Lab Exercise 18- Scanning IaC Templates for Vulnerabilities

Name- Gourav das

SAP ID- 500122586

Batch- B2

Objective

- Learn how to scan Infrastructure as Code (IaC) templates for security vulnerabilities.

- Use open-source IaC security tools to detect misconfigurations.

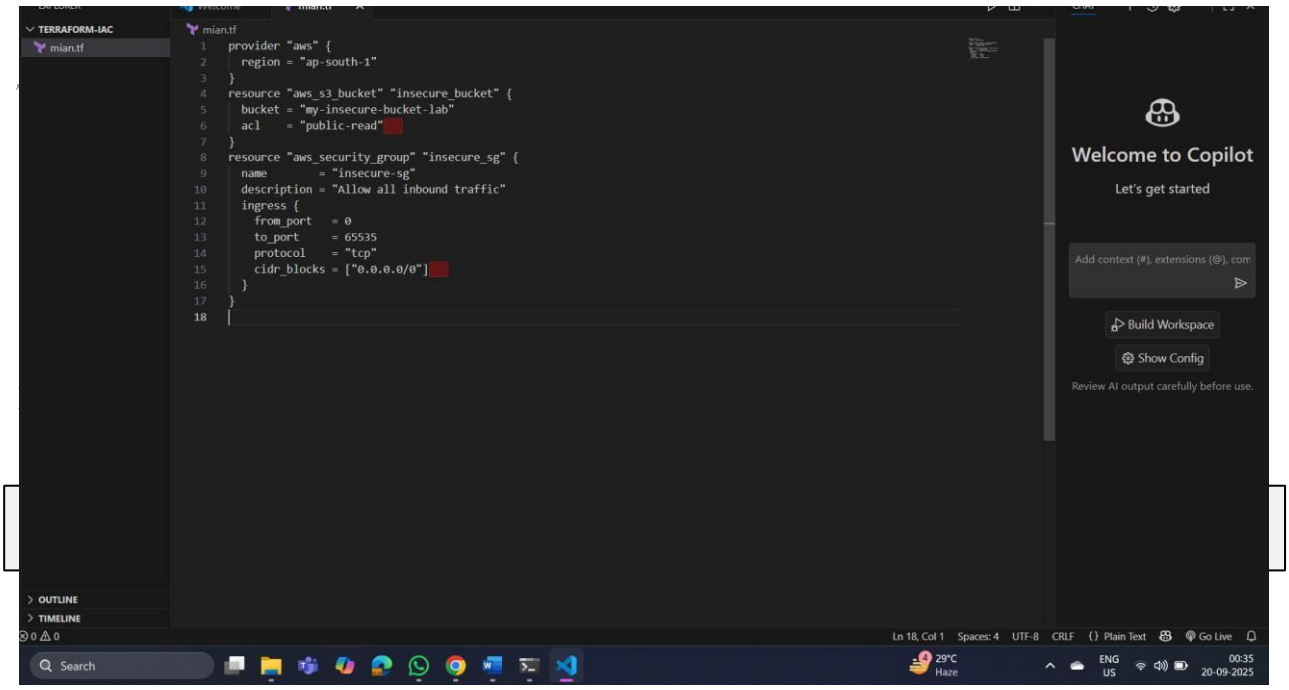
- Understand common risks such as public access, unencrypted resources, and insecure network rules.

