



# Security Architecture Lab Guide

Sponsored by



August 2018

v1.0

## Logistics

- **Format:** Overview presentation and lab setup, followed by paced exercises with a section recap.
- **Workshop Duration:** 1.5-2 hours
- **Target Audience:** Technical security users (security engineers, architects, DevOps) who have heard of Dome9 and know what Dome9 offers
- The organizing team is comprised of one speaker and 1-2 technical staff to help out and answer questions
- Participants bring their own laptops and have an AWS account setup (preferably beforehand) - **Please do this early on since it takes a few hours for a new AWS account to sync with a CFT template**
- Participants need to download the Cloudformation (**CFT**) template to run this lab
  - Please download CFT from the [Github here](#)

# Table of Contents

- 1. Lab Overview and AWS Setup (15-20 min)**
    - a. What is a security architecture review?
    - b. Walkthrough of lab environment and exercises
    - c. Exercise 1.1: Setup AWS account
    - d. Exercise 1.2: Deploy sandbox environment in AWS
  - 2. AWS Security Policy Lab (30- 40 minutes)**
    - a. Overview of infrastructure security challenges in the cloud
    - b. Exercise 2.1: Identify zombie security group (no instance, but permissive rule)
    - c. Exercise 2.2: Identify an exposed internal asset in the AWS environment - Part 1
    - d. Exercise 2.3: Identify an exposed internal asset in the AWS environment - Part 2
    - e. Section Wrap Up
  - 3. AWS S3 Bucket Lab (45 minutes)**
    - a. Overview of S3 bucket security
    - b. Exercise 3.1: S3 Access Controls (exposed ACLs and Bucket policies)
    - c. Exercise 3.2: S3 Encryption Best Practices
    - d. Exercise 3.3: S3/CloudTrail Logging Best Practices
    - e. Section wrap up
  - 4. Dome9 Overview (5 min)**
  - 5. Offboarding**
-

# AWS Setup

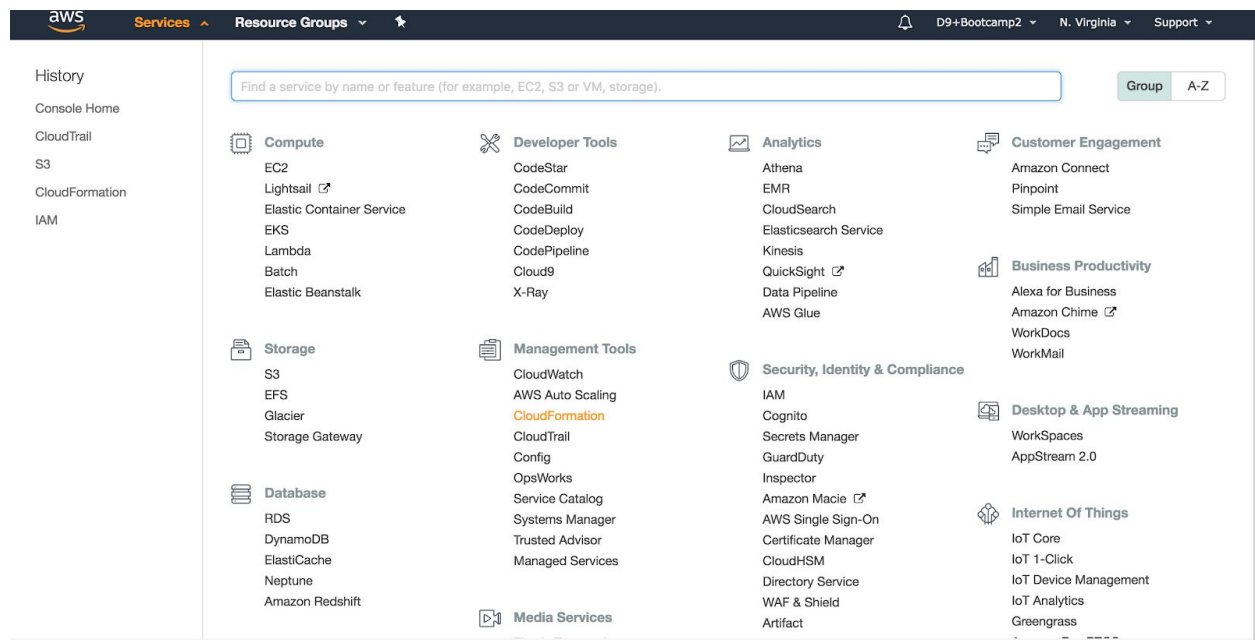
## Exercise 1.1: Setup lab environment (Login to your lab AWS account)

The instructor should provide you with an AWS Account for this workshop. Ensure you have an AWS account setup before proceeding.

