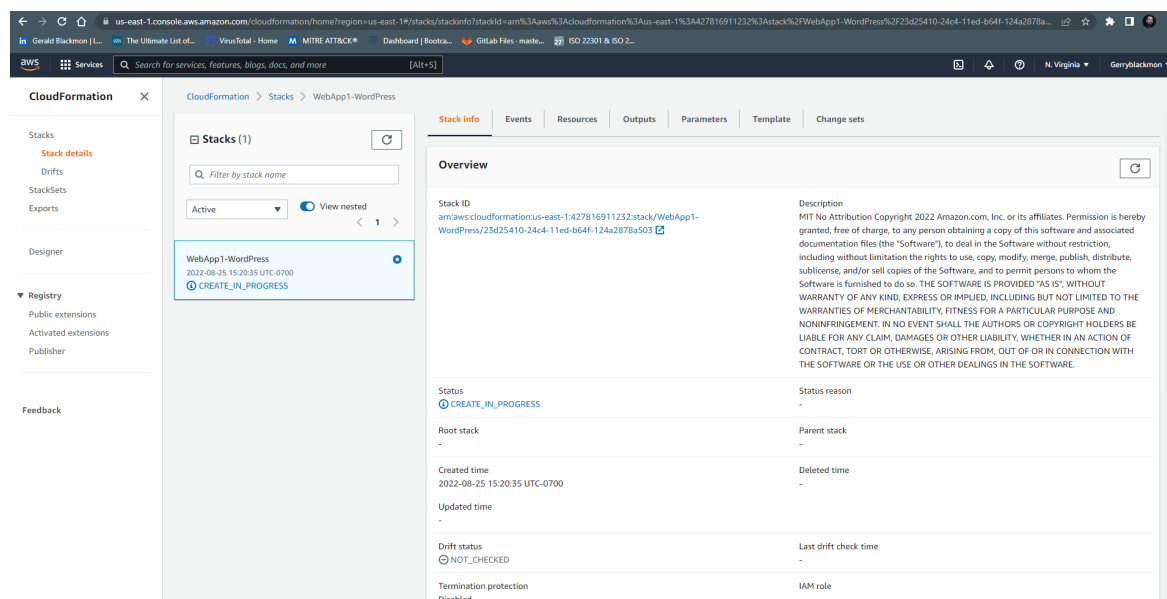


4. For **Prepare template** select **Template is ready**, and select **Upload a template file** under **Template source**. Choose the CloudFormation template that you downloaded at the beginning of this section and click **Next**.
5. Enter **WebApp1-WordPress** for the **Stack name**.

Deployed URL:

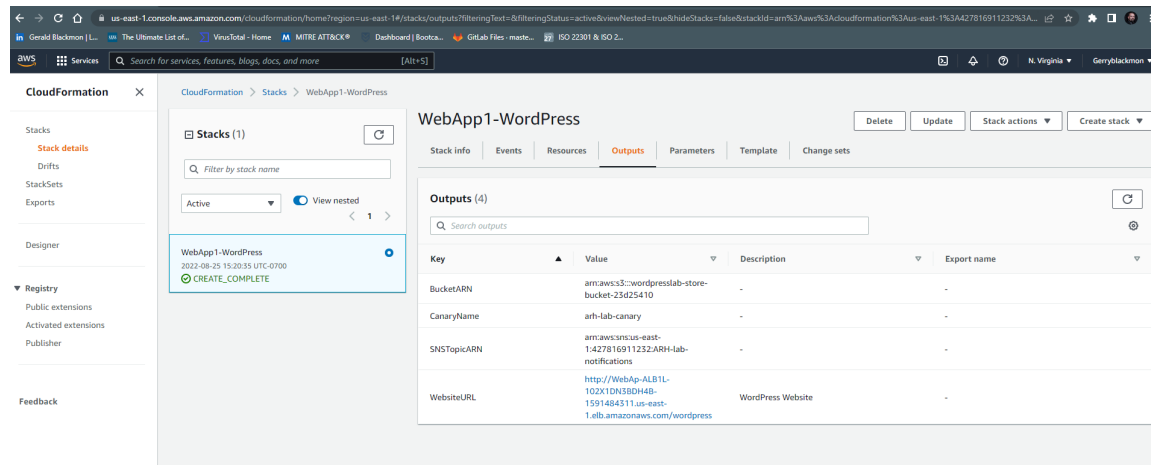
<http://WebAp-ALB1L-102X1DN3BDH4B-1591484311.us-east-1.elb.amazonaws.com/wordpress>

Stack ID

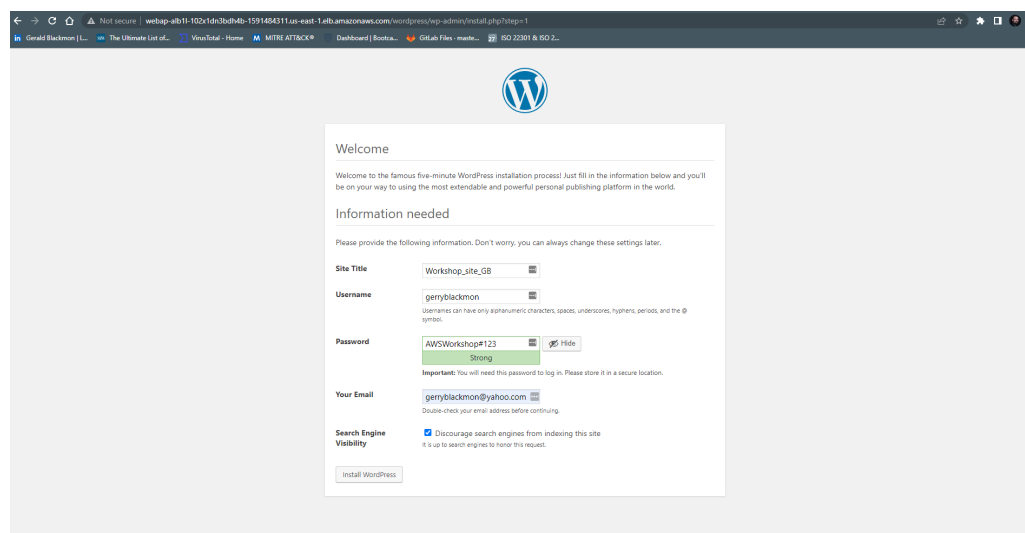


Outputs

BucketARN	arn:aws:s3:::wordpress-slab-store-bucket-23d25410	-	-
CanaryName	arh-lab-canary	-	-
SNSTopicARN	arn:aws:sns:us-east-1:427816911232:ARH-lab-notifications	-	-
WebsiteURL	http://WebAp-ALB1L-102X1DN3BDH4B-1591484311.us-east-1.elb.amazonaws.com/wordpress		



Wordpress site credentials



We selected the policy from a suggested policy bank in AWS. The policy selected was for a **Mission Critical Application**. This gave us the Application RTO & RPO as well as the Infrastructure RTO & RPO

Step 2: Identify resources

Edit

Resources (9) Info

AWS Resiliency Hub will assess the listed resources, unless you choose to exclude them.

