Pentesting Infrastructure

Category	IaC
Date	@May 5, 2025
Last Updated	@May 5, 2025
Status	Completed
Tags	Best Practices
Documentation	Completed

Pentesting Terraform infrastructure using Tfsec and Terrascan

-These two tool are efficient in performing static code analysis on Terraform (Iac) files.

1. Downloading the tools on the server (Cloud instance or VM)

-Terrascan

```
root@Zwi-VirtualBox:/home/zwi# curl -L -o terrascan.tar.gz https://github.com/tenable/terrascan/releases/download/
v1.19.9/terrascan_1.19.9_Linux_x86_64.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 24.2M  100 24.2M    0     0  4038k      0  0:00:06  0:00:06 --:--:-- 6024k
root@Zwi-VirtualBox:/home/zwi# tar -xvzf terrascan.tar.gz
CHANGELOG.md
LICENSE
README.md
terrascan
root@Zwi-VirtualBox:/home/zwi# sudo mv terrascan /usr/local/bin/
root@Zwi-VirtualBox:/home/zwi# terrascan version
version: v1.19.9
```

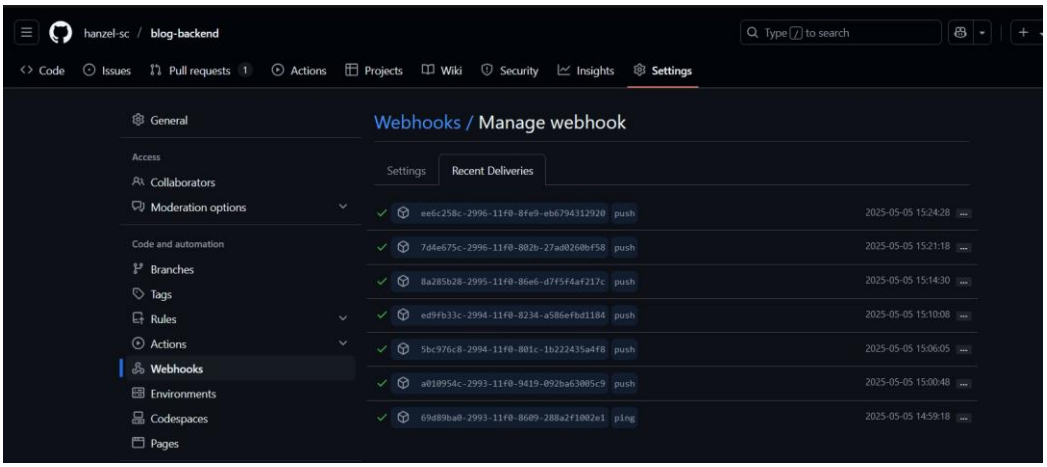
-Tfsec

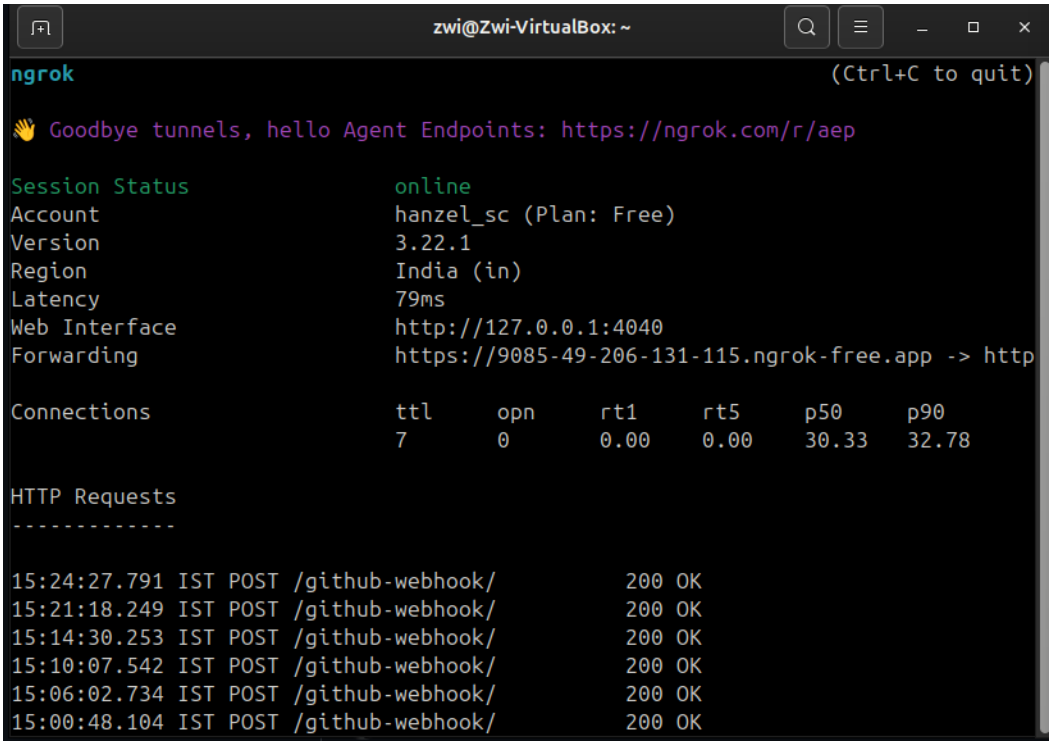
```
root@Zwi-VirtualBox:/home/zwi# curl -s https://raw.githubusercontent.com/aquasecurity/tfsec/master/scripts/install
_linux.sh | bash
arch=amd64
remote_filename=tfsec-linux-amd64
local_filename=tfsec
checkgen_filename=tfsec-checkgen-linux-amd64


====================================================