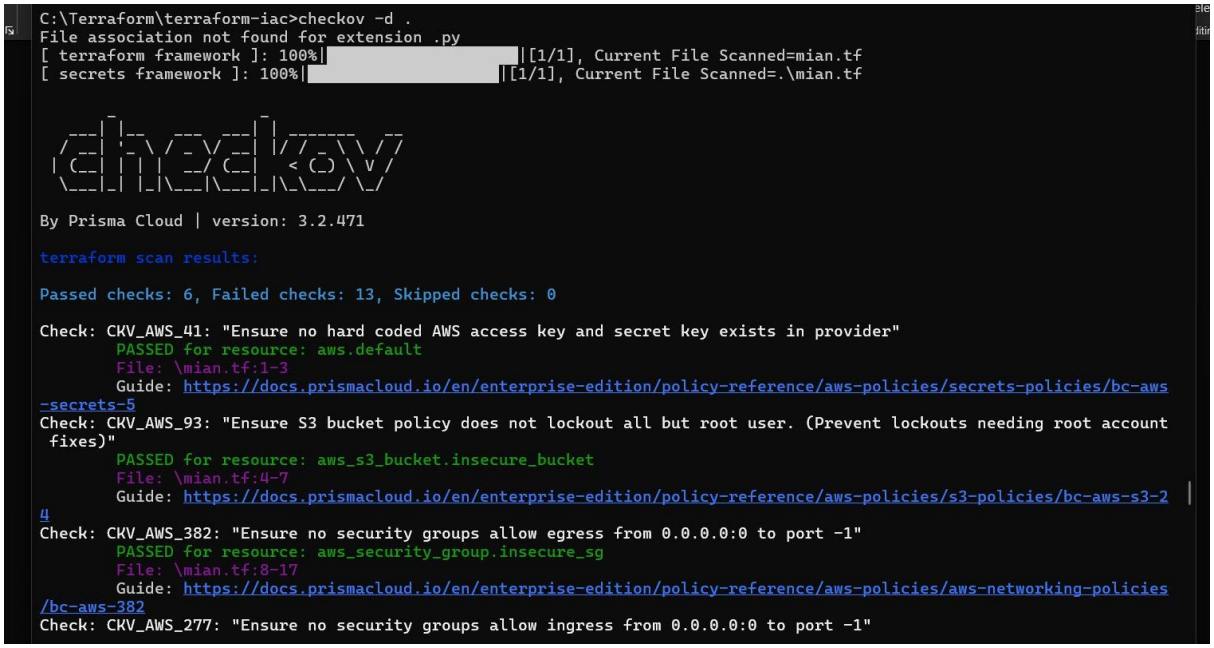
Step 1: Create an Insecure IaC Template

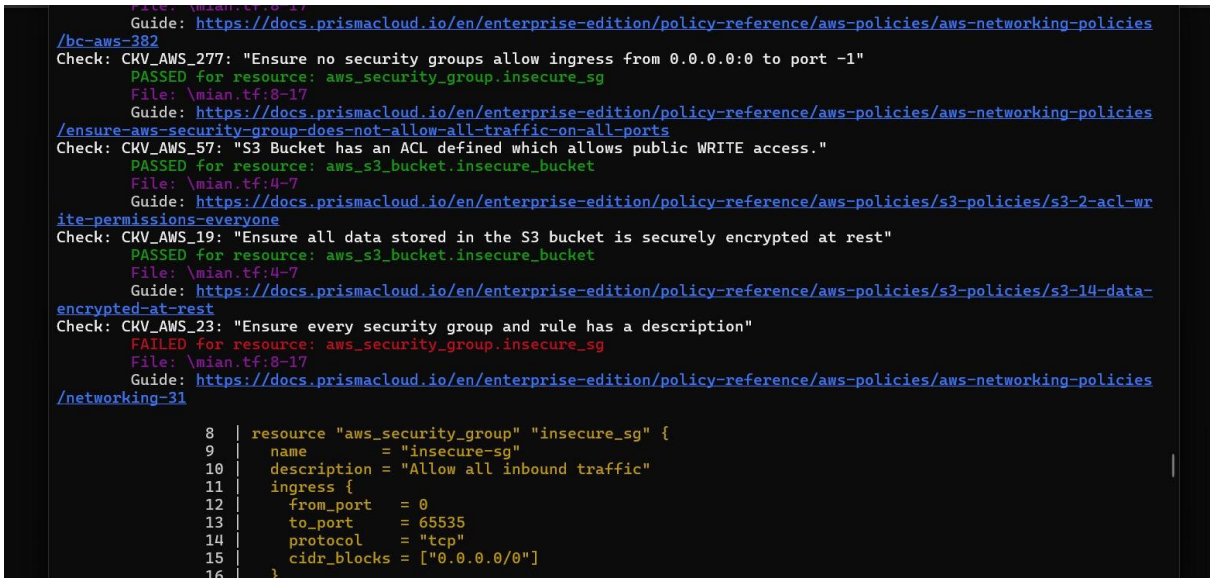
Create a file named main.tf with the following Terraform code:

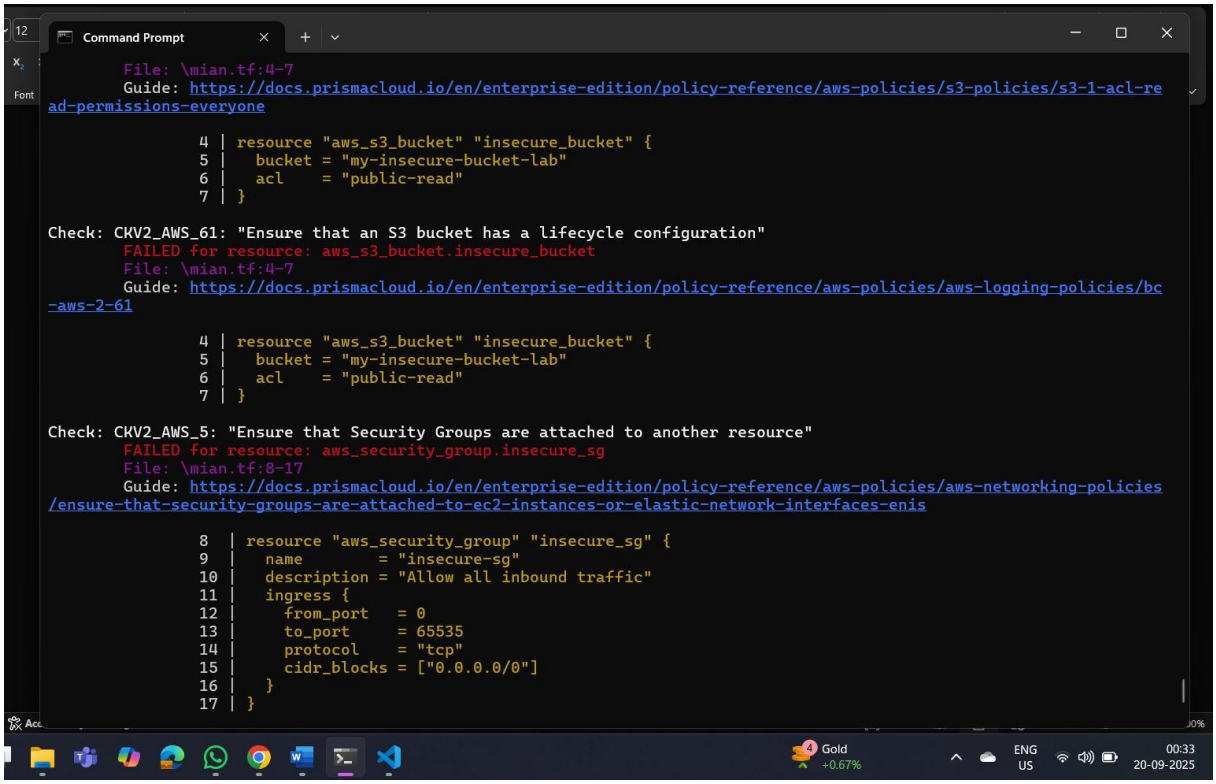
```
  provider

"aws" { region

= "us-east-1"
```

```
 resource "aws_s3_bucket"
"insecure_bucket" { bucket = "my-
insecure-bucket-lab"
  acl = "public-read"
}
 resource "aws_security_group"
"insecure_sg" { name       = "insecure-
sg"
  description = "Allow all
inbound traffic" ingress {
     from_port = 0
     to_port = 65535
 protocol = "tcp"
```


TERRAFORM-IAC
mian.tf
mian.tf
provider "aws" {
region = "ap-south-1"
}
resource "aws_s3_bucket" "insecure_bucket" {
bucket = "my-insecure-bucket-lab"
acl = "public-read"
}
resource "aws_security_group" "insecure_sg" {
name = "insecure-sg"
description = "Allow all inbound traffic"
ingress {
from_port = 0
to_port = 65535
protocol = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
}
Welcome to Copilot
Let's get started
Add context (#), extensions (@), com
Build Workspace
Show Config
Review AI output carefully before use.
OUTLINE
TIMELINE
Ln 18, Col 1
Spaces: 4
UTF-8
CRLF
Plain Text
Go Live
Search
29°C
Haze
ENG
US
00:35
20-09-2025