< 1 >

⚙️

Any resource type ▾

Logical ID ▾	Resource type ▾	Component name ▾	Physical ID ▾	Status ▾
ALB1LoadBalancer	AWS::ElasticLoa...	networkingappcom...	arn:aws:elastico...	✔️ Included
CanaryBucket	AWS::S3::Bucket	storageappcompon...	arn:aws:s3::canary...	✔️ Included
RDSDBInstance1	AWS::RDS::DBIns...	databaseappcompo...	webapp1-wordp...	✔️ Included
S3BucketStore	AWS::S3::Bucket	storageappcompon...	wordpresslab-st...	✔️ Included
Web1InstanceA...	AWS::AutoScalin...	computeappcompo...	Webapp 🔗	✔️ Included
vpcnatgw	AWS::EC2::NatG...	networkingappcom...	nat-0cc1f981a3...	✔️ Included
DB1KMSKey	AWS::KMS::Key	-	343a6b5b-9c54...	⚠️ Not supported
SNSTopic	AWS::SNS::Topic	-	arn:aws:sns:us-e...	⚠️ Not supported
DB1Secret	AWS::SecretsMa...	-	arn:aws:secrets...	⚠️ Not supported

Resiliency policy

Policy name

ARH-lab

Description

-

Tier

Mission critical

Customer Application RTO and RPO Info

Type	RTO	RPO
Application	1h	15m

Cloud Infrastructure RTO and RPO Info

Type	RTO	RPO
Infrastructure	5m	5m
Availability Zone	5m	5m
Region	-	-


Assess Workload Resiliency

Once the application has been published, we take note of the workflow described by Resiliency Hub for the application.

AWS Resilience Hub > Applications > myWebApp


myWebApp ⓘ Not assessed Info Actions ▾

▼ Workflow




1. Publish application
Publish your application and its resources

Republish




2. Assess resiliency
Run an assessment to receive recommendations to improve resiliency

Assess resiliency



3. Set up recommendations
Set up recommended alarms, SOPs, and tests

Set up recommendations



4. Run experiments
Run experiments on a regular basis to validate resiliency posture

Run experiments

1. Click **Assess resiliency**, enter a report name and then **run**. Resilience Hub now assesses the resiliency posture of our application and compares it to the targets that were defined when we created the Resiliency policy.
2. The **Assessments** tab should then auto select, you may see the status go to **Pending** briefly before going to **Success**.
3. After the assessment is complete and the status has changed to **Success**, look at the **Compliance status** for the report.

After completing the assessment Resilience Hub has determined that our resiliency targets cannot be met with the current architecture. This means that if an incident were to occur, our application will not be able to recover **within the RTO and RPO** defined in the policy used for the assessment.

Resiliency assessments (1) Info Refresh Delete Run new resiliency assessment

Find assessments

<input type="checkbox"/>	Name	Status	Compliance status	Invoker	Start time	End time	ARN
<input type="checkbox"/>	Assessment-report-bczixvt809-Gerry	Success	Policy breached	User	August 25, 2022 at 4:06 PM	August 25, 2022 at 4:06 PM	arn:aws:re...

Results

1. Click on the Assessment report to expand more details about the policy breach.
2. In the **Results** tab we are able to see a few things as a summary and specifically, what has caused the breach and what the Estimated RTO/RPO is for each Disruption type. (1 example of the results shown below)

Customer Application RTO and RPO				
Application				
RTO Targeted	1h	RPO Targeted	15m	
Estimated	unrecoverable	Estimated	unrecoverable	
Breaches				
Estimated RTO and RPO				
Component	Targeted RTO	Estimated RTO	Targeted RPO	Estimated RPO
networkingappcomponent-mpg	1h	20m	15m	0s
databaseappcomponent-zgr	1h	unrecoverable	15m	unrecoverable
storageappcomponent-pav	1h	unrecoverable	The S3 bucket specified does not use Glacier storage class and does not have versioning enabled. Glacier keeps an archive of your data and versioning keeps previous versions of your objects. Consider enabling versioning and/or Glacier on this bucket.	unrecoverable
networkingappcomponent-hku	1h	0s		0s
computeappcomponent-pth	1h	30m		0s
storageappcomponent-nmj	1h	5m		0s

Resiliency recommendations

In addition to assessing your application's resiliency, Resilience Hub also provides guidance on how improvements can be made to architecture in order to satisfy the RTO/RPO requirements defined in the resilience policy that we created.

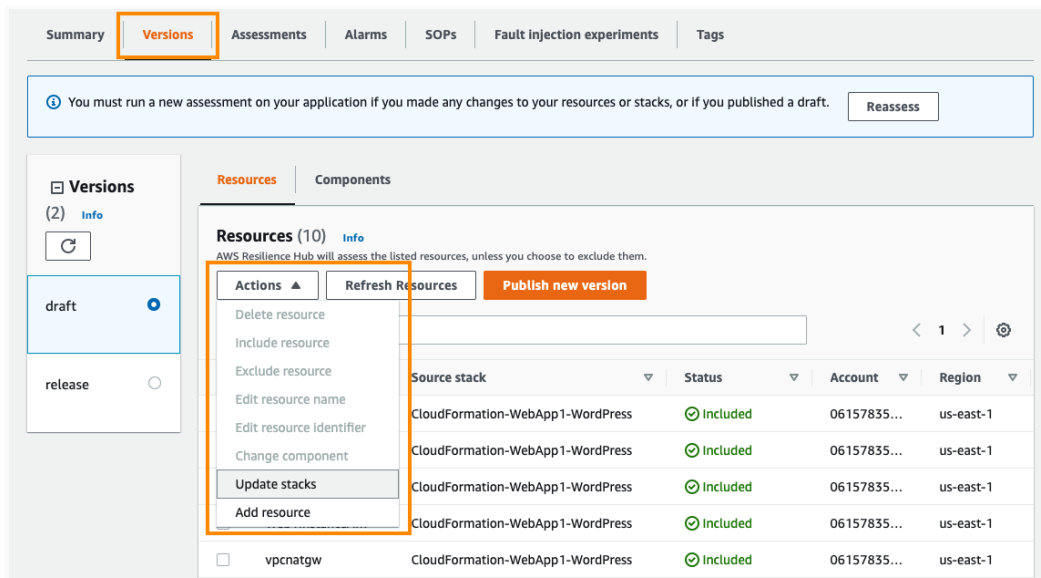
1. Select the **Resiliency recommendations** tab and scroll down and select the **databaseappcomponent** component. Resilience Hub shows us some different options here that we can take to meet our resiliency needs. These changes have associated costs and Resilience Hub gives an **Estimated cost & Architecture type**. Below are the recommendations for our database component. An important aspect to note is the cost difference between optimizing for RTO/RPO and that of optimizing for cost and minimal changes.
2. An example provided below outlines the recommendations for **databaseappcomponent-zgr**