Looking up the latest version...
Downloading tfsec v1.28.14
Downloading tfsec-linux-amd64...
Downloaded file "tfsec-linux-amd64" successfully
Downloading tfsec-checkgen-linux-amd64...
Downloaded file "tfsec-checkgen-linux-amd64" successfully
Downloading tfsec_checksums.txt...
Downloaded file "tfsec_checksums.txt" successfully
Checksum verified successfully


====================================================
Installing /tmp/tfsec.I1dknlKd0p/tfsec to /usr/local/bin/...
'/tmp/tfsec.I1dknlKd0p/tfsec' -> '/usr/local/bin/tfsec'
Cleaning downloaded files...


====================================================

====================================================
tfsec is joining the Trivy family

tfsec will continue to remain available
for the time being, although our engineering
attention will be directed at Trivy going forward.

You can read more here:
https://github.com/aquasecurity/tfsec/discussions/1994
====================================================
Current tfsec version: v1.28.14
root@Zwi-VirtualBox:/home/zwi# tfsec --version
```

The two tools have been successfully downloaded:

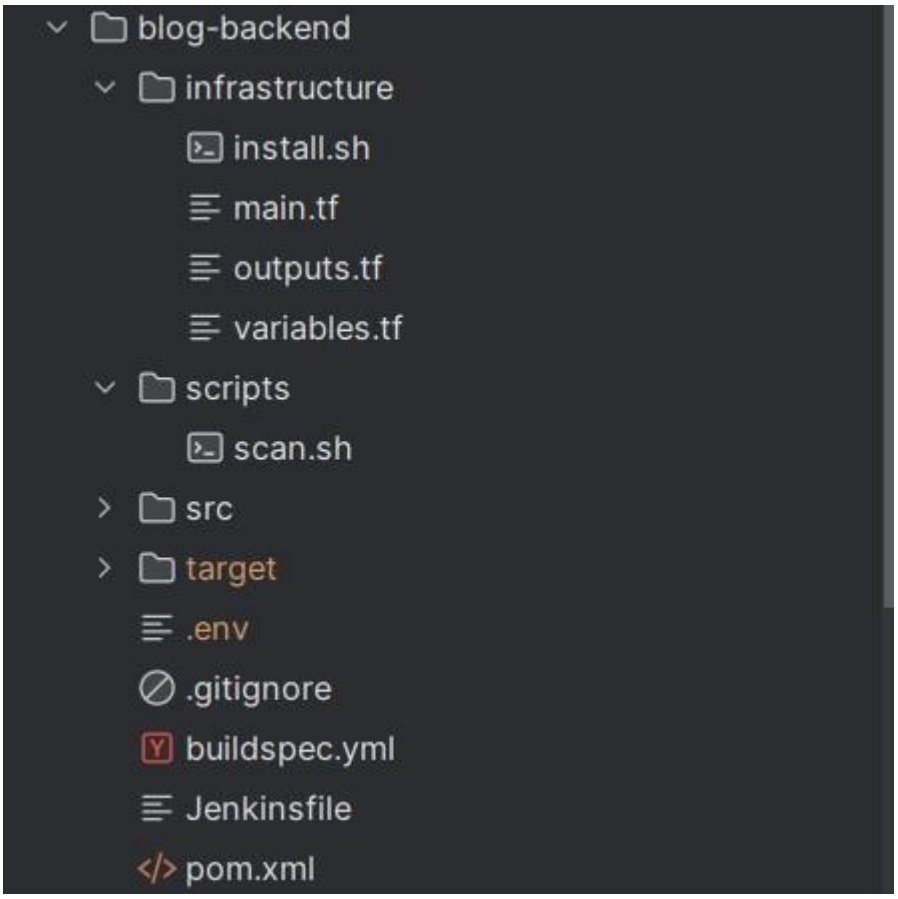
```
root@Zwi-VirtualBox:/home/zwi# cd /usr/local/bin
root@Zwi-VirtualBox:/usr/local/bin# ls
terrascan  tfsec
```

