

## Infrastructure Scanning and Vault Security

Objectives: To implement two pipelines –

1. Infrastructure scanning i.e scanning of terraform/IAC code files for vulnerabilities.
2. Incorporate Hashicorp Vault to secure server credentials which are to be retrieved only during deployment.

Target repositories:

1. <https://github.com/Msocial123/EverNorth-Terraform-Project>
2. [https://github.com/Msocial123/fss-Retail-App\\_kubernetes](https://github.com/Msocial123/fss-Retail-App_kubernetes)

1.The repositories were forked and cloned. An EC2 server instance was set up on AWS.

**Instance summary for i-09aecf79db0bea5a2 (Dev-SecOps-Jenkins-Server)** Info Connect Instance state ▼ Actions ▼

Updated less than a minute ago

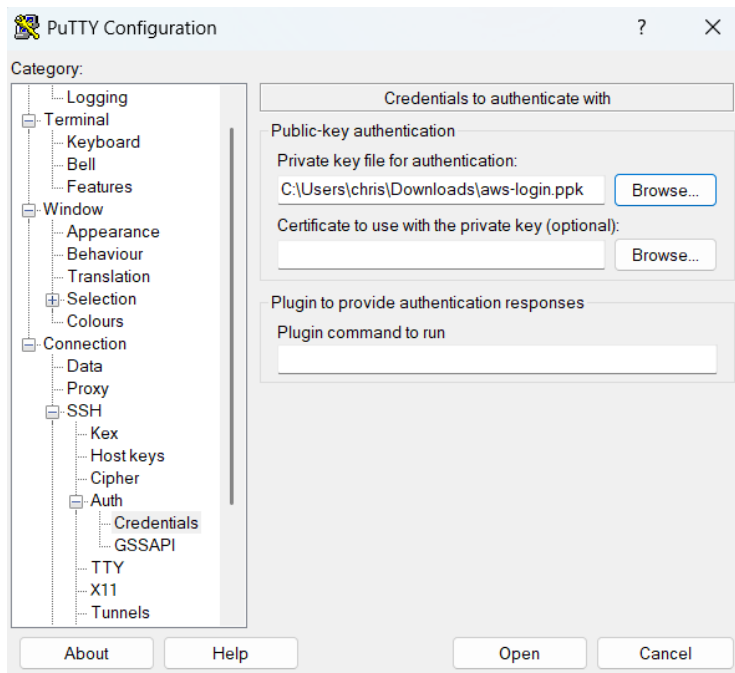
<b>Instance ID</b> i-09aecf79db0bea5a2	<b>Public IPv4 address</b> 3.34.106.82   <a href="#">open address</a>	<b>Private IPv4 addresses</b> 172.31.41.134
<b>IPv6 address</b> –	<b>Instance state</b> Running	<b>Public IPv4 DNS</b> ec2-3-34-106-82.ap-northeast-2.compute.amazonaws.com   <a href="#">open address</a>
<b>Hostname type</b> IP name: ip-172-31-41-134.ap-northeast-2.compute.internal	<b>Private IP DNS name (IPv4 only)</b> ip-172-31-41-134.ap-northeast-2.compute.internal	<b>Elastic IP addresses</b> 3.34.106.82 (Dev-SecOps-Server) [Public IP]
<b>Answer private resource DNS name</b> IPv4 (A)	<b>Instance type</b> t2.medium	