Components (6)			
Info			
Find components			
1			
computeappcomponent-pth			
AWS::AutoScaling::AutoScalingGroup			
Policy compliance			
Current: Breached Expected: Meets			
databaseappcomponent-zgr			
AWS::RDS::DBInstance			
Policy compliance			
Current: Breached Expected: Meets			
networkingappcomponent-hku			
AWS::EC2::NatGateway			
Policy compliance			
Current: Breached Expected: Meets			
databaseappcomponent-zgr			
Info			
Optimize for Availability Zone RTO/RPO			
These changes will help you achieve the lowest possible RTO and RPO in the event of an Availability Zone disruption.			
Description			
Aurora database cluster with one read replica, with backtracking window of 24 hours.			
Estimated cost \$63.59 per month			
Architecture type WarmStandby			
Changes			
Add read replica in the same region			
Change to Aurora			
Enable cluster backtracking			
Enable instance backup with retention period 7			
Estimated RTO and RPO			
Customer Application			
Application			
RTO RPO			
15m 5m			
Cloud infrastructure			
Infrastructure			
Availability Zone			
2m 0s			
Optimize for cost			
Optimize your application to reach the lowest cost that will still meet your policy.			
Description			
Single MAZ (not Aurora) instance.			
Estimated cost \$27.19 per month			
Architecture type WarmStandby			
Changes			
Enable instance MultiAZ			
Enable instance backup with retention period 7			
Estimated RTO and RPO			
Customer Application			
Application			
RTO RPO			
30m 5m			
Cloud infrastructure			
Infrastructure			
Availability Zone			
2m 0s			
Optimize for minimal changes			
Reach your policy limit while keeping implementation changes minimal.			
Description			
Single MAZ (not Aurora) instance.			
Estimated cost \$27.19 per month			
Architecture type WarmStandby			
Changes			
Enable instance MultiAZ			
Enable instance backup with retention period 7			
Estimated RTO and RPO			
Customer Application			
Application			
RTO RPO			
30m 5m			
Cloud infrastructure			
Infrastructure			
Availability Zone			
2m 0s			

To implement the recommendations made by Resilience Hub, we will once again use AWS CloudFormation.

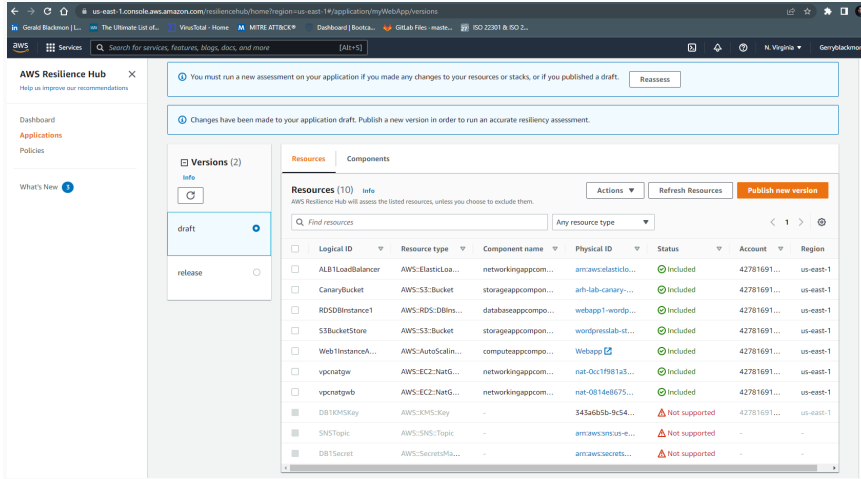
1. Download the `wordpress_multiaz.yaml` CloudFormation template, it already contains the necessary changes to implement the best practices outlined in the table above.
2. Navigate to the AWS CloudFormation console , select the stack for our workload - WebApp1-WordPress, and click Update.

3. Under Prerequisite - Prepare template, choose Replace current template. Select Upload a template file under Specify template, provide the new template that you downloaded - wordpress_multiaz.yaml
- 1.
2. Once the update is complete, go back to the Resilience Hub dashboard for the application.
3. Select the Versions tab, click Actions and select Update stacks. Since we have made changes to our CloudFormation stack, we need to import the new resources that were created into the application on Resilience Hub.

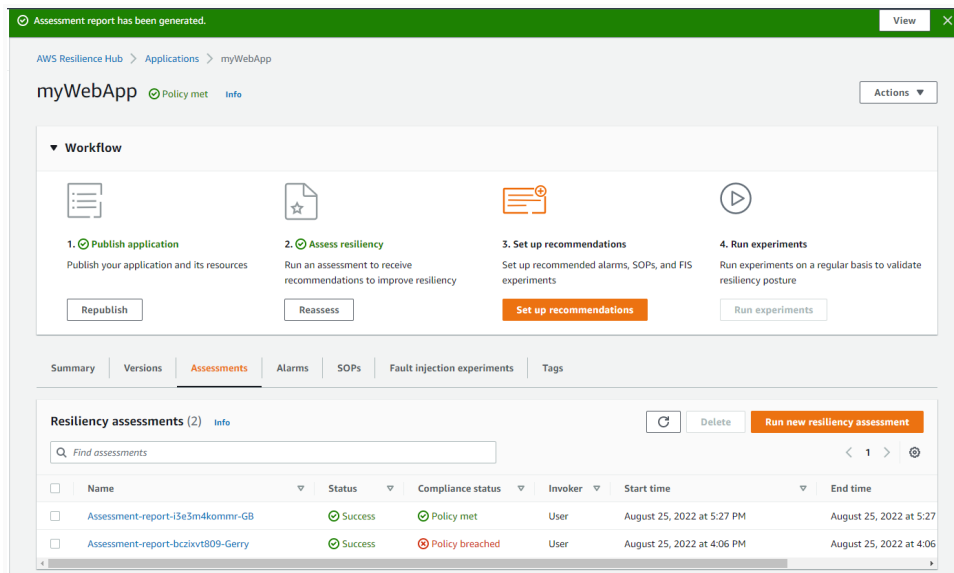


From the dropdown, select the WebApp1-WordPress stack and click Update. Resilience Hub will now pull in any new resources that may have been created as part of the CloudFormation stack update.

Now that the new resources have been imported, it is time to publish a new version of the application that reflects the updated architecture. Click on Publish new version and then Publish. The application has now been updated on Resilience Hub.



From the workflow section, select Reassess to run a new assessment of the application resiliency against our policy. The results are now “Policy met”



Click on the report to view additional details. The resiliency recommendations summary shows us the Targeted and Estimated RPO and RTO assessments against the different Disruption types.

RTO			RPO		
Resiliency recommendations			Resiliency recommendations		
Disruption type	Targeted	Estimated	Disruption type	Targeted	Estimated
Application	1h	30m	Application	15m	5m
Infrastructure	5m	2m	Infrastructure	5m	0s
Availability Zone	5m	2m	Availability Zone	5m	0s
Region	-	-	Region	-	-

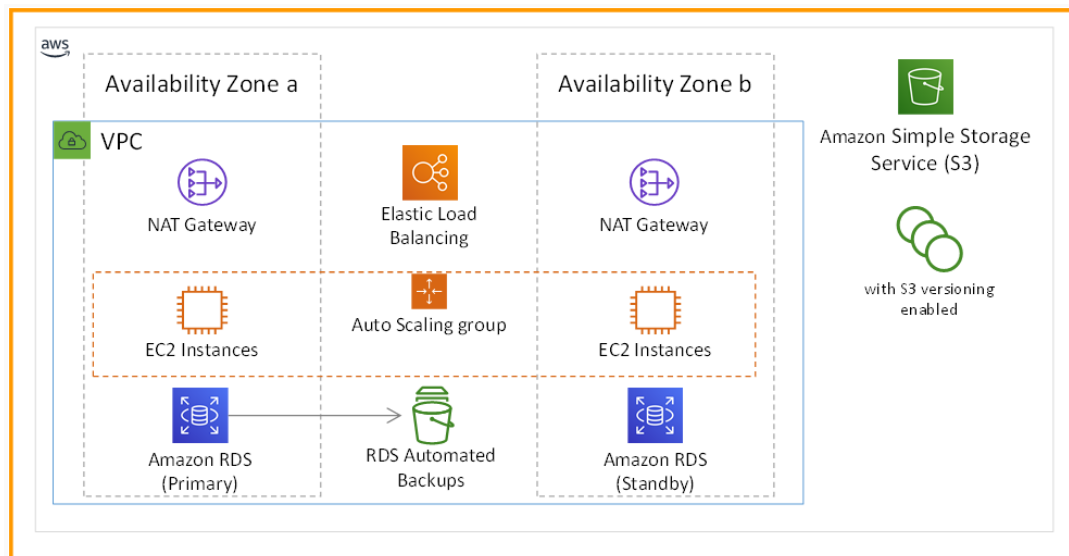
Next check the Resiliency recommendations, four out of five components state that AWS Resilience Hub has no further recommendations for this component. This is for **Computeapp**, **Networkingapp** & both the **Storageapp** components. The **databaseapp** component does still show us a potential resiliency improvement based around switching to using Aurora and backtracking.