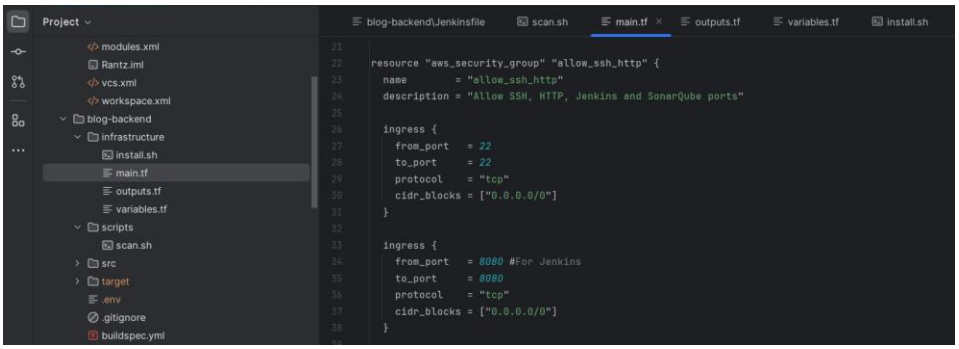
2.Using ngrok to expose the VM's Jenkins Server. Configure GitHub webhook to trigger Jenkins pipeline on code push





3.Ensuring that the terraform (infrastructure files) are placed in the root directory and have some basic terraform code





4.Writing a simple shell script file to run the two scans :

