



ASPM User Playbook



