Configuration Manual For Research Project on Infrastructure as Code Tools for Security of Serverless Deployments.

Hrishin Suresh x23159596 Research in Computing National College of Ireland

December 11, 2024

1 Introduction

This research focuses on the creation of a pipeline for securely deploying serverless applications using Terraform which is an Infrastructure as Code tool developed by Hashicorp. The AWS services used are Lambda, S3 bucket, API Gateway and DynamoDB for the serverless application. Along with this a Jenkins pipeline has been used with a variety of security scanning tools which will be explored in the following sections.

2 System Requirements

2.1 Hardware

The hardware used for this research is a Windows 10 laptop equipped with 16GB of dual-channel DDR4 RAM, configured as two 8GB sticks. The processor is an AMD Ryzen 7 4800H which is an 8 core CPU running at 2.9GHz. An NVidia GeForce GTX 1650 is the graphics card used with 4GB of memory. Since there is not much processing happening on the local machine, any decent computer with atleast 8GB of RAM and storage to have Docker and Jenkins running in the background will work just as well.

2.2 Software

All the code for this research was done on VSCode with a local Jenkins server running on port 8080 for deployment of the pipeline. Docker Desktop is also running in the background to create a docker image of the IaC files. A zip utility like 7zip is also required to zip the python files for uploading to Lambda. There is no other software installed for this research. Everything else that was used is cloud based like AWS services and Datadog and none of these require any additional downloads. Jenkins, however, requires the installation of multiple plugins for integrating security tools into the pipeline.

2.3 Tools and Libraries

AWS cli - v2.18.3 or higher Terraform - v1.9.7 Boto3 - v1.35.44 or higher

3 Configuration

The code can be written in any code editor, but VS code has been used due to its widespread support for extensions and plugins. Once written, the code can be uploaded to Github.

Once the code has been successfully pushed to Git, we can check the output of the "terraform plan" command to see what services will be created, and whether there are any errors before deployment.

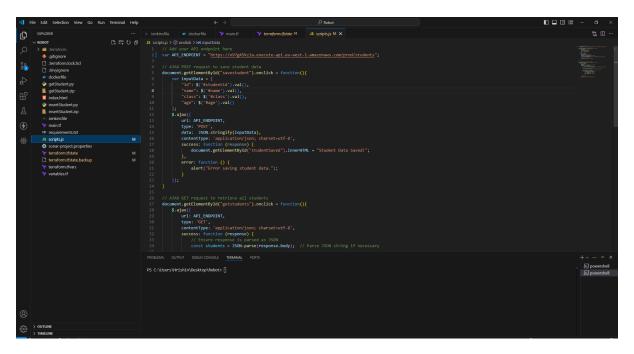


Figure 1: Code written in VSCode

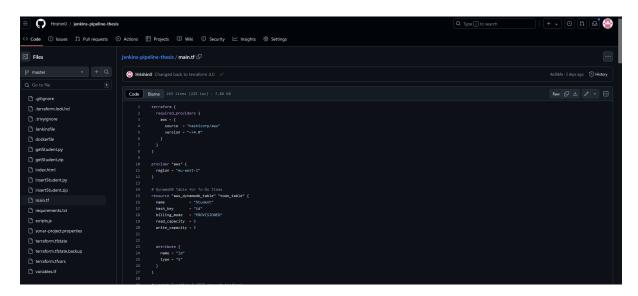


Figure 2: Code as seen on GitHub

```
# aws_lambda_permission.apigw_lambdaget will be created
+ resource "aws_lambda_permission" "apigw_lambdaget" {
                          = "lambda:InvokeFunction"
= "getStudent"
    + action
    + function_name
                          = (known after apply)
   + id
   + principal
                          = "apigateway.amazonaws.com"
                          = (known after apply)
    + source_arn
   + statement id
                          = "AllowExecutionFromAPIGatewayGET"
    + statement_id_prefix = (known after apply)
  }
# aws_lambda_permission.apigw_lambdapost will be created
+ resource "aws_lambda_permission" "apigw_lambdapost" {
                         = "lambda:InvokeFunction"
    + action
                          = "insertStudent"
   + function_name
                          = (known after apply)
= "apigateway.amazonaws.com"
   + id
   + principal
                          = (known after apply)
    + source arn
                          = "AllowExecutionFromAPIGatewayPOST"
    + statement_id
    + statement_id_prefix = (known after apply)
  }
# aws_s3_bucket.frontend_bucket will be created
+ resource "aws_s3_bucket" "frontend_bucket" {
    + acceleration_status
                                  = (known after apply)
    + acl
                                  = (known after apply)
   + arn
                                  = (known after apply)
                                  = "hrishin-test-111"
   + bucket
    + bucket domain name
                                  = (known after apply)
                                  = (known after apply)
   + bucket_prefix
   + bucket_regional_domain_name = (known after apply)
    + force_destroy
                                  = false
   + hosted zone id
                                  = (known after apply)
   + id
                                  = (known after apply)
   + object_lock_enabled
                                  = (known after apply)
                                  = (known after apply)
   + policy
   + region
                                  = (known after apply)
                                  = (known after apply)
   + request_payer
                                  = (known after apply)
    + tags all
    + website_domain
                                  = (known after apply)
    + website_endpoint
                                  = (known after apply)
```