The screenshot shows the AWS Resilience Hub interface. On the left, a list of components is shown: computeappcomponent-ydw, databaseappcomponent-yya (selected), networkingappcomponent-bbk, and storageappcomponent-lbr. The main panel displays recommendations for databaseappcomponent-yya. A red box highlights the 'Optimize for Availability Zone RTO/RPO' recommendation, which suggests switching to Aurora database cluster with one read replica and backtracking window of 24 hours. The estimated cost is \$37.00 per month. The architecture type is WarmStandby. The current RTO and RPO are 15m and 5m respectively. The recommended RTO and RPO are 2m and 0s respectively. The recommended changes are: Add read replica in the same region, Change to Aurora, and Enable cluster backtracking. The current RTO and RPO are 30m and 5m respectively. The recommended RTO and RPO are 2m and 0s respectively. The recommended changes are: Add read replica in the same region, Change to Aurora, and Enable cluster backtracking.

However, the architecture that we have currently deployed still *Meets* the criteria we have setup in our Resiliency policy within Resilience Hub.

Final Architecture

In phase 2, we implement the recommendations provided by Resilience Hub results in this final architecture. The changes are exactly as described in the table above.



Operational recommendations

In addition to resiliency recommendations, Resilience Hub also provides guidance on how to improve operations. Operational recommendations contain recommendations to set up alarms, Standard Operating Procedures (SOPs), AWS Fault Injection Simulator experiments, and AWS CloudFormation tests (learn more here). In this section, we will review the operational recommendations provided by Resilience Hub and see how they can be implemented for the application.

1. Navigate to the AWS Resilience Hub console .
2. Select the application that was created as part of this workshop, click on the Assessments tab and open the most recent assessment report.
3. Select the Operational recommendations tab. Resilience Hub provides 3 types of operational guidance:
 - Alarms - are used to monitor the health of your application and alert you if a specified metric reaches a threshold that you've configured.
 - SOPs - is a prescriptive set of steps designed to efficiently recover your application in the event of an outage or alarm.
 - Fault injection experiment templates - are designed to stress-test your application's resilience by simulating outages to your AWS resources.

The screenshot displays the AWS Resilience Hub console interface. At the top, there are four tabs: 'Results', 'Resiliency recommendations', 'Operational recommendations' (which is highlighted with an orange border), and 'Tags'. Below the tabs, a warning box with a red triangle icon states: 'Additional information may be required. The recommendations provided here are based on AWS best practices, but your application may have different resiliency needs. So you should validate each of your recommendations by testing your application against relevant failure scenarios (customer application error or infrastructure issues, for example). Note that there's an extra cost for running FIS experiments.' Below this is a 'Summary' section with the text: 'AWS Resilience Hub recommend setting up alarms, SOPs, and fault injection tests to enhance your application's resiliency. You can then create CloudFormation templates, which allow you to quickly provision and validate these resiliency measures in the Cloud.' It follows with three numbered steps: 1. Within one tab (such as Alarms) > In a tab (Alarms, for example), select recommendations you would like to set up. 2. Select Create CloudFormation template and enter a name. AWS Resilience Hub will create a template for you based on the selected recommendations. 3. From the "Templates" tab, you can access your created templates through an S3 URL. Repeat steps 1-2 for SOPs and fault injection experiments, if required. Below the summary is the 'Operational recommendations' section, which has four sub-tabs: 'Alarms' (highlighted with an orange border), 'Standard operating procedures', 'Fault Injection experiment templates', and 'Templates'. Under the 'Alarms' tab, it shows 'Alarms (16)' with an 'Info' link. There is a search bar with the placeholder 'Find Alarms', a dropdown menu set to 'Not implemented', and a 'Create CloudFormation template' button. Below these is a table with columns for 'Name' and 'Description'. The first row in the table shows a checkbox, the name 'AWSResilienceHub-RDSInstanceConnectionSpi...', and the description 'Reports when database connection count is anomalous'.

Results | Resiliency recommendations | **Operational recommendations** | Tags

Additional information may be required
The recommendations provided here are based on AWS best practices, but your application may have different resiliency needs.
So you should validate each of your recommendations by testing your application against relevant failure scenarios (customer application error or infrastructure issues, for example).
Note that there's an extra cost for running FIS experiments.

Summary

AWS Resilience Hub recommend setting up alarms, SOPs, and fault injection tests to enhance your application's resiliency. You can then create CloudFormation templates, which allow you to quickly provision and validate these resiliency measures in the Cloud.

1. Within one tab (such as Alarms) > In a tab (Alarms, for example), select recommendations you would like to set up.
2. Select Create CloudFormation template and enter a name. AWS Resilience Hub will create a template for you based on the selected recommendations.
3. From the "Templates" tab, you can access your created templates through an S3 URL. Repeat steps 1-2 for SOPs and fault injection experiments, if required.

Operational recommendations

Alarms | Standard operating procedures | Fault Injection experiment templates | Templates

Alarms (16) [Info](#) [Create CloudFormation template](#)

< 1 > ⚙

<input type="checkbox"/>	Name	Description
<input type="checkbox"/>	AWSResilienceHub-RDSInstanceConnectionSpi...	Reports when database connection count is anomalous

In the following sections, we will implement the Alarms and Fault injection experiment templates recommendations. Implementing SOPs is a similar process but is outside the scope of this lab. Learn more about implementing SOPs.

Alarms

Select the Alarms tab to see a list of alarms that Resilience Hub recommends.

Some of the alarms such as `AWSResilienceHub-SyntheticCanaryInRegionAlarm_2021-04-01` require additional configuration. You can find details on what the prerequisites are by clicking on Configuration for the alarms that need one.