2.An SSH connection was established to the instance via PuTTY

-Download the ppk file from AWS

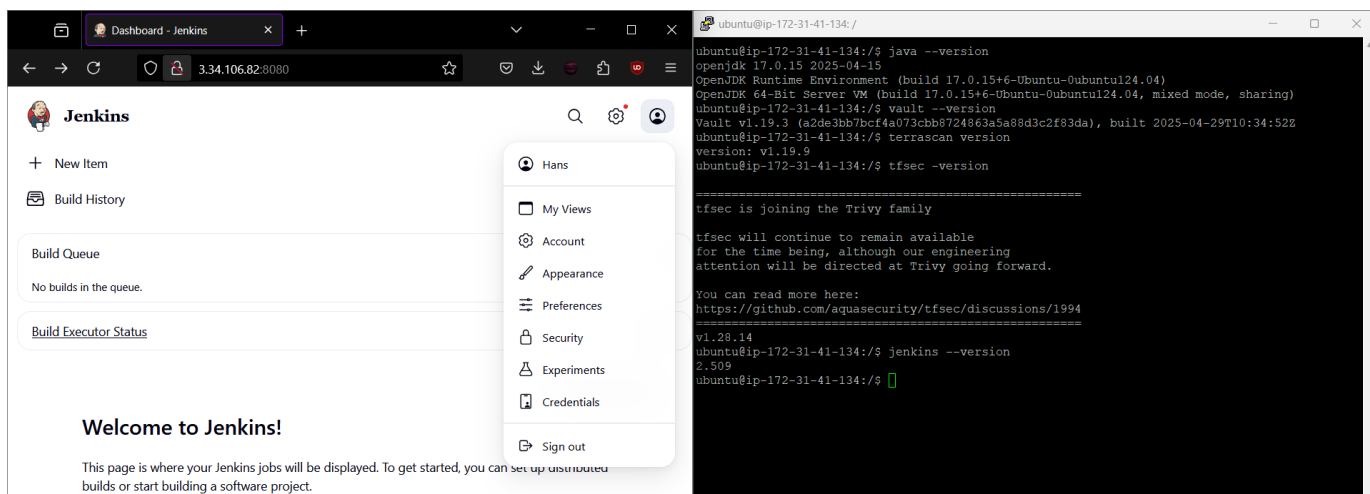
-Set host as <username>@<instance-ip>

-Add the ppk file in the PuTTY configuration terminal under credentials



3. Downloaded the required tools and software on the server.

-Jenkins, Java, Vault, Terrascan, Tfsec, NodeJS, PM2



```
ubuntu@ip-172-31-41-134:~$ pm2 --version
6.0.5
ubuntu@ip-172-31-41-134:~$ npm -v
10.8.2
ubuntu@ip-172-31-41-134:~$ node -v
v18.20.8
ubuntu@ip-172-31-41-134:~$
```

#### 4. Export vault credentials and set-up Vault server

```
ubuntu@ip-172-31-41-134:/$ export VAULT_ADDR='http://127.0.0.1:8200'
export VAULT_TOKEN='hvs.UludKY9MQEPsgkFu6XvGppGs'
ubuntu@ip-172-31-41-134:/$ vault status
Key          Value
---          -
Seal Type    shamir
Initialized  true
Sealed       false
Total Shares 1
Threshold    1
Version      1.19.3
Build Date   2025-04-29T10:34:52Z
Storage Type  inmem
Cluster Name  vault-cluster-6320fcb5
Cluster ID    542997fb-b13d-395e-d24b-187915e0481b
HA Enabled    false
ubuntu@ip-172-31-41-134:/$
```