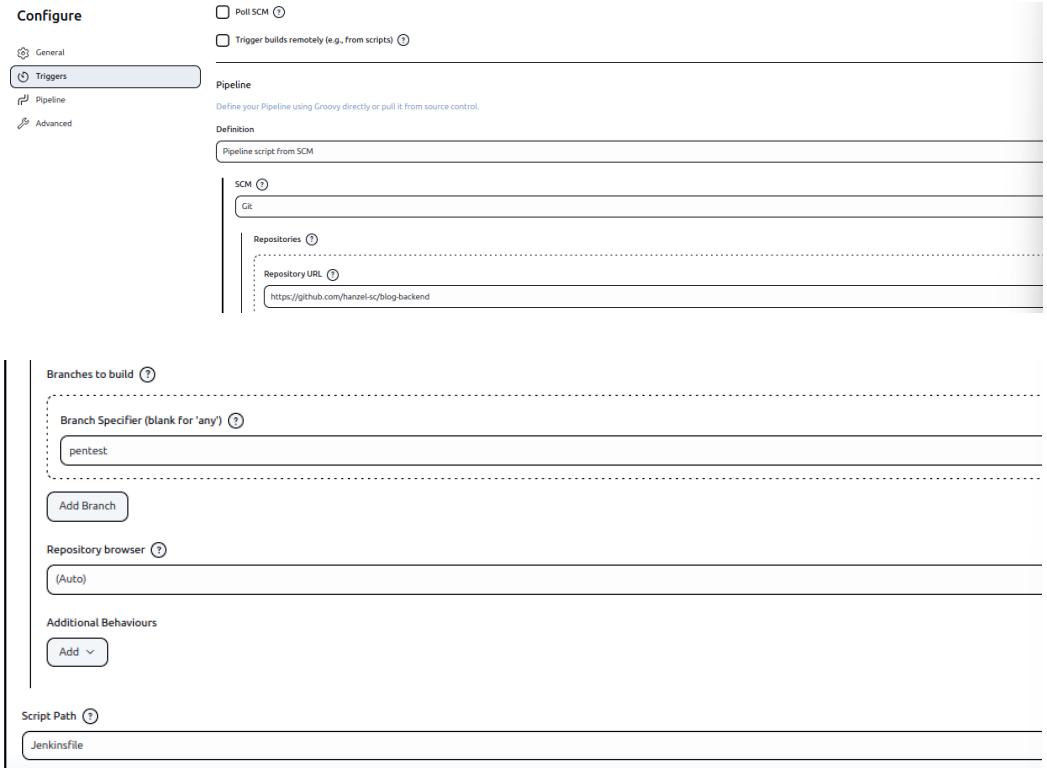
```
#!/bin/bash

mkdir -p reports

# Running terrascan scan
/usr/local/bin/terrascan scan -t aws -d infrastructure/ -o json > reports/terrascan_report.json

#Running TFsec scan
/usr/local/bin/tfsec infrastructure/ --format json > reports/tfsec_report.json

echo "Terrascan and tfsec reports saved to $(pwd)/reports/"
```

5.Implemented a Jenkins pipeline to automate build and pentest process as well as publishing the reports as artifacts on the Jenkins Server

```
        stage('Build') {
            steps {
                script {
                    if (isUnix()) {
                        // Linux commands
                        sh "${MAVEN_HOME}/bin/mvn test"
                        sh "${MAVEN_HOME}/bin/mvn clean verify"
                        sh "${MAVEN_HOME}/bin/mvn clean install"
                    } else {
                        // Windows commands
                        bat "${MAVEN_HOME}\\bin\\mvn test"
                        bat "${MAVEN_HOME}\\bin\\mvn clean verify"
                        bat "${MAVEN_HOME}\\bin\\mvn clean install"
                    }
                }
            }
        }

        stage('Run Pentest Scan') {
                    steps {
                        sh 'chmod +x scripts/scan.sh'
                        sh './scripts/scan.sh'
                    }
                }
        stage('Publish Reports') {
            steps {
                archiveArtifacts artifacts: 'reports/*.json', fingerprint: true
            }
        }
    }
}
```

Note : Important - ensure that the scripts have executable privileges and that the reports directory exists in the jenkins repository on the server.

6.Configure Jenkins server with the necessary plugins -Java, Maven, Terraform, etc. Further configure a pipeline job to listen to changes on the "pentest" branch of the repository containing the source code - https://github.com/hanzel-sc/blog-backend



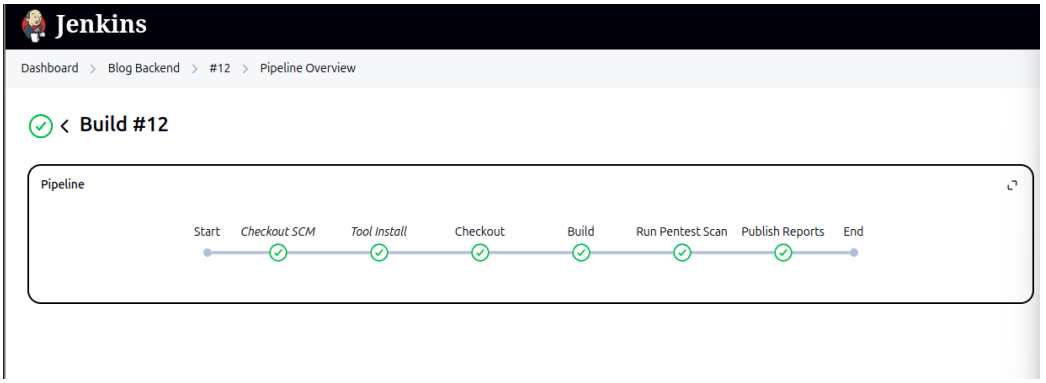
7.Pushes the code changes

```
To https://github.com/hanzel-sc/blog-backend.git
 * [new branch]      pentest -> pentest
PS C:\Users\chris\CS_Projects\Rantz\blog-backend> git add .
PS C:\Users\chris\CS_Projects\Rantz\blog-backend> git commit -m "Incorporated Infrastructure Pentest tools - Tfsec and Terrascan"
[pentest 4f125ec] Incorporated Infrastructure Pentest tools - Tfsec and Terrascan
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\chris\CS_Projects\Rantz\blog-backend> git push origin pentest
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 337 bytes | 337.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
```