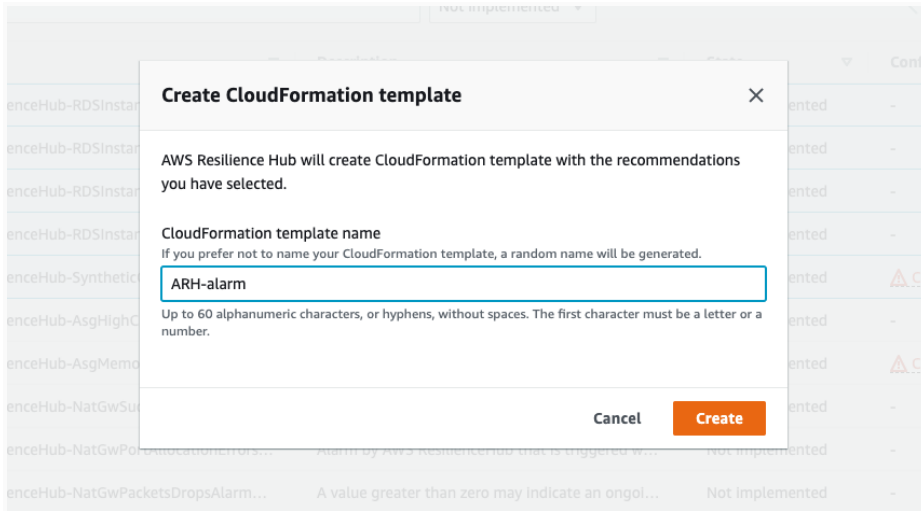
For this lab, select the following synthetics canary and RDS alarms:

- `AWSResilienceHub-SyntheticCanaryInRegionAlarm_2021-04-01`
- `AWSResilienceHub-RDSInstanceLowStorageAl`
- `AWSResilienceHub-RDSInstanceLowMemoryAlarm_2020-04-01`
- `AWSResilienceHub-RDSInstanceOverUtilizedCpuAlarm_2020-04-01`
- `AWSResilienceHub-RDSInstanceConnectionSpikeAlarm_2020-04-01`

The screenshot shows the AWS Resilience Hub interface. The 'Operational recommendations' section is active, and the 'Alarms' tab is selected. The 'Alarms (5/16)' section shows a list of recommended alarms. The first five alarms are selected, and a 'Create CloudFormation template' button is visible in the top right corner.

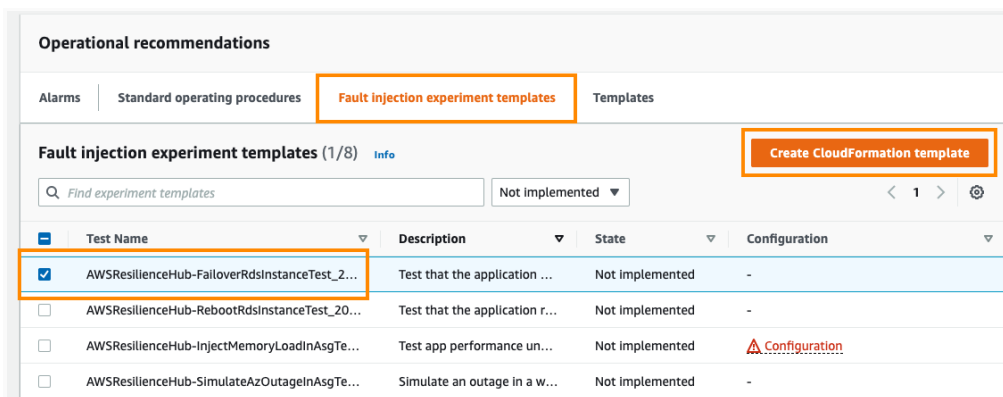
Name	Description
<input checked="" type="checkbox"/> AWSResilienceHub-RDSInstanceConnectionSpikeAlarm_20...	Reports when database connection count is anomalous
<input checked="" type="checkbox"/> AWSResilienceHub-RDSInstanceOverUtilizedCpuAlarm_202...	Reports when database used CPU is high
<input checked="" type="checkbox"/> AWSResilienceHub-RDSInstanceLowMemoryAlarm_2020-0...	Reports when database free memory is low
<input checked="" type="checkbox"/> AWSResilienceHub-RDSInstanceLowStorageAlarm_2020-0...	Reports when database free storage is low
<input checked="" type="checkbox"/> AWSResilienceHub-SyntheticCanaryInRegionAlarm_2021-0...	A Monitor for the entire App. Configured to constantly verify applicaiton API/endpoints are ava
<input type="checkbox"/> AWSResilienceHub-AsgHighCpuUtilizationAlarm_2020-07-13	Reports when autoscale group used CPU is high
<input type="checkbox"/> AWSResilienceHub-AsgMemoryUtilizationAlarm_2021-04-05	Reports when memory utilization is high
<input type="checkbox"/> AWSResilienceHub-NatGwSuccessfulConnectionPercentage...	Alarm by AWS ResilienceHub that is triggered when ConnectionEstablishedCount is less than 50
<input type="checkbox"/> AWSResilienceHub-NatGwPortAllocationErrorsAlarm_2020...	Alarm by AWS ResilienceHub that is triggered when too many concurrent connections are open

Resilience Hub makes it easy to implement these best practice alarms by providing CloudFormation templates. Click Create CloudFormation template, enter ARH-alarm for the template name and click Create. We will cover how to use these templates in a later section of the lab.

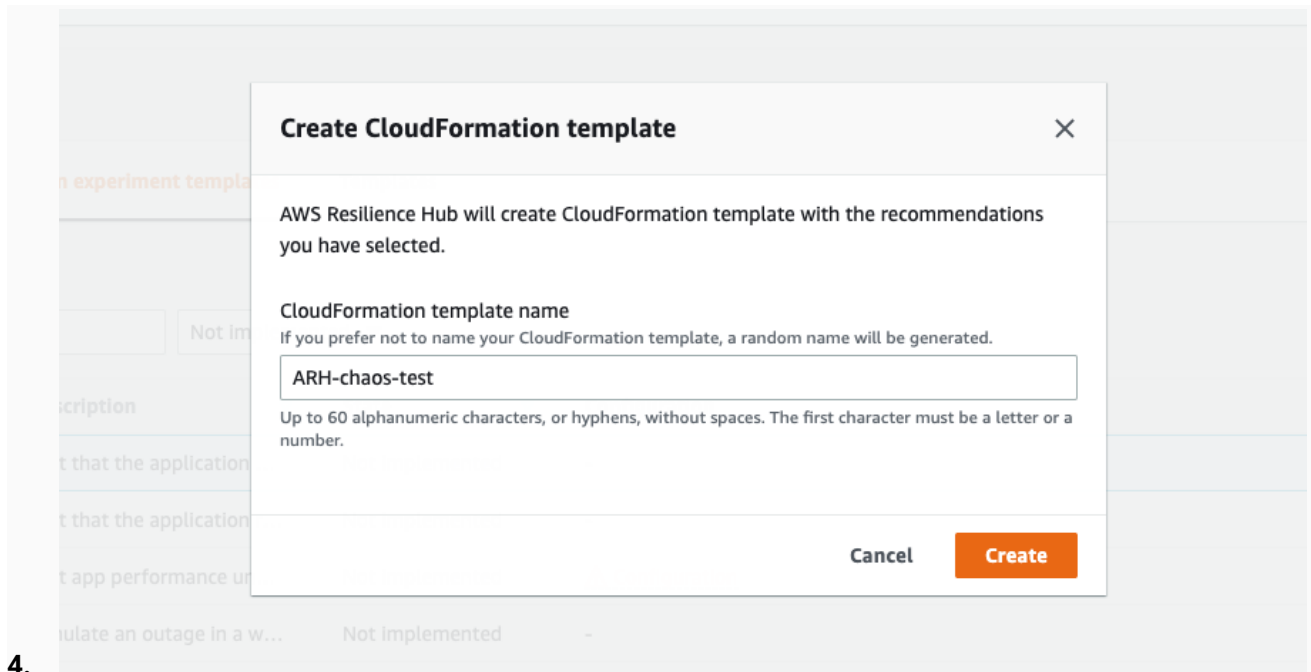


Fault injection experiment templates

1. Select the Fault injection experiment templates tab to see a list of fault injection experiments you can run on your application.
2. Some of the experiment templates such as `AWSResilienceHub-InjectCpuLoadInAsgTest_2021-09-22` requires additional configuration. You can find details on what the prerequisites are by clicking on Configuration for the templates that need one.
3. For this lab, select the `AWSResilienceHub-FailoverRdsInstanceTest_2020-04-01` experiment template. The Description for the experiment template provides information on what it does.