#### 5. Store application server credentials in the vault

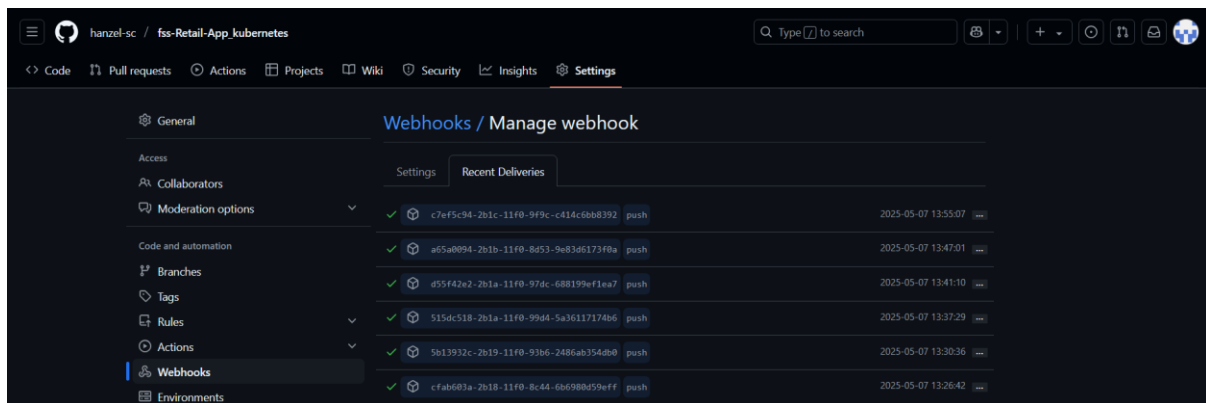
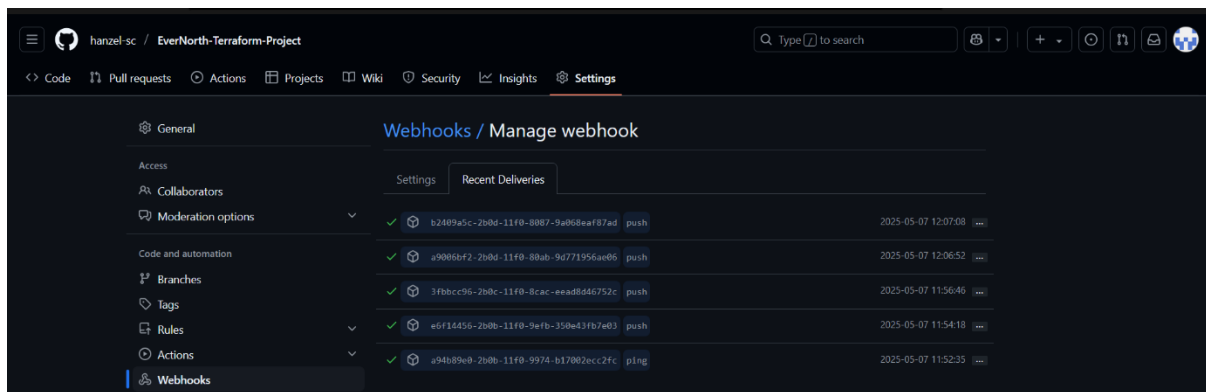
```
ubuntu@ip-172-31-41-134:~$ export VAULT_ADDR='http://127.0.0.1:8200'
export VAULT_TOKEN='hvs.UludKY9MQEPsgkFu6XvGppGs'
ubuntu@ip-172-31-41-134:~$ vault kv put secret/fss-retail-app EMAIL_USER="chagantyteja2502@gmail.com" EMAIL_PASS="yxqo bjuk rdnt alzp" PORT="3130" SESSION_SECRET="1234" MONGO_URI="mongodb://localhost:27017/myDatabase"
===== Secret Path =====
secret/data/fss-retail-app

===== Metadata =====
Key          Value
---          -
created_time  2025-05-07T07:07:16.946078739Z
custom_metadata <nil>
deletion_time n/a
destroyed     false
version       1
ubuntu@ip-172-31-41-134:~$ vault kv get secret/fss-retail-app
===== Secret Path =====
secret/data/fss-retail-app

===== Metadata =====
Key          Value
---          -
created_time  2025-05-07T07:07:16.946078739Z
custom_metadata <nil>
deletion_time n/a
destroyed     false
version       1

===== Data =====
Key          Value
---          -
EMAIL_PASS    yxoq bjuk rdnt alzp
EMAIL_USER    chagantyteja2502@gmail.com
MONGO_URI     mongodb://localhost:27017/myDatabase
PORT          3130
SESSION_SECRET 1234
ubuntu@ip-172-31-41-134:~$
```