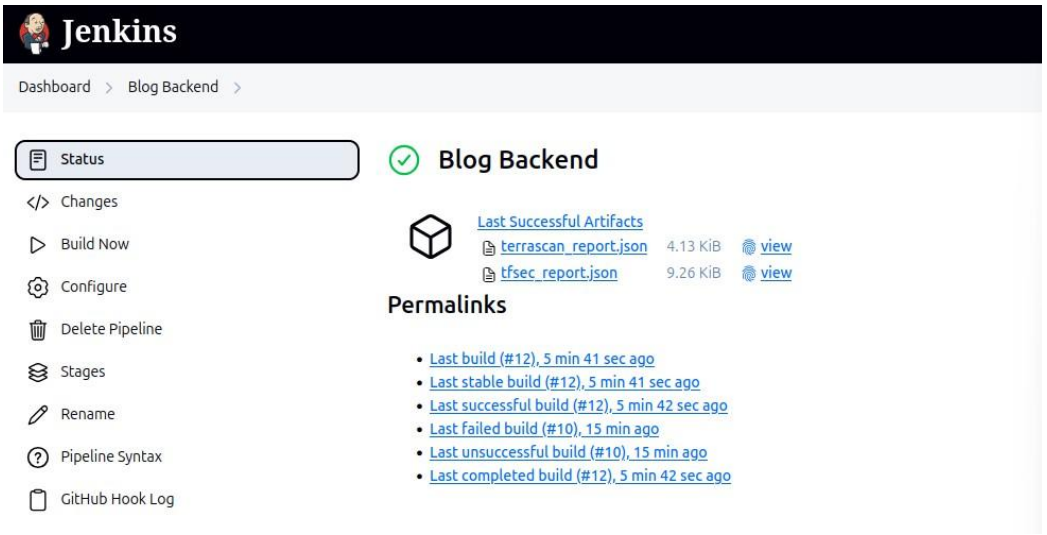
8.The webhook triggers the pipeline. Ensure that all stages are successful.

```
Terrascan and tfsec reports saved to /var/lib/jenkins/workspace/Blog Backend/reports/
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Publish Reports)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] archiveArtifacts
Archiving artifacts
Recording fingerprints
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

-Completed, successful execution of the pipeline

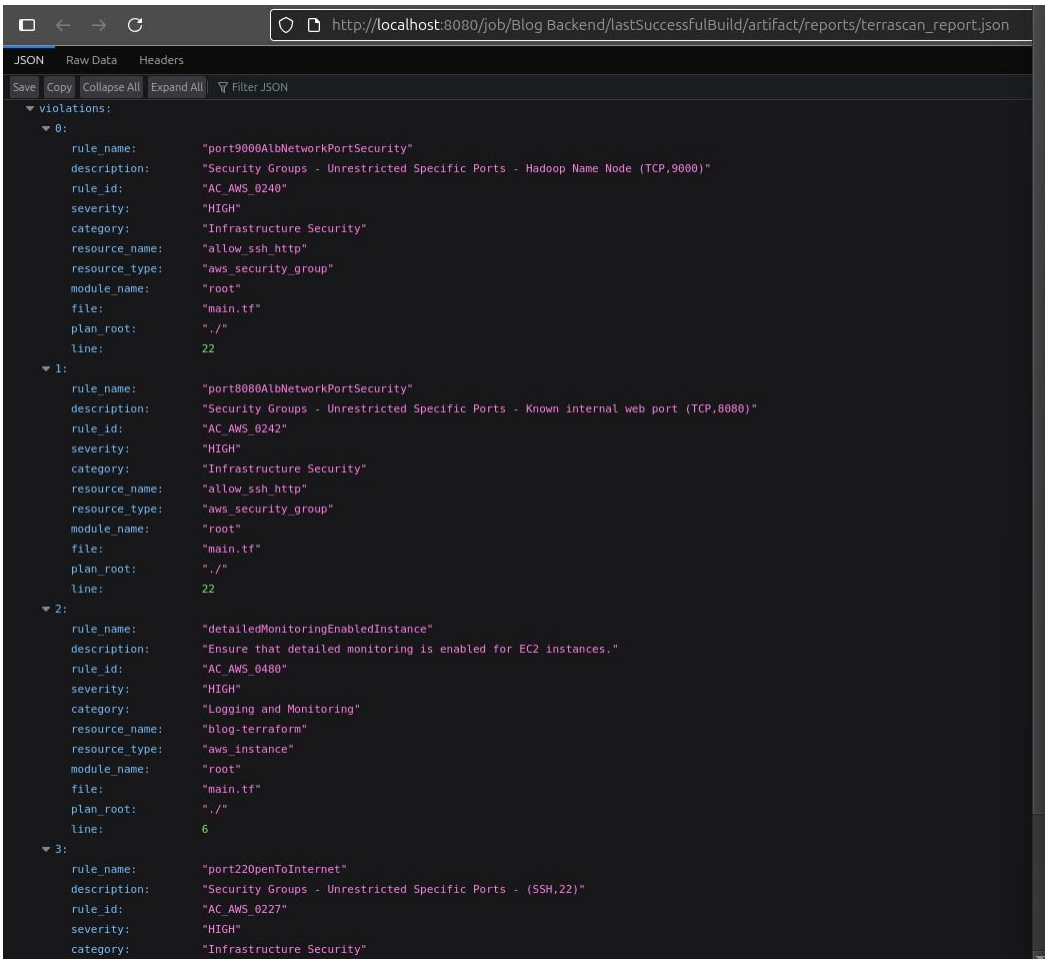


9. **The results of the two scans will be present as build artifacts on the Jenkins Server.**



10.Analyze the results. (The results are in JSON format)

-Terrascan scan results



-Tfsec scan results