Click Create CloudFormation template and enter ARH-chaos-test for the template name.

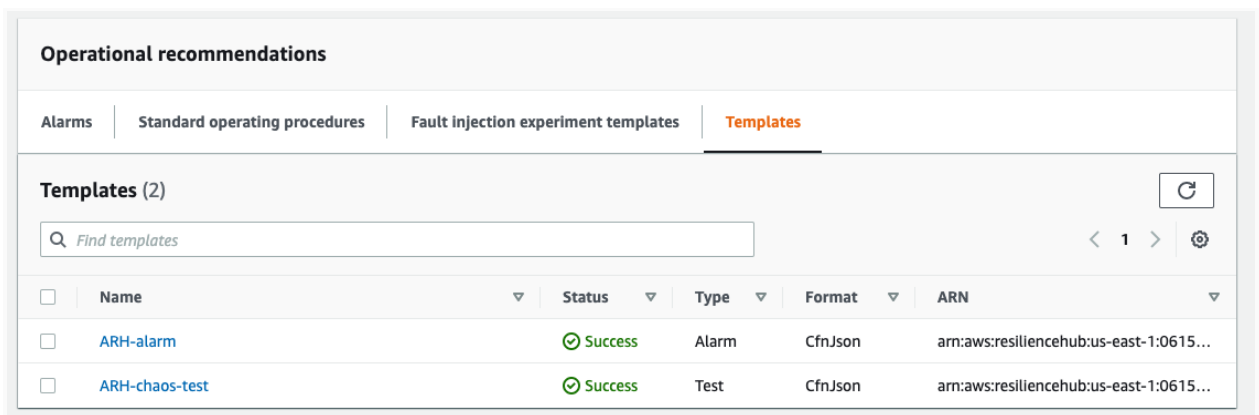


4.

CloudFormation templates

In the previous sections, we selected the operational recommendations to implement from the list provided by Resilience Hub and generated CloudFormation templates for them. Now, we will deploy the templates generated by Resilience Hub and dive into the benefits that the application gains.

1. Under Operational recommendations, select the Templates tab. You will see a list of CloudFormation templates generated by Resilience Hub. Click on the ARH-alarm template.



2. ARH stores the CloudFormation templates in Amazon Simple Storage Service (S3) for durability and ease of access. Click on the link under Templates S3 Path to navigate to the S3 location where the template is stored.

AWS Resilience Hub > Applications > myWebApp > Assessment Reports > Assessment-report-7qy3vtngp7d > ARH-alarm

ARH-alarm Delete

Template details

ARN arn:aws:resiliencehub:us-east-1:██████████:recommendation-template/962b7ae9-fd62-4b53-906c-01fa0de37c4c	Template status Success	Start time June 07, 2022, 6:48 PM
Recommendation type Alarm	Template format CfnJson	End time June 07, 2022, 6:48 PM
Templates S3 Path s3://aws-resilience-hub-artifacts-██████████-8-03zho0udxyu7/myWebApp/ARH-alarm/		Details -

- Depending on the type of recommendation the template was created for, you should see a folder named alarm, sop, or test. Since this is the ARH-alarm template, you should see a folder named alarm in the bucket.

Amazon S3 > Buckets > aws-resilience-hub-artifacts-061578351048-03zho0udxyu7 > myWebApp/ > ARH-alarm/

ARH-alarm/ Copy S3 URI

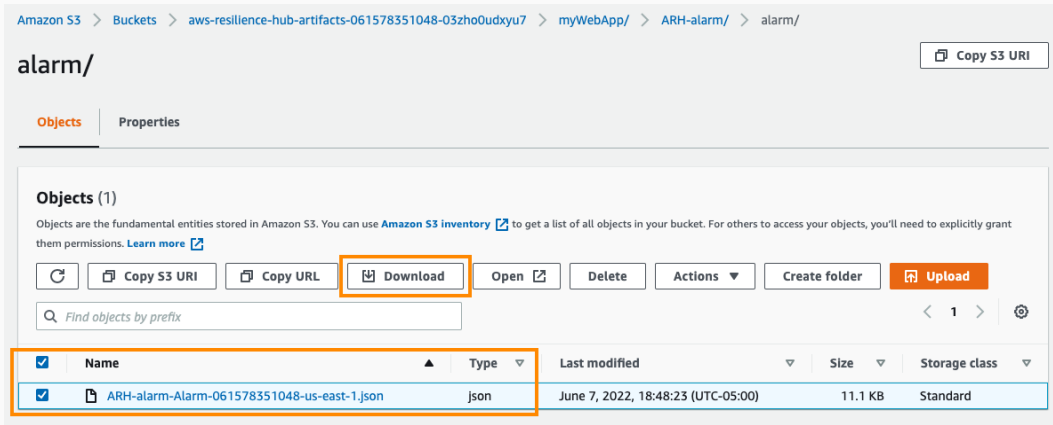
Objects | Properties

Objects (3)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 Inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	alarm/	Folder	-	-	-
<input type="checkbox"/>	manifest.json	json	June 7, 2022, 18:48:24 (UTC-05:00)	2.3 KB	Standard
<input type="checkbox"/>	README.md	md	June 7, 2022, 18:48:24 (UTC-05:00)	1.4 KB	Standard

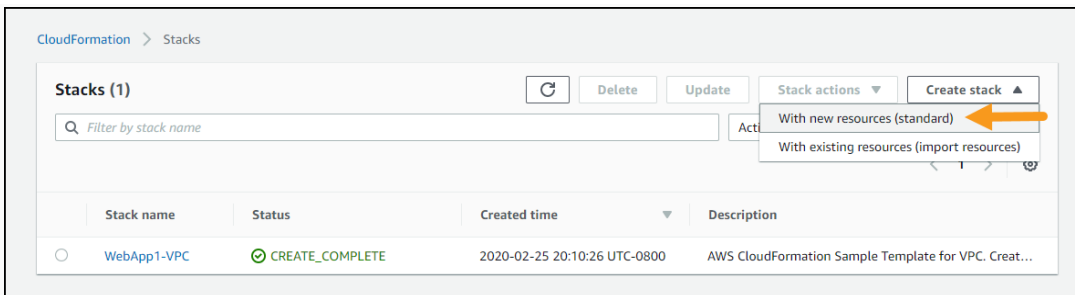
- Navigate into the alarm folder and you will see the CloudFormation template (a JSON file).
- Download the template file to your local machine by clicking Download.



- 6.
7. Repeat steps 1-7 for the ARH-chaos-test template.

Once both templates have been downloaded, it is time to deploy them using CloudFormation.

1. Navigate to the CloudFormation console and click Create Stack > With new resources (standard).



- 2.
3. For Prepare template select Template is ready, and select Upload a template file under Template source. Select the ARH-alarm template that was downloaded in the previous section.
4. Enter WebApp1-Monitoring for the Stack name.
5. For the parameters, enter the value of the SNS Topic ARN and canary name obtained from the Outputs section of the WebApp1-WordPress stack.