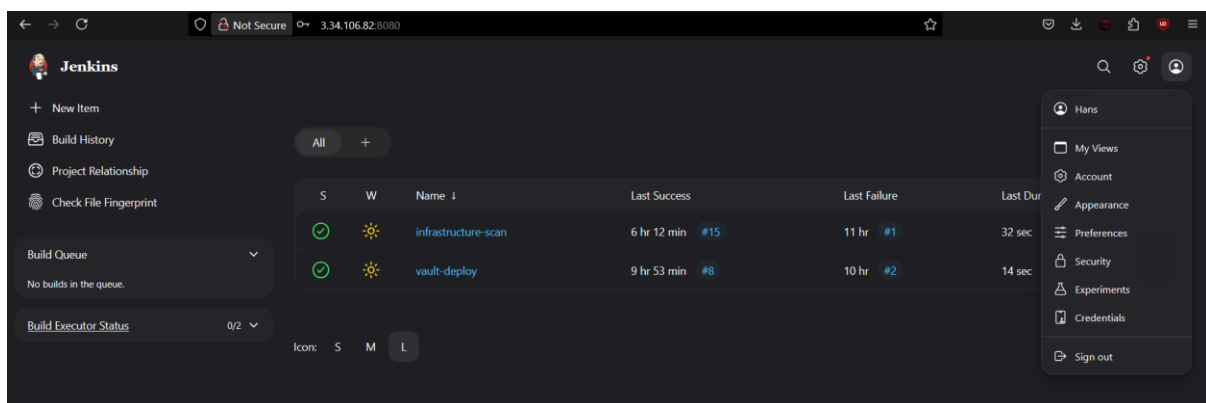
## 6. Set up two GitHub webhooks – one for each repository



## 7. After installing the pre-requisites start the Jenkins Server using :

- sudo systemctl start jenkins

Configure Jenkins with the necessary installations and create two pipeline jobs, to be triggered by Git SCM polling



8. Implement pipeline scripts on both repositories.

a. Infrastructure scanning pipeline

```
pipeline {
  agent any

  environment {
    TERRASCAN = '/usr/local/bin/terrascan'
    TFSEC = '/usr/local/bin/tfsec'
  }

  stages {
    stage('Checkout') {
      steps {
        git branch: 'test', url: 'https://github.com/hanzel-sc/EverNorth-Terraform-Project'
      }
    }

    stage('Run Pentest Scan') {
      steps {
        sh 'chmod +x scan.sh'
        sh './scan.sh'
      }
    }

    stage('Publish Reports') {
      steps {
        archiveArtifacts artifacts: 'reports/*.json', fingerprint: true
        archiveArtifacts artifacts: 'reports/*.txt', fingerprint: true
        archiveArtifacts artifacts: 'logfile/*.txt', fingerprint: true
      }
    }
  }
}
```

```
    }  
  }  
}  
}
```

## b. Vault security pipeline

```
pipeline {  
  agent any  
  
  environment {  
  
    EMAIL_PASS="  
    EMAIL_USER="  
    PORT="  
    MONGODB_URI="  
    SESSION_SECRET="  
  
  }  
  
  tools {  
  
    nodejs "NodeJS"  
  
  }  
  
  stages {
```

```
stage('Git checkout') {
    steps {
        git url: 'https://github.com/hanzel-sc/fss-Retail-App_kubernetes.git', branch:
'test'
    }
}

stage('Fetch Secrets from Vault') {
    steps {
        withVault(
            configuration: [vaultUrl: 'http://127.0.0.1:8200', vaultCredentialId: 'vault-
token'],
            vaultSecrets: [[
                path: 'secret/fss-retail-app',
                secretValues: [
                    [envVar: 'EMAIL_PASS', vaultKey: 'EMAIL_PASS'],
                    [envVar: 'EMAIL_USER', vaultKey: 'EMAIL_USER'],
                    [envVar: 'PORT', vaultKey: 'PORT'],
                    [envVar: 'MONGODB_URI', vaultKey: 'MONGODB_URI'],
                    [envVar: 'SESSION_SECRET', vaultKey: 'SESSION_SECRET']
                ]
            ]]
        ){
            echo "Secrets retrieved successfully."
            echo "Email password is: ${env.EMAIL_PASS.take(2)}****"
        }
    }
}
```

```
stage('Install Dependencies'){
    steps {
        script {
            sh 'npm install'
        }
    }
}

stage('Deploy'){
    steps {
        script {
            sh 'pm2 delete all || true'
            sh "pm2 start server.js --name nodejs-backend"
            echo "Server is up and running"
        }
    }
}

post {
    failure {
        echo 'Build failed! Check Jenkins logs.'
    }
    success {
        echo 'Successfully deployed NodeJS application with secrets from Vault!!'
    }
}
}
```



9. Implement a simple shell script to run the pentest scans (scan.sh file). Optionally, using grep and awk – log critical errors into log files for review.

