




 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... 


 Security Hotspot


Using unencrypted EFS file systems is security-sensitive


 Security Hotspot


 Using unencrypted SQS queues is security-sensitive


 Using unencrypted SNS topics is security-sensitive


 Using unencrypted SageMaker notebook instances is security-sensitive


 Using unencrypted Elasticsearch domains is security-sensitive


 Using unencrypted RDS databases is security-sensitive

 Using unencrypted EBS volumes is security-sensitive

 Disabling logging is security-sensitive


 Administration services access should be restricted to specific IP addresses



 Unversioned Google Cloud Storage buckets are security-sensitive


 Disabling S3 bucket MFA delete is security-sensitive

Authorizing anonymous access to Azure resources is security-sensitive

Analyze your code

 Security Hotspot

 Major 

 cwe owasp azure

Authorizing anonymous access can reduce an organization's ability to protect itself against attacks on its Azure resources.

Security incidents may include disrupting critical functions, data theft, and additional Azure subscription costs due to resource overload.

Using authentication coupled with fine-grained authorizations helps bring Defense-In-Depth and bring traceability to investigators of security incidents.

Depending on the affected Azure resource, multiple authentication choices are possible: Active Directory Authentication, OpenID implementations (Google, Microsoft, etc.) or native Azure mechanisms.

Ask Yourself Whether

- This Azure resource is essential for the information system infrastructure.
- This Azure resource is essential for mission-critical functions.
- This Azure resource stores or processes sensitive data.
- Compliance policies require access to this resource to be authenticated.

There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

Enable authentication in this Azure resource, and disable anonymous access.

If only Basic Authentication is available, enable it.

Sensitive Code Example

For [App Services and equivalent](#):

```
resource "azurerm_function_app" "example" {
  name = "example"

  auth_settings {
    enabled = false # Sensitive
  }

  auth_settings {
    enabled = true
    unauthenticated_client_action = "AllowAnonymous" # Sensitive
  }
}
```





For [API Management](#):

```
resource "azurerm_api_management_api" "example" { # Sensitive
  name = "example-api"
}

resource "azurerm_api_management" "example" {
```

https://rules.sonarsource.com/terraform/RSPEC-6380

1/3

<div>Disabling versioning of S3 buckets is security-sensitive</div> <div> Security Hotspot</div>
<div>Disabling server-side encryption of S3 buckets is security-sensitive</div> <div> Security Hotspot</div>
<div>AWS tag keys should comply with a naming convention</div> <div> Code Smell</div>
<div>Terraform parsing failure</div> <div> Code Smell</div>

```
sign_in {
  enabled = false # Sensitive
}
```

For [Data Factory](#) Linked Services:

```
resource "azurerm_data_factory_linked_service_sftp" "example" {
  authentication_type = "Anonymous"
}
```

For [Storage Accounts](#):

```
resource "azurerm_storage_account" "example" {
  allow_blob_public_access = true # Sensitive
}

resource "azurerm_storage_container" "example" {
  container_access_type = "blob" # Sensitive
}
```

For [Redis Caches](#):

```
resource "azurerm_redis_cache" "example" {
  name = "example-cache"

  redis_configuration {
    enable_authentication = false # Sensitive
  }
}
```

Compliant Solution

For [App Services and equivalent](#):

```
resource "azurerm_function_app" "example" {
  name = "example"

  auth_settings {
    enabled = true
    unauthenticated_client_action = "RedirectToLoginPage"
  }
}
```

For [API Management](#):

```
resource "azurerm_api_management_api" "example" {
  name = "example-api"

  openid_authentication {
    openid_provider_name = azurerm_api_management_openid_con
  }
}

resource "azurerm_api_management" "example" {
  sign_in {
    enabled = true
  }
}
```

For [Data Factory](#) Linked Services:

```
resource "azurerm_data_factory_linked_service_sftp" "example" {
  authentication_type = "Basic"
  username            = local.creds.username
  password            = local.creds.password
}

resource "azurerm_data_factory_linked_service_odata" "example" {
  basic_authentication {
    username = local.creds.username
    password = local.creds.password
  }
}
```

For [Storage Accounts](#):

```
resource "azurerm_storage_account" "example" {  
  allow_blob_public_access = true  
}  
  
resource "azurerm_storage_container" "example" {  
  container_access_type = "private"  
}
```

For [Redis Caches](#):

```
resource "azurerm_redis_cache" "example" {  
  name = "example-cache"  
  
  redis_configuration {  
    enable_authentication = true  
  }  
}
```

See

- [OWASP Top 10 2021 Category A1](#) - Broken Access Control
- [OWASP Top 10 2017 Category A5](#) - Broken Access Control
- [MITRE, CWE-668](#) - Exposure of Resource to Wrong Sphere

Available In:

sonarcloud  | **sonarqube** 

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.
[Privacy Policy](#)