Specify stack details

Stack name

Stack name

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

CanaryName

The name of a CloudWatch Synthetics canary indicating the overall health status of the application.

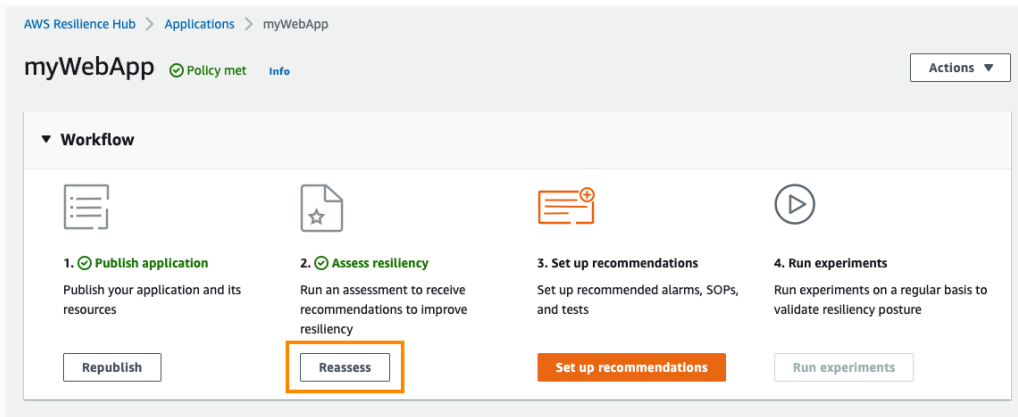
SNSTopicARN

The ARN of the SNS topic to which alarm status changes are to be sent. This must be in the same region being deployed.

Cancel Previous Next

- 6.

7. Click Next till you get to the Review page and click Create stack.
8. Wait for CREATE_COMPLETE
9. Wait for the CloudFormation stack status to change to CREATE_COMPLETE before proceeding.
10. After the stack reaches CREATE_COMPLETE, navigate to the ARH console and select the application myWebApp.
11. Under Workflow, click Reassess to run a new assessment of the application.



- 12.
13. After the assessment is complete, consider: What changes would you expect you see in the Resilience Hub results? (hint: the architecture did not change, so we would *not* expect changes to the Resiliency recommendations.)
 - Click Applications on the left side
 - Click your application named myWebApp
 - Click the Summary tab to see the Resiliency Score. You should see that the Resiliency Score of the application has increased. This is because of the operational recommendations (creating alarms) that have been implemented.
14. Navigate back to the CloudFormation console and select the WebApp1-Monitoring stack. Click on the Resources tab and locate the logical ID `AWSResilienceHubSyntheticCanaryInRegionAlarm20210401useast1Alarm`. Copy the value of the corresponding physical ID, this will be used as the parameter value for the chaos testing stack.

WebApp1-Monitoring

Stack info | Events | **Resources** | Outputs | Parameters | Template | Change sets

Resources (15)

Search resources

Logical ID	Physical ID	Type	Status	Status reason
AWSResilienceHubRDSInstanceConnectionSpikeAlarm20200401wrqojna5qzev0qAlarm	AWSResilienceHub-RDSInstanceConnectionSpikeAlarm-2020-04-01_myWebApp_wrqojna5qzev0q	AWS::CloudWatch::Alarm	CREATE_COMPLETE	-
AWSResilienceHubRDSInstanceLowMemoryAlarm20200401wrqojna5qzev0qAlarm	AWSResilienceHub-RDSInstanceLowMemoryAlarm-2020-04-01_myWebApp_wrqojna5qzev0q	AWS::CloudWatch::Alarm	CREATE_COMPLETE	-
AWSResilienceHubRDSInstanceLowStorageAlarm20200401wrqojna5qzev0qAlarm	AWSResilienceHub-RDSInstanceLowStorageAlarm-2020-04-01_myWebApp_wrqojna5qzev0q	AWS::CloudWatch::Alarm	CREATE_COMPLETE	-
AWSResilienceHubRDSInstanceOverUtilizedCpuAlarm20200401wrqojna5qzev0qAlarm	AWSResilienceHub-RDSInstanceOverUtilizedCpuAlarm-2020-04-01_myWebApp_wrqojna5qzev0q	AWS::CloudWatch::Alarm	CREATE_COMPLETE	-
AWSResilienceHubSyntheticCanaryInRegionAlarm20210401useast1Alarm	AWSResilienceHub-SyntheticCanaryInRegionAlarm-2021-04-01_myWebApp_us-east-1	AWS::CloudWatch::Alarm	CREATE_COMPLETE	-
appcommonalarmsyntheticcanary20210401useast1AlarmInfoSSMParameter	/ResilienceHub/Info/Alarm/5063ce26-855d-492a-8cd3-09d96fbec9a/app-common-alarm-synthetic-canary-2021-04-01_us-east-1	AWS::SSM::Parameter	CREATE_COMPLETE	-

15. Repeat steps 1-7 and create a CloudFormation stack using the ARH-chaos-test template downloaded in the previous section. Use WebApp1-FIS as the stack name and enter the physical ID of the canary alarm (obtained from the previous step) for the parameter CanaryAlarmName.
- 16.

Specify stack details

Stack name

Stack name

WebApp1-FIS

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

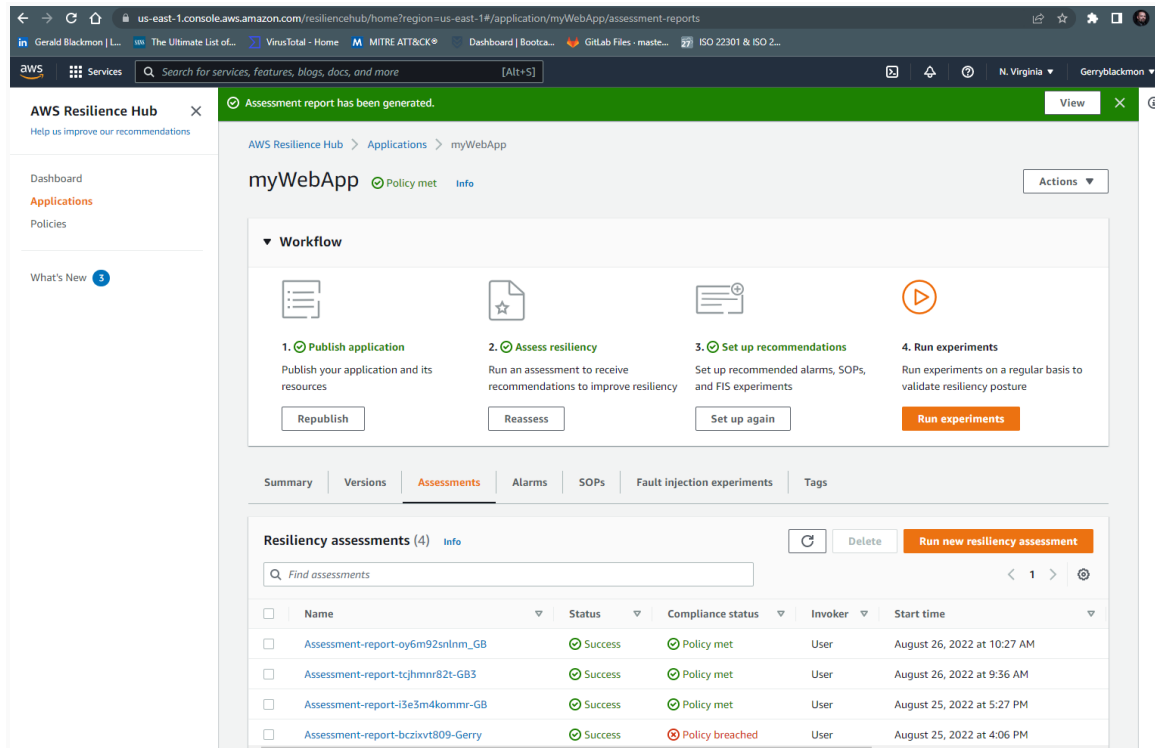
CanaryAlarmName

Name of a CloudWatch alarm indicating application-level health. Alarm status should be OK after test/SOP execution.