## Exercise Complete!

## Exercise 1.2: Deploy sandbox AWS environment

Navigate to AWS Cloudformation



Click on create stack and select “upload a template to S3” and choose the CFT file that you downloaded and click next

The screenshot shows the AWS CloudFormation console with the 'Create Stack' wizard. The 'Select Template' step is active. The left sidebar shows 'Select Template' as the current step, with 'Specify Details', 'Options', and 'Review' as subsequent steps. The main content area has a heading 'Select Template' and a description: 'Select the template that describes the stack that you want to create. A stack is a group of related resources that you manage as a single unit.' There are two main options: 'Design a template' (with a 'Design template' button) and 'Choose a template'. Under 'Choose a template', there are three radio buttons: 'Select a sample template', 'Upload a template to Amazon S3' (which is selected), and 'Specify an Amazon S3 template URL'. The 'Upload a template to Amazon S3' option has a 'Choose File' button and a 'No file chosen' label. The 'Specify an Amazon S3 template URL' option has a text input field and a 'No file chosen' label. At the bottom right, there are 'Cancel' and 'Next' buttons.

Create a stack name such as “<yourname>LoftLab” and select us-east1-a, 1-b, and 1-c for subnetAza, subnetAzb, subnetAzc and click next

The screenshot shows the AWS CloudFormation console with the 'Create Stack' wizard. The 'Specify Details' step is active. The left sidebar shows 'Specify Details' as the current step, with 'Select Template', 'Options', and 'Review' as subsequent steps. The main content area has a heading 'Specify Details' and a description: 'Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. Learn more.' There is a 'Stack name' input field with the value 'LoftLab'. Below this is the 'Parameters' section. It contains four parameter groups: 'AmiName' with a text input field containing 'ami-14c5486b' and a description 'Name of the AWS AMI IN THIS REGION.'; 'SubnetAZa' with a dropdown menu showing 'us-east-1a' and a description 'First Availability Zone of the Subnets'; 'SubnetAZb' with a dropdown menu showing 'us-east-1b' and a description 'Second Availability Zone of the Subnets'; and 'SubnetAZc' with a dropdown menu showing 'us-east-1c' and a description 'Third Availability Zone of the Subnets'. At the bottom right, there are 'Cancel', 'Previous', and 'Next' buttons.

Click next **twice** and select “**I acknowledge that AWS Cloudformation might create resources**” and click create.

Rollback Triggers

No monitoring time provided

No rollback triggers provided

Advanced

Notification	
Termination Protection	Disabled
Timeout	none
Rollback on failure	Yes

Capabilities

**i** The following resource(s) require capabilities: [AWS::IAM::Role]

This template contains Identity and Access Management (IAM) resources that might provide entities access to make changes to your AWS account. Check that you want to create each of these resources and that they have the minimum required permissions. [Learn more.](#)

☐ I acknowledge that AWS CloudFormation might create IAM resources.

[Quick Create Stack](#) (Create stacks similar to this one, with most details auto-populated)

[Cancel](#) [Previous](#) [Create](#)

You will need to wait 5 minutes for the CFT to automatically deploy the environment in your AWS account. At the end you should see the below screen:

The screenshot shows the AWS CloudFormation console. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a search icon. Below this, the 'CloudFormation' service is selected, and the 'Stacks' view is active. A 'Create Stack' button and a 'Design template' button are visible. The main area shows a table of stacks with the following data:

Stack Name	Created Time	Status	Description
demo	2018-08-18 12:36:00 UTC-0700	CREATE_COMPLETE	

Below the stack list, there are tabs for 'Overview', 'Outputs', 'Resources', 'Events', 'Template', 'Parameters', 'Tags', 'Stack Policy', 'Change Sets', and 'Rollback Triggers'. The 'Events' tab is selected, showing a list of events for the 'demo' stack. The events are filtered by 'Status' and show the following data:

Time	Status	Type	Logical ID	Status Reason
2018-08-18 12:38:50 UTC-0700	CREATE_COMPLETE	AWS::CloudFormation::Stack	demo	
2018-08-18 12:38:47 UTC-0700	CREATE_COMPLETE	AWS::ElasticLoadBalancingV2::LoadBalancer	Alb	
2018-08-18 12:38:20 UTC-0700	CREATE_COMPLETE	AWS::EC2::Instance	RabbitMQ1	
2018-08-18 12:37:40 UTC-0700	CREATE_COMPLETE	AWS::EC2::Instance	AgentService2	
2018-08-18 12:37:40 UTC-0700	CREATE_COMPLETE	AWS::EC2::Instance	monitoring2	
2018-08-18 12:37:33 UTC-0700	CREATE_COMPLETE	AWS::EC2::Instance	appserver2	
2018-08-18 12:37:33 UTC-0700	CREATE_COMPLETE	AWS::EC2::Instance	appserver1	
2018-08-18 12:37:27 UTC-0700	CREATE_COMPLETE	AWS::ElasticLoadBalancingV2::LoadBalancer	WebLB	

**Exercise Complete!** You have now deployed the sandbox AWS environment in your account.

## AWS Security Policy Lab

### Exercise 2.1: Identify zombie security group

In your AWS account, navigate to **N.Virginia region**. Explore EC2 instances, Security Groups, IAM and other services.

The screenshot shows the AWS Management Console interface. On the left is a navigation menu with categories like EC2 Dashboard, INSTANCES, IMAGES, ELASTIC BLOCK STORE, NETWORK & SECURITY, ELASTIC BALANCING, AUTO SCALING, and SYSTEMS MANAGER SERVICES. The main content area displays a list of Security Groups. The 'Common SG' (sg-a3e455d8) is selected, showing its details and a list of inbound rules.

Name	Group ID	Group Name	VPC ID	Description
Application-Load-Balancer	sg-60b2019	Application-Load-Balancer	vpc-80e113ec	Application-Load-Balancer
Main Java application	sg-80e455e5	App1 Servers	vpc-80e113ec	Main Java application
no description	sg-233d6645	App2_ApplicationServers	vpc-80e113ec	no description
all DB servers belongs to App2	sg-473d9629	App2_DB	vpc-80e113ec	all DB servers belongs to App2
all ELBs that belongs to App2	sg-b50a0d3	App2_LoadBalancers	vpc-80e113ec	all ELBs that belongs to App2
web servers that belongs to application2	sg-0735981	App2_Web	vpc-80e113ec	web servers that belongs to application2
SG for basic services	sg-93b0eaf	Base SG	vpc-80e113ec	SG for basic services
Enable HTTPS access via port 443	sg-288b91b	CloudFormer-WebServerSe...	vpc-80e113ec	Enable HTTPS access via port 443
Enable HTTPS access via port 443	sg-6291aa51	CloudFormer-WebServerSe...	vpc-80e113ec	Enable HTTPS access via port 443
shared rules for all instances	sg-a3e455d8	Common SG	vpc-80e113ec	shared rules for all instances

Type	Protocol	Port Range	Source	Description
All TCP	TCP	0 - 65535	sg-bde455d8 (DB servers)	
Custom UDP Rule	UDP	151 - 162	212.25.105.39/32	
Custom UDP Rule	UDP	151 - 162	212.25.105.40/32	
Custom UDP Rule	UDP	151 - 162	sg-e4e45581 (JB-Web)	
Custom TCP Rule	TCP	443	4.4.4.4/32	
Custom TCP Rule	TCP	443	50.50.50.50/32	
Custom TCP Rule	TCP	443	100.2.3.4/32	
Custom TCP Rule	TCP	443	4.5.77.88/32	
Custom TCP Rule	TCP	443	9.8.99.88/32	
Custom TCP Rule	TCP	443	77.66.1.2/32	
Custom TCP Rule	TCP	443	5.4.3.3/32	