Figure 3: TF plan output

🏘 Jenkins			Q Search (CTRL+K)	2 🗘 🚺 🛈 🕕 Hris	shin Suresh $\sim ightarrow \log$ log out
Dashboard $ ightarrow$ terraform-project $ ightarrow$ #16					
🗐 Status	🕢 Bu	ild #16 (Nov 18, 2024, 9:19:11 AM)			Keep this build forever
✓ Changes ► Console Output				$\ensuremath{\mathscr{O}}$ Add description	Started 15 days ago Took 2 min 29 sec
C Edit Build Information	\heartsuit	Build Artifacts P trivy-main-tf-report.txt 0 B 🚳 view			
Delete build '#16'	\odot	Started by user Hrishin Suresh			
Parameters Timings Git Build Data	٢	This run spent: • 90 ms waiting: • 2 min 29 sec build duration:			
Y Pipeline Overview		2 min 29 sec total from scheduled to completion.			
> Pipeline Console	🚯 git	Revision: 817430199657a4ffaaa4740246ee143820a451fb Repository: https://github.com/Hrishin0/jenkins-pipeline-thesis.git			
🛱 Replay		refs/remotes/origin/master			
Pipeline Steps					
Workspaces					
 ← Previous Build → Next Build 					
•					

Figure 4: Trivy report on Jenkins dashboard

C:\ProgramData\Jenkins\	.jenkin	s\workspace\ternaform-project>trivy imageseverity HIGH,CRITICALexit-code 1format table -o trivy-docker-image-report.txt iac-scanning:latest
2024-12-01T19:26:06Z	INFO	[vuln] Vulnerability scanning is enabled
2024-12-01T19:26:06Z	INFO	[secret] Secret scanning is enabled
2024-12-01T19:26:06Z	INFO	[secret] If your scanning is slow, please try 'scanners vuln' to disable secret scanning
2024-12-01T19:26:06Z	INFO	[secret] Please see also https://aquasecurity.github.io/trivy/v0.57/docs/scanner/secret#recommendation for faster secret detection
2024-12-01T19:26:12Z	INFO	[python] License acquired from METADATA classifiers may be subject to additional terms name="pip" version="23.0.1"
2024-12-01T19:26:12Z	INFO	[python] License acquired from METADATA classifiers may be subject to additional terms name="setuptools" version="58.1.0"
2024-12-01T19:26:15Z	INFO	Detected OS family="debian" version="12.8"
2024-12-01T19:26:15Z	INFO	[debian] Detecting vulnerabilities os_version="12" pkg_num=105
2024-12-01T19:26:15Z	INFO	Number of language-specific files num=2
2024-12-01T19:26:15Z	INFO	[gobinary] Detecting vulnerabilities
2024-12-01T19:26:15Z	INFO	[python-pkg] Detecting vulnerabilities
2024-12-01T19:26:15Z	WARN	Using severities from other vendors for some vulnerabilities. Read https://aquasecurity.github.io/trivy/v0.57/docs/scanner/vulnerability8severity-
selection for details.		
2024-12-01T19:26:15Z	INFO	Table result includes only package filenames. Use 'format ison' option to get the full path to the package file.

Figure 5: Trivy finding vulnerabilities

Next, we can create and run the pipeline on Jenkins which has been hosted on the localhost port 8080. The pipeline can be run with autoApprove enabled for convenience, but having it disabled is recommended for security. Once run, if the Trivy stage has successfully executed, the report can be seen from the Jenkins dashboard for that build. The pipeline will fail if any vulnerabilities are detected by Trivy

The "terraform state list" command can be executed to show all the aws services that have been created.

The pipeline will pass the sonarcloud analysis stage only if the sonarcloud quality gate has passed. To see this, we can refer to the dashboard on the sonarcloud website.

Once the services have been deployed to AWS, go into the API gateway that was deployed, click Stages in the left side bar and copy the invoke url for the /students tab. Once copied, paste this into the scripts.js file under the API ENDPOINT and upload this new scripts.js into the S3 bucket.

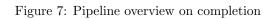
Next, go to API Gateway and configure method responses, as well as set the integration response on POST route to normal. Default setting will be "Lambda proxy integration". Once this is done, enable CORS for the /students route and deploy the API again and the application will work as expected. The application can be accessed via the s3 bucket url which is also given as the output of the pipeline.

Datadog can be set up as per necessity by following the normal instructions on the datadog website for AWS Lambda.