AWSResilienceHub-SyntheticCanaryInRegionAlarm-2021-04-01_myWebApp_us-east-1

Cancel Previous Next

After the stack has reached CREATE_COMPLETE, reassess the application on Resilience Hub and see if the Resiliency Score changes. Learn more about what the resiliency score is and how it is calculated



In the next section, we will run a series of tests to verify the resilience as well as the operational improvements made to the application.

Chaos testing with AWS Fault Injection Simulator

Chaos Engineering is the discipline of experimenting on a system (such as by injecting faults) in order to build confidence in the system's capability to withstand turbulent conditions in production. While Resilience Hub is a great way to understand your application's resiliency and make improvements, chaos testing will increase your confidence as you will be able to see your application's response to failures. In this section, we will inject faults into our application using AWS Fault Injection Simulator .

Fault injection

1. Navigate to the Resilience Hub console and select the application myWebApp.
2. Under Workflow, click Run experiments. This will take you to the Fault injection experiments tab where you will see the FIS experiment template that was created in the previous section.
3. Select the experiment template listed there (this was created in the previous section using CloudFormation) and read through the Description to understand what the experiment will be.
4. Click Start experiment and confirm you want to run this experiment.

AWS Resilience Hub > Applications > myWebApp

myWebApp Policy met [Info](#) Actions

Workflow

1. Publish application
Publish your application and its resources

Republish

2. Assess resiliency
Run an assessment to receive recommendations to improve resiliency

Reassess

3. Set up recommendations
Set up recommended alarms, SOPs, and tests

Set up again

4. Run experiments
Run experiments on a regular basis to validate resiliency posture

Run experiments

Summary | Versions | Assessments | Alarms | SOPs | **Fault injection experiments** | Tags

Experiment templates | Experiments

Experiment templates (1/1) [Info](#) Start experiment

Find experiment templates

Experiment template ID	Description	Creation time	Last update time
EXTew8T89ReHL7AV6	Runs AWSResilienceHub-FailoverRdsInstanceT...	June 07, 2022, 7:25 PM	June 07, 2022, 7:25

- The Experiments tab should auto-select and you should see the experiment is Initiating. The State will change to Running after a few seconds and eventually reach Completed. The experiment we are running forces the RDS database to failover to the secondary instance by simulating an outage of the primary instance or Availability Zone. This is running the RebootDBInstance API call under hood and forcing the failover to simulate failure.
- Navigate to the RDS console and select the database instance that was created as part of this workshop. You should see that the Status is Rebooting.

Wait for the experiment to reach Completed

Summary | Versions | Assessments | Alarms | SOPs | **Fault injection experiments** | Tags

Experiment templates | **Experiments**

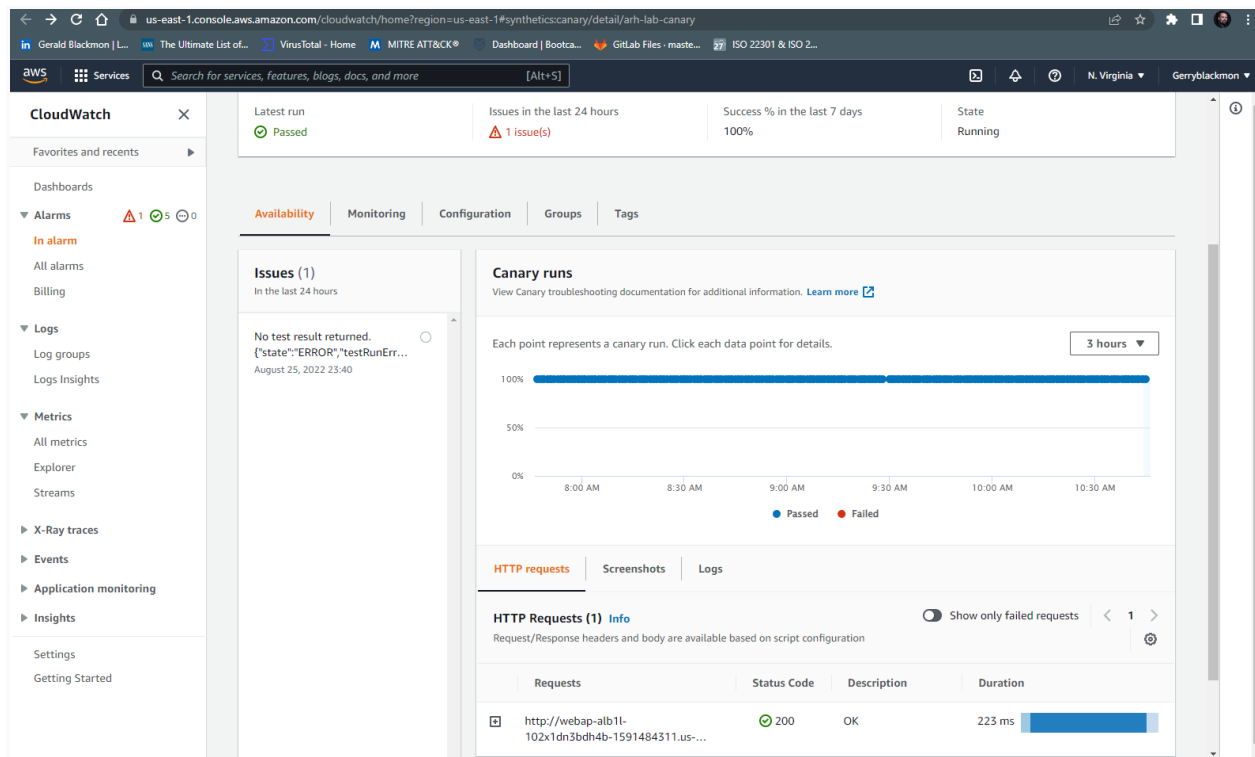
Experiments (1) [Info](#) Stop experiment

List of active AWS Fault Injection Simulator experiments

Find experiments

Experiment ID	Experiment template ID	State
EXPfYkUPW44GRG6heA	EXTew8T89ReHL7AV6	Completed

Navigate to the CloudWatch Synthetics Canaries console. This is the Synthetics canary that has been monitoring our application endpoint by periodically sending it a request and verifying that it is reachable. Observe the Canary runs data to see if there was an application outage when the FIS experiment was running. You might have to wait a few minutes for the graphs to be updated.



Conclusion

You should see that there was no application outage even when an RDS "outage" occurred and the primary database was no longer reachable. This is because we implemented the resiliency recommendation made by Resilience Hub and deployed the RDS instances across multiple AZs. With chaos testing completed and results verified, you can now be confident that your application can withstand the loss of the primary RDS database or Availability Zone.