C:\Terraform\terraform-iac>checkov -d .
File association not found for extension .py
[terraform framework]: 100%|[1/1], Current File Scanned=mian.tf
[secrets framework]: 100%|[1/1], Current File Scanned=.\mian.tf
By Prisma Cloud | version: 3.2.471
terraform scan results:
Passed checks: 6, Failed checks: 13, Skipped checks: 0
Check: CKV_AWS_41: "Ensure no hard coded AWS access key and secret key exists in provider"
PASSED for resource: aws.default
File: \mian.tf:1-3
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/secrets-policies/bc-aws-secrets-5
Check: CKV_AWS_93: "Ensure S3 bucket policy does not lockout all but root user. (Prevent lockouts needing root account fixes)"
PASSED for resource: aws_s3_bucket.insecure_bucket
File: \mian.tf:4-7
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/s3-policies/bc-aws-s3-24
Check: CKV_AWS_382: "Ensure no security groups allow egress from 0.0.0.0:0 to port -1"
PASSED for resource: aws_security_group.insecure_sg
File: \mian.tf:8-17
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/aws-networking-policies/bc-aws-382
Check: CKV_AWS_277: "Ensure no security groups allow ingress from 0.0.0.0:0 to port -1"


Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/aws-networking-policies/bc-aws-382
Check: CKV_AWS_277: "Ensure no security groups allow ingress from 0.0.0.0:0 to port -1"
PASSED for resource: aws_security_group.insecure_sg
File: \mian.tf:8-17
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/aws-networking-policies/ensure-aws-security-group-does-not-allow-all-traffic-on-all-ports
Check: CKV_AWS_57: "S3 Bucket has an ACL defined which allows public WRITE access."
PASSED for resource: aws_s3_bucket.insecure_bucket
File: \mian.tf:4-7
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/s3-policies/s3-2-acl-write-permissions-everyone
Check: CKV_AWS_19: "Ensure all data stored in the S3 bucket is securely encrypted at rest"
PASSED for resource: aws_s3_bucket.insecure_bucket
File: \mian.tf:4-7
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/s3-policies/s3-14-data-encrypted-at-rest
Check: CKV_AWS_23: "Ensure every security group and rule has a description"
FAILED for resource: aws_security_group.insecure_sg
File: \mian.tf:8-17
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-policies/aws-networking-policies/networking-31
8 | resource "aws_security_group" "insecure_sg" {
9 | name = "insecure-sg"
10 | description = "Allow all inbound traffic"
11 | ingress {
12 | from_port = 0
13 | to_port = 65535
14 | protocol = "tcp"
15 | cidr_blocks = ["0.0.0.0/0"]



Expected Findings:

- Public S3 bucket access (public-read)
- Security group open to all inbound traffic

Expected Findings:

- Warns about S3 bucket without encryption
- Flags open Security Group rules

Step 4: Review the Report

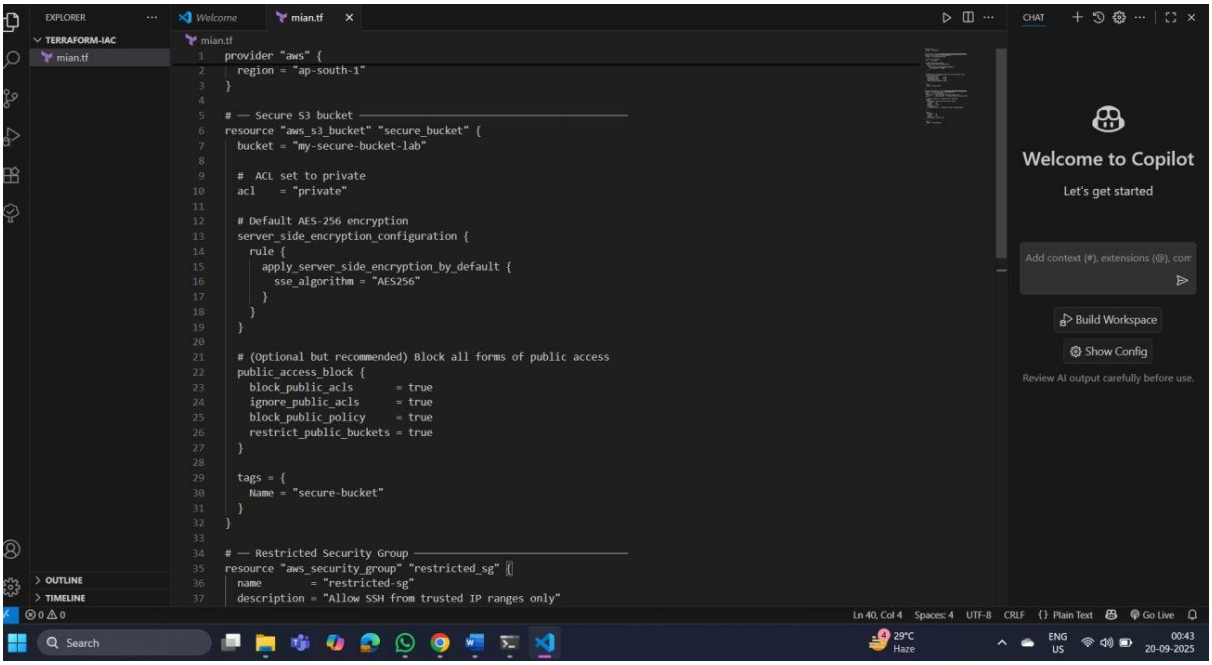
Example output (Checkov):

```
Check: CKV_AWS_20: "S3 Bucket allows public

read access" FAILED for resource:

aws_s3_bucket.insecure_bucket

Check: CKV_AWS_260: "Security group allows ingress from 0.0.0.0/0"
```

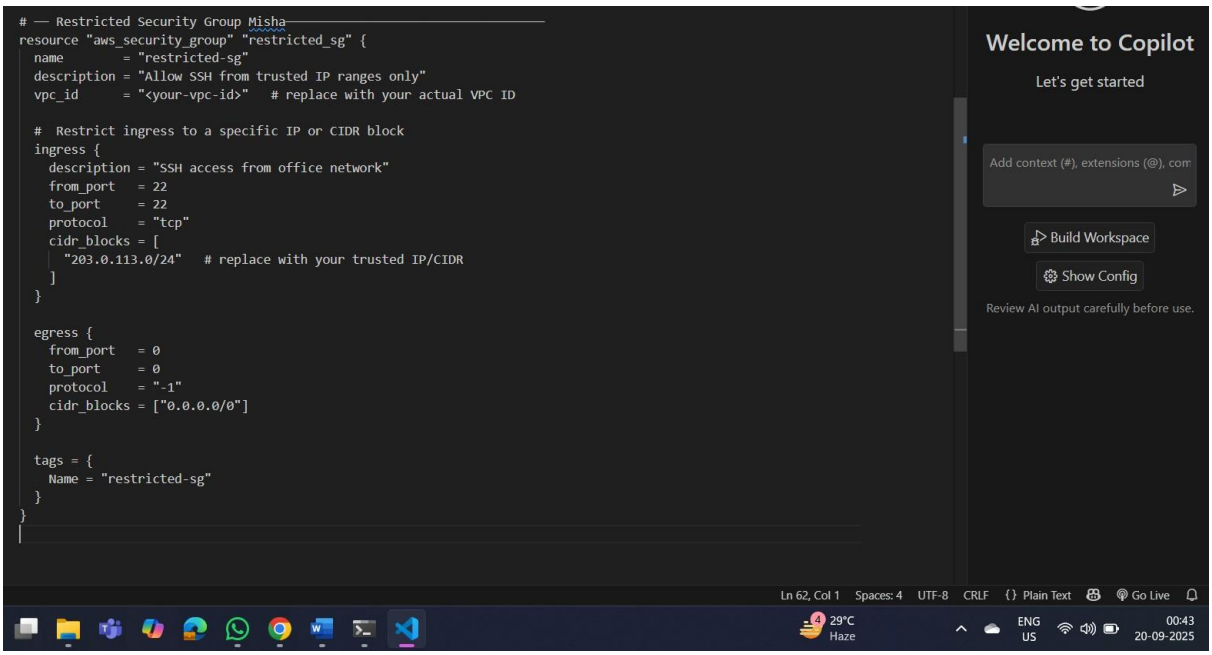
Step 5: Apply Fixes (Optional)