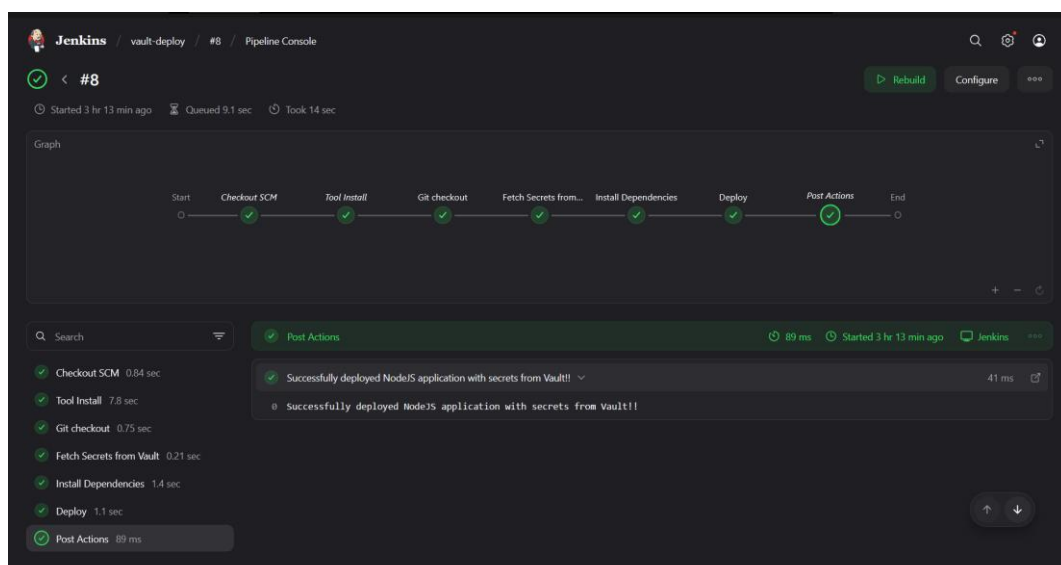
```

Jenkinsfile  scan.sh  X
EverNorth-Terraform-Project > $ scan.sh
1  #!/bin/bash
2
3  rm -rf reports
4  rm -rf logfile
5  mkdir -p reports
6  mkdir -p logfile
7
8  #timestamp=$(date +"%Y%m%d%H%M%S") //attaching timestamp if needed.
9
10 # Running the terrascan scan
11 /usr/local/bin/terrascan scan -t aws -d . -o json > reports/terrascan_report.json
12 /usr/local/bin/terrascan scan -t aws -d . -o human > reports/terrascan_report.txt
13
14 #Running the TFsec scan
15 /usr/local/bin/tfsec . --format text > reports/tfsec_report.txt
16 /usr/local/bin/tfsec . --format json > reports/tfsec_report.json
17
18 #Logging the critical and severe errors
19 awk '
20 BEGIN { RS=""; FS="\n" }
21 /Severity[[:space:]]*:[[:space:]]*HIGH/ { print "\n---\n" $0 }
22 ' reports/terrascan_report.txt > logfile/terrascan_issues.txt
23
24 grep -i -E 'critical|high|severe' reports/tfsec_report.txt > logfile/tfsec_issues.txt
25
26
27 echo "Terrascan and tfsec reports saved to $(pwd)/reports/"
28 echo "Critical issues are logged in $(pwd)/logfile/"
29

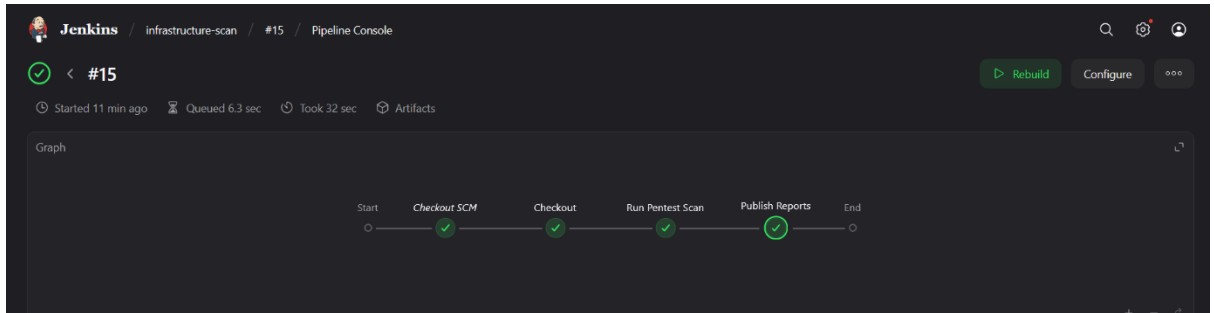
```

