sonar RULES

Products

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- **Terraform**
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML

Terraform static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your TERRAFORM code

All rules 50 | Vulnerability 5 | Security Hotspot 43 | Code Smell 2

Tags ⌄ | Search by name...

Excluding users or groups activities from audit logs is security-sensitive
Security Hotspot

Defining a short log retention duration is security-sensitive
Security Hotspot

Enabling Attribute-Based Access Control for Kubernetes is security-sensitive
Security Hotspot

Creating custom roles allowing privilege escalation is security-sensitive
Security Hotspot

Creating App Engine handlers without requiring TLS is security-sensitive
Security Hotspot

Excessive granting of GCP IAM permissions is security-sensitive
Security Hotspot

Enabling project-wide SSH keys to access VM instances is security-sensitive
Security Hotspot

Granting public access to GCP resources is security-sensitive
Security Hotspot

Creating GCP SQL instances without requiring TLS is security-sensitive
Security Hotspot

Creating DNS zones without DNSSEC enabled is security-sensitive
Security Hotspot

Creating keys without a rotation period is security-sensitive
Security Hotspot

Weak SSL/TLS protocols should not be used

Analyze your code

Vulnerability Critical

aws azure cwe gcp privacy owasp sans-top25

This rule raises an issue when an insecure TLS protocol version (i.e. a protocol different from "TLSv1.2", "TLSv1.3", "DTLSv1.2", or "DTLSv1.3") is used or allowed.

It is recommended to enforce TLS 1.2 as the minimum protocol version and to disallow older versions like TLS 1.0. Failure to do so could open the door to downgrade attacks: a malicious actor who is able to intercept the connection could modify the requested protocol version and downgrade it to a less secure version.

See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2021 Category A7 - Identification and Authentication Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- Mobile AppSec Verification Standard - Network Communication Requirements
- OWASP Mobile Top 10 2016 Category M3 - Insecure Communication
- MITRE, CWE-327 - Inadequate Encryption Strength
- MITRE, CWE-326 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses
- SSL and TLS Deployment Best Practices - Use secure protocols

Noncompliant Code Example

For Amazon OpenSearch domains:

```
resource "aws_elasticsearch_domain" "example" {
  domain_name = "example"
  domain_endpoint_options {
    enforce_https = true
    tls_security_policy = "Policy-Min-TLS-1-0-2019-07"
  }
```

For Amazon API Gateway:

```
resource "aws_api_gateway_domain_name" "example" {
  domain_name = "api.example.com"
  security_policy = "TLS_1_0" # Noncompliant
}
```

```
resource "aws_apigatewayv2_domain_name" "example" {
  domain_name = "api.example.com"
```

Security Hotspot

Granting highly privileged GCP resource rights is security-sensitive

Security Hotspot

Using unencrypted cloud storages is security-sensitive

Security Hotspot

Azure role assignments that grant access to all resources of a subscription are security-sensitive

Security Hotspot

Disabling Role-Based Access Control on Azure resources is security-sensitive

Security Hotspot

```
  domain_name_configuration {} # Noncompliant
}
```

For Google Cloud load balancers

```
resource "google_compute_ssl_policy" "example" {
  name            = "example"
  min_tls_version = "TLS_1_0" # Noncompliant
  # ...
}
```

Compliant Solution

For Amazon OpenSearch domains:

```
resource "aws_elasticsearch_domain" "example" {
  domain_name = "example"
  domain_endpoint_options {
    enforce_https = true
    tls_security_policy = "Policy-Min-TLS-1-2-2019-07"
```

For Amazon API Gateway:

```
resource "aws_api_gateway_domain_name" "example" {
  domain_name = "api.example.com"
  security_policy = "TLS_1_2"
}
```

```
resource "aws_apigatewayv2_domain_name" "example" {
  domain_name = "api.example.com"
  domain_name_configuration {
    security_policy = "TLS_1_2"
  }
}
```

For Google Cloud load balancers

```
resource "google_compute_ssl_policy" "example" {
  name            = "example"
  min_tls_version = "TLS_1_2"
  # ...
}
```

See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2021 Category A7 - Identification and Authentication Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- Mobile AppSec Verification Standard - Network Communication Requirements
- OWASP Mobile Top 10 2016 Category M3 - Insecure Communication
- MITRE, CWE-327 - Inadequate Encryption Strength
- MITRE, CWE-326 - Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 - Porous Defenses
- SSL and TLS Deployment Best Practices - Use secure protocols
- Amazon API Gateway - Choosing a minimum TLS version
- Google Cloud Load Balancing - Defining an SSL policy

Available In:

sonarcloud sonarqube