Modify the IaC template to:

- Set S3 bucket ACL to private
- Enable encryption (AES256)
- Restrict Security Group to specific IP ranges



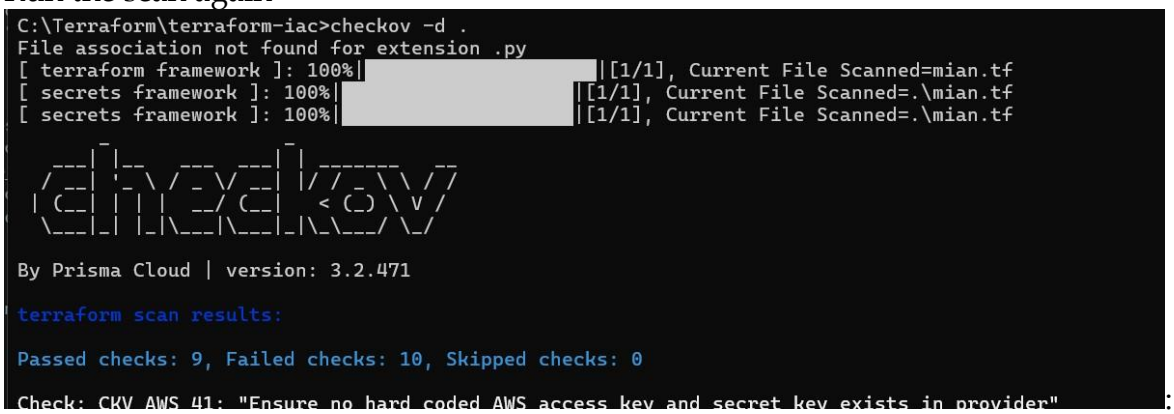

```hcl
provider "aws" {
  region = "ap-south-1"
}

# — Secure S3 bucket ————————————————————————
resource "aws_s3_bucket" "secure_bucket" {
  bucket = "my-secure-bucket-lab"

  #  ACL set to private
  acl    = "private"

  # Default AES-256 encryption
  server_side_encryption_configuration {
    rule {
      apply_server_side_encryption_by_default {
        sse_algorithm = "AES256"
      }
    }
  }

  # (Optional but recommended) Block all forms of public access
  public_access_block {
    block_public_acls       = true
    ignore_public_acls      = true
    block_public_policy     = true
    restrict_public_buckets = true
  }

  tags = {
    Name = "secure-bucket"
  }
}

# — Restricted Security Group ———————————————————
resource "aws_security_group" "restricted_sg" {
  name        = "restricted-sg"
  description = "Allow SSH from trusted IP ranges only"
```



```hcr
# — Restricted Security Group Misha————————————————————
resource "aws_security_group" "restricted_sg" {
  name        = "restricted-sg"
  description = "Allow SSH from trusted IP ranges only"
  vpc_id      = "<your-vpc-id>"   # replace with your actual VPC ID

  #  Restrict ingress to a specific IP or CIDR block
  ingress {
    description = "SSH access from office network"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [
      "203.0.113.0/24"   # replace with your trusted IP/CIDR
    ]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "restricted-sg"
  }
}
```

Step 6: Rescan the Template

Run the scan again



Now the findings should be **resolved or reduced**.

Step 7: Document Findingss

Create a simple findings log:

Before the securing, terraform scan results

Passed checks: **6**, **Failed** checks: **13**, Skipped checks: 0

After securing-

terraform scan

results:

Passed checks: **9**, Failed checks: **10**, Skipped

checks: The number of failed test checks reduced,.