9. Push the code changes onto GitHub to trigger the pipelines. Ensure all stages of the pipeline are completed successfully.

- Vault Security pipeline with application deployment



- Infrastructure code scanning



10. Observer the Build artifacts which contain the scan reports and logfiles containing the critical issues

The screenshot shows the Jenkins Build Artifacts view for a job named 'infrastructure-scan'. The build is #15, dated May 7, 2025, 12:05:52 PM, and is in a 'Completed' state, indicated by a green checkmark. The left sidebar contains a list of links: Status, Changes, Console Output, Edit Build Information, Delete build '#15', Polling Log, Timings, Git Build Data, See Fingerprints, Pipeline Console, Restart from Stage, Replay, Pipeline Steps, Workspaces, and Previous Build. The main content area displays the following information:

- Build Artifacts:** A table listing the artifacts and their sizes, with a 'view' link for each.

Artifact	Size	View
terrascan_issues.txt	6.32 KiB	<a href="#">view</a>
tfsec_issues.txt	3.46 KiB	<a href="#">view</a>
terrascan_report.json	18.65 KiB	<a href="#">view</a>
terrascan_report.txt	16.53 KiB	<a href="#">view</a>
tfsec_report.json	50.00 KiB	<a href="#">view</a>
tfsec_report.txt	83.42 KiB	<a href="#">view</a>
- Started by:** Started by GitHub push by hanzel-sc
- This run spent:**
  - 6.3 sec waiting;
  - 32 sec build duration;
  - 39 sec total from scheduled to completion.
- Revision:** 6e08fd300c9fe78dde22b3310844a0d134eb09c2
- Repository:** <https://github.com/hanzel-sc/EverNorth-Terraform-Project>
  - refs/remotes/origin/test
- Changes:**
  - 1. Configured report formats to text files. Further logged critical errors into a log file ([details](#) / [githubweb](#))