**Hint:** A zombie security group is a security group that has a permissive rule but has no instances tied to it!

**Exercise Complete!** You have now found your zombie policy!

## Exercise 2.2: Identify an exposed internal asset in the AWS environment - Part 1



## Dome9 Lab Guide

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public
appserver2	i-007936eda68ce9980	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-54-160-12-37.com...	54.160.12.3
agentservice1	i-023aa5ba3311bc0ae	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-54-90-180-225.co...	54.90.180.2
DB1	i-0332d2dfcbde92aea	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-55-112-2.comp...	52.55.112.2
webapp1	i-040ff87d30686aabd	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-34-235-163-45.co...	34.235.163.4
appserver1	i-04733d6d0ed7832...	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-107-23-207-78.co...	107.23.207.7
rabbitMQ	i-04a0360b33f65a436	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-90-183-107.co...	52.90.183.1
notificationr	i-04da8ba8f35d6e636	t2.nano	us-east-1b	running	2/2 checks ...	None	ec2-34-236-237-219.co...	34.236.237.2
monitoring2	i-04f821c018b2daabc	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-54-79-241.com...	52.54.79.24
webapp2	i-050dfc294f75b1f6a	t2.nano	us-east-1b	running	2/2 checks ...	None	ec2-34-201-13-61.com...	34.201.13.6
monitoring	i-07af2582b19947fd6	t2.nano	us-east-1a	running	2/2 checks ...	None	-	-
mongodb	i-0a796e390ba0900...	t2.nano	us-east-1a	running	1/2 checks ...	None	ec2-54-173-177-56.co...	54.173.177.1
bastion	i-0b0049503e8d238...	t2.nano	us-east-1c	running	2/2 checks ...	None	ec2-54-210-215-178.co...	54.210.215.1

There is an internal asset that is exposed to the public. Can you find it?

### Exercise 2.3: Identify an exposed internal asset in the AWS environment - Part 2

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public
appserver2	i-007936eda68ce9980	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-54-160-12-37.com...	54.160.12.3
agentservice1	i-023aa5ba3311bc0ae	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-54-90-180-225.co...	54.90.180.2
DB1	i-0332d2dfcbde92aea	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-55-112-2.comp...	52.55.112.2
webapp1	i-040ff87d30686aabd	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-34-235-163-45.co...	34.235.163.4
appserver1	i-04733d6d0ed7832...	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-107-23-207-78.co...	107.23.207.7
rabbitMQ	i-04a0360b33f65a436	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-90-183-107.co...	52.90.183.1
notificationr	i-04da8ba8f35d6e636	t2.nano	us-east-1b	running	2/2 checks ...	None	ec2-34-236-237-219.co...	34.236.237.2
monitoring2	i-04f821c018b2daabc	t2.nano	us-east-1a	running	2/2 checks ...	None	ec2-52-54-79-241.com...	52.54.79.24
webapp2	i-050dfc294f75b1f6a	t2.nano	us-east-1b	running	2/2 checks ...	None	ec2-34-201-13-61.com...	34.201.13.6
monitoring	i-07af2582b19947fd6	t2.nano	us-east-1a	running	2/2 checks ...	None	-	-
mongodb	i-0a796e390ba0900...	t2.nano	us-east-1a	running	1/2 checks ...	None	ec2-54-173-177-56.co...	54.173.177.1
bastion	i-0b0049503e8d238...	t2.nano	us-east-1c	running	2/2 checks ...	None	ec2-54-210-215-178.co...	54.210.215.1

There is another internal asset that is exposed to the public (this one is harder to find)  
Good luck!

**Exercise Complete!** You have now found your exposed assets!

## Section Wrap Up

- Summary of misconfiguration issues
- Dome9 Clarity

# S3 Security Lab

## Overview of S3 bucket security

In this lab, you will build upon each exercise by starting from an exposed bucket and adding JSON policies to secure S3 buckets. There are endless realms of possibilities with S3, hence we will focus on a few key items in this lab.

### Exercise 3.1: S3 Access Controls (exposed ACLs and Bucket policies)

The goal of this exercise is to help you understand how to control S3 bucket access. We will focus on how to ensure specific role can interact with S3 and thereby not allowing any anonymous/outside users have access to sensitive data in the bucket. We will also look at least privilege concept, where you as an admin decide who has access (whitelist) to the most sensitive S3 operations and deny everyone else.