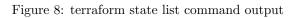
ERROR: script returned exit code 1 Finished: FAILURE

Figure 6: Pipeline failing when Trivy detects vulnerabilities

Source of the second se		Console Configure
Pipeline Start Checkout SCM checkout Trivy Terraform S SonarCloud Ana Plan Approval Apply End	2	Details ▲ Manually run by Hrishin Suresh ③ Started 16 days ago ☑ Queued 35 ms ④ Took 2 min 29 sec



	PS C:\Users\Hrishin\Desktop\Robot> terraform state list
•	aws_api_gateway_deployment.deploy
	aws_api_gateway_integration.cors_integration
	aws_api_gateway_integration.getIntegration
	aws_api_gateway_integration.postIntegration
	<pre>aws_api_gateway_integration_response.cors_integration_response</pre>
	aws_api_gateway_method.cors_options
	aws_api_gateway_method.get_student_method
	aws_api_gateway_method.post_student_method
	<pre>aws_api_gateway_method_response.cors_method_response</pre>
	aws_api_gateway_resource.student_resource
	aws_api_gateway_rest_api.student_api
	aws_api_gateway_rest_api.student_api
	aws_api_gateway_stage.prod
	aws_dynamodb_table.todo_table
	aws_iam_role.lambda_role
	aws_iam_role_policy.lambda_policy
	aws_lambda_function.lambda_func_1
	aws_lambda_function.lambda_func_2
	aws_lambda_permission.apigw_lambdaget
	aws_lambda_permission.apigw_lambdapost
	aws_s3_bucket.frontend_bucket
	aws_s3_bucket_object.index_html
0	
	PS C:\Users\Hrishin\Desktop\Robot>



SonarQube	My Projects My Issues Explore Q				f) 🛛 + 🐹		
jenkins-pipeline-thesis	Hrishin0 > jenkins-pipeline-thesis > I master d Summary Issues Security Hotspots Measures Code Activity				A The last analysis has v	varnings. <u>See details</u>		
88 Overview	Main Branch Summary 404 Lines of Code 🐠							
1 Main Branch								
11 Pull Requests 0	Struggling with too many issues? Discover 'Clean as ' Learn how to improve your code base by cleaning only new co							
P Branches 1	Take the Tour Not now							
	Quality Gate: <u>Sonar way</u> Passed New Code Overall Code	Passed						
	New code Since 16 days ago	New code Since 16 days ago						
	New Issues		Accepted Issues					
	0		0					
Information	No conditions set		Valid issues that were not fi					
3 Administration	Coverage	Duplications		Security Hotspots				
← Collapse	There is not enough lines to compute coverage	0.0%		0				

Figure 9: SonarCloud analysis

🙋 EC2 🛛 🚳 Elastic Beanstalk 🖉 Cloud9 🗟 C	iodePipeline 🔉 Lambda 🔞 CloudFront 📴 53 😰 DynamoDB 🚦	d API Gateway	
API Gateway > APIs > StudentAPI (o	97g45kzia) > Stages		0 0
API Gateway <	Stages	Stage actions v Create stage	
APIs Custom domain names Updated Domain name access associations New VPC links	prod / / students GET	Method overrides Create override By default, methods inherit stage-level settings. To customize settings for a method, configure method overrides. Image: Create override Image: Create override Image: Create override Image: Create override	
API: StudentAPI Resources Stages	OPTIONS POST	Imoke URL Thtps://997g45kzia.execute-api.eu-west-1.amazonaws.com/prod/students	
Authorizers Gateway responses Models			
Resource policy Documentation Dashboard API settings	4		

Figure 10: Invoke Url on API Gateway

JS sci	ripts.js > 🕅 o	onclick > 囪 in	putData			
			dpoint here			
	var API_	ENDPOINT =	"https://o97g45kzia.exe	ecute-api.eu-west	t-1.amazonaws.com	<pre>i/prod/students";</pre>

Figure 11: API Endpoint

API Gateway > APIs > Resources - StudentAF					
PI Gateway <	 Successfully enabled CORS Details 				
ustom domain names Updated					
omain name access associations New	Resources				API actions 🔻 Deploy AF
PC links					
	Create resource	Resource details		Delete Update	documentation Enable CORS
I: StudentAPI		Path		Resource ID	
ources	□ / □ /students	/students		wbgna2	
ges	GET				
iorizers	OPTIONS	Mathada (7)			Delete Create method
eway responses	POST	Methods (3)			Create method
lels		Method type	 Integration type 	▼ Authorization	▼ API key ▼
urce policy		O GET	Lambda	None	Not required
imentation					
board			Mock	None	Not required
settings		O POST	Lambda	None	Not required
e plans					
eys .					
it certificates					
ings					

Figure 12: Enable CORS

☆ AWS Lambda ~ (************************************	rsion 👻			1h Past 1 Hour	☆ Share ✿ Configure
Want real-time metrics on AWS Lambda errors, cold starts, memory usage, and cost? Enable <u>enhanced Lambda metrics</u> and browse to the AWS Lambda (Enhanced Metrics) default disabboard. See <u>Lambda integration documentation</u> . And <u>learn more</u> about the <u>resource</u> and <u>executedVersion</u> tags.		Invocations (No data)			and re-run lambda functions using App B Learn More + Connect Data
Invocations Top invoked functions No Data	Execution Time Slowest functions No Data	Errors Functions with most errors No Data		data.	
Invocations by function 150 100 0	p95 execution time by function	Errors by function	23:00 23:15 23:30		

Figure 13: Datadog Dashboard