- Analyse the custom log files to review the critical issues

```

← → ↺ Not Secure 3.34.106.82:8080/job/infrastructure-scan/15/artifact/logfile/tfsec_issues.txt
Result #1 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #2 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #3 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #4 CRITICAL Security group rule allows ingress from public internet.
Result #5 CRITICAL Security group rule allows ingress from public internet.
Result #6 CRITICAL Security group rule allows ingress from public internet.
Result #7 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #8 CRITICAL Security group rule allows ingress from public internet.
Result #9 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #10 CRITICAL Security group rule allows egress to multiple public internet addresses.
Result #11 HIGH Root block device is not encrypted.
Result #12 HIGH Instance does not have storage encryption enabled.
Results #13-14 HIGH Subnet associates public IP address. (2 similar results)
Result #15 HIGH IAM policy document uses sensitive action 'ec2:AuthorizeSecurityGroupIngress' on wildcarded resource '*'
Result #16 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:CreateLoadBalancer' on wildcarded resource '*'
Result #17 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:CreateListener' on wildcarded resource '*'
Results #18-20 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:AddTags' on wildcarded resource 'arn:aws:elasticloadbalancing:*:*:targetgroup/*/*' (3 similar results)
Result #21-24 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:AddTags' on wildcarded resource 'arn:aws:elasticloadbalancing:*:*:listener/net/*/*/*' (4 similar results)
Result #25 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:ModifyLoadBalancerAttributes' on wildcarded resource '*'
Result #26 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:RegisterTargets' on wildcarded resource 'arn:aws:elasticloadbalancing:*:*:targetgroup/*/*'
Result #27 HIGH IAM policy document uses sensitive action 'elasticloadbalancing:SetWebAcl' on wildcarded resource '*'
Result #28 HIGH IAM policy document uses sensitive action 'iam:CreateServiceLinkedRole' on wildcarded resource '*'
Result #29 HIGH IAM policy document uses sensitive action 'cognito-idp:DescribeUserPoolClient' on wildcarded resource '*'
Result #30 HIGH IAM policy document uses sensitive action 'ec2:AuthorizeSecurityGroupIngress' on wildcarded resource '*'
Result #31 HIGH IAM policy document uses sensitive action 'ec2:CreateSecurityGroup' on wildcarded resource '*'
Result #32 HIGH IAM policy document uses sensitive action 'ec2:CreateTags' on wildcarded resource 'arn:aws:ec2:*:*:security-group/*'
Result #33 HIGH IAM policy document uses sensitive action 'ec2:CreateTags' on wildcarded resource 'arn:aws:ec2:*:*:security-group/*'
Result #34 HIGH IAM policy document uses sensitive action 'ec2:GetInstanceTypesFromInstanceRequirements' on wildcarded resource '*'
Result #35 HIGH IAM policy document uses sensitive action 'autoscaling:SetDesiredCapacity' on wildcarded resource '*'
Result #36 HIGH Instance does not require IMDS access to require a token
Result #37 HIGH Root block device is not encrypted.
Result #38 HIGH Instance does not require IMDS access to require a token
Result #39 HIGH Root block device is not encrypted.
critical      10
high          29

```

```

← → ↺ Not Secure 3.34.106.82:8080/job/infrastructure-scan/lastSuccessfulBuild/artifact/logfile/terrascan_issues.txt
Description : Security Groups - Unrestricted Specific Ports - Known internal web port (TCP,8080)
File : 22_security_group_jenkins.tf
Module Name : root
Plan Root : ./
Line : 2
Severity : HIGH

-----

Description : Security Groups - Unrestricted Specific Ports - Hadoop Name Node (TCP,9000)
File : 22_security_group_jenkins.tf
Module Name : root
Plan Root : ./
Line : 2
Severity : HIGH

-----

Description : Ensure that detailed monitoring is enabled for EC2 instances.
File : git::https://github.com/terraform-aws-modules/terraform-aws-ec2-instance?ref=5b17f94c354eb3fa2b3bc435c861b916c9668e05/main.tf
Module Name : ec2_bastion_instance
Plan Root : ./
Line : 199
Severity : HIGH

-----

Description : Ensure that detailed monitoring is enabled for EC2 instances.
File : 23_jenkins_server.tf
Module Name : root
Plan Root : ./
Line : 2
Severity : HIGH

-----

Description : Ensure that detailed monitoring is enabled for EC2 instances.
File : 23_jenkins_server.tf
Module Name : root
Plan Root : ./
Line : 29
Severity : HIGH

-----

Description : Ensure that detailed monitoring is enabled for EC2 instances.
File : git::https://github.com/terraform-aws-modules/terraform-aws-ec2-instance?ref=5b17f94c354eb3fa2b3bc435c861b916c9668e05/main.tf
Module Name : ec2_bastion_instance
Plan Root : ./
Line : 21
Severity : HIGH

-----

```

Links to the forked, local repositories:

1. <https://github.com/hanzel-sc/EverNorth-Terraform-Project>
2. [https://github.com/hanzel-sc/fss-Retail-App\\_kubernetes](https://github.com/hanzel-sc/fss-Retail-App_kubernetes)