Navigate to the S3 console in AWS (FYI - You should see 6 buckets) We will go chronologically, starting from bucket 1

Stream Video to AWS for Analytics – Easily capture, process, and store video streams for analytics and machine learning. [Learn More »](#) [Documentation](#)

Amazon S3 [Discover the new console](#) [Quick tips](#)

Search for buckets

[+ Create bucket](#) [Delete bucket](#) [Empty bucket](#)

6 Buckets 2 Public 1 Regions [↻](#)

Bucket name	Access	Region	Date created
awsloft-s3bucket1-rohw1yk24pf	Public	US East (N. Virginia)	Aug 21, 2018 5:38:51 PM GMT-0400
awsloft-s3bucket2-si1m6gzuusrf	Public	US East (N. Virginia)	Aug 21, 2018 5:38:50 PM GMT-0400
awsloft-s3bucket3-e3vxx6gtt6uf	Not public *	US East (N. Virginia)	Aug 21, 2018 5:38:50 PM GMT-0400
awsloft-s3bucket4-1jpyh3i02a98z	Not public *	US East (N. Virginia)	Aug 21, 2018 5:38:51 PM GMT-0400
awsloft-s3bucket5-4mp0szrrf0kn	Not public *	US East (N. Virginia)	Aug 21, 2018 5:38:51 PM GMT-0400
cf-templates-ob34rlv71ld-us-east-1	Not public *	US East (N. Virginia)	Aug 19, 2018 2:03:30 PM GMT-0400

\* Objects might still be publicly accessible due to object ACLs. [Learn more](#)

[Feedback](#) [English \(US\)](#) © 2008 – 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

1. Fix <yourCFT>S3Bucket1 (this is straightforward)
2. Now let's move to <yourCFT>S3Bucket2. Write JSON policy (start from the existing configuration seen below)

Amazon S3 > demo-s3bucket2-1fras4mi5msbu

Overview Properties **Permissions** Management

Access Control List **Bucket Policy** CORS configuration

Bucket policy editor ARN: arn:aws:s3:::demo-s3bucket2-1fras4mi5msbu  
Type to add a new policy or edit an existing policy in the text area below.

Delete Cancel Save

```

1 {
2   "Version": "2012-10-17",
3   "Id": "MyPolicy",
4   "Statement": [
5     {
6       "Sid": "Put",
7       "Effect": "Allow",
8       "Principal": "*",
9       "Action": "s3:GetObject",
10      "Resource": "arn:aws:s3:::demo-s3bucket2-1fras4mi5msbu/*"
11    }
12  ]
13 }

```

Documentation Policy generator

- Access: Ensure S3 bucket is not publicly accessible
- Bucket Policy: Access Management Allow **S3-Role** to put and get objects in this bucket
- IAM policy: Modify **S3-Role** to allow put and get objects for S3 (useful when you want to set overall S3 policies at a user/role level rather than bucket)
- Bucket Policy: Least Privilege enforcement - Ensure deletion of S3 bucket can only be done based on your specific AWS account id (this is important for sensitive buckets that may have other accounts accessing it, and you want to make sure the most critical operations are whitelisted)

**Exercise Complete!** You have now ensured no buckets are publicly exposed!

### Exercise 3.2: S3 Encryption Best Practices

Explore permissions of <yourCFT>S3Bucket3. Now that this bucket is not public and has the right permission settings, let's see how we can properly encrypt S3. A key aspect of data security is protecting data-in-flight and data at rest. Best practice dictates that all data in the cloud be encrypted both at rest as well as in flight when data is read from or written to a bucket. This can be done easily using Secure Sockets Layer/Transport Layer Security (SSL/TLS). Encrypting data in flight helps protect against man-in-the-middle and sniffing attacks.

For your <yourCFT>S3Bucket3:

1. Data at rest: Turn on server-side encryption (AES-256)
2. Data in flight: Bucket policy should only allow get/Put to Amazon S3 objects only through HTTPS

**Exercise Complete!** You have now following a few S3 encryption best practices

### Exercise 3.3: S3 Logging

This exercise will focus on enabling logging on your S3 bucket. There are two types, object level and server level. Object level enables AWS CloudTrail trail to log data events for objects in an S3 bucket by using the Amazon S3 console. You can use it to control what buckets/prefixes, objects will be audited. Server access logging provides detailed records for the requests that are made to a bucket. This is managed by S3 alone and cannot filter what events will be logged.

For your <yourCFT>S3Bucket4:

1. Enable logging server access logging

**Exercise Complete!** You have now following a few logging best practices

## Section wrap up

- Discuss challenges in continuous monitoring and assessment
- Dome9 Compliance Engine

## Offboarding

Please navigate to the Cloudformation and click the stack and delete stack. You are now done!