```
5:
    rule_id:            "AVD-AWS-0028"
    long_id:            "aws-ec2-enforce-http-token-imds"
    rule_description:   "aws_instance should activate session tokens for Instance Metadata Service."
    rule_provider:      "aws"
    rule_service:       "ec2"
    impact:             "Instance metadata service can be interacted with freely"
    resolution:         "Enable HTTP token requirement for IMDS"
  links:
      0:                "https://aquasecurity.github.io/tfsec/v1.28.14/checks/aws/ec2/enforce-http-token-imds/"
      1:                "https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance#metadata-options"
    description:        "Instance does not require IMDS access to require a token"
    severity:           "HIGH"
    warning:            false
    status:             0
    resource:           "aws_instance.blog-terraform"
  location:
      filename:         "/var/lib/jenkins/workspace/Blog Backend/infrastructure/main.tf"
      start_line:       6
      end_line:         20
6:
    rule_id:            "AVD-AWS-0104"
    long_id:            "aws-ec2-no-public-egress-sgr"
    rule_description:   "An egress security group rule allows traffic to /0."
    rule_provider:      "aws"
    rule_service:       "ec2"
    impact:             "Your port is egressing data to the internet"
    resolution:         "Set a more restrictive cidr range"
  links:
      0:                "https://aquasecurity.github.io/tfsec/v1.28.14/checks/aws/ec2/no-public-egress-sgr/"
      1:                "https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security_group"
    description:        "Security group rule allows egress to multiple public internet addresses."
    severity:           "CRITICAL"
    warning:            false
    status:             0
    resource:           "aws_security_group.allow_ssh_http"
  location:
      filename:         "/var/lib/jenkins/workspace/Blog Backend/infrastructure/main.tf"
      start_line:       51
      end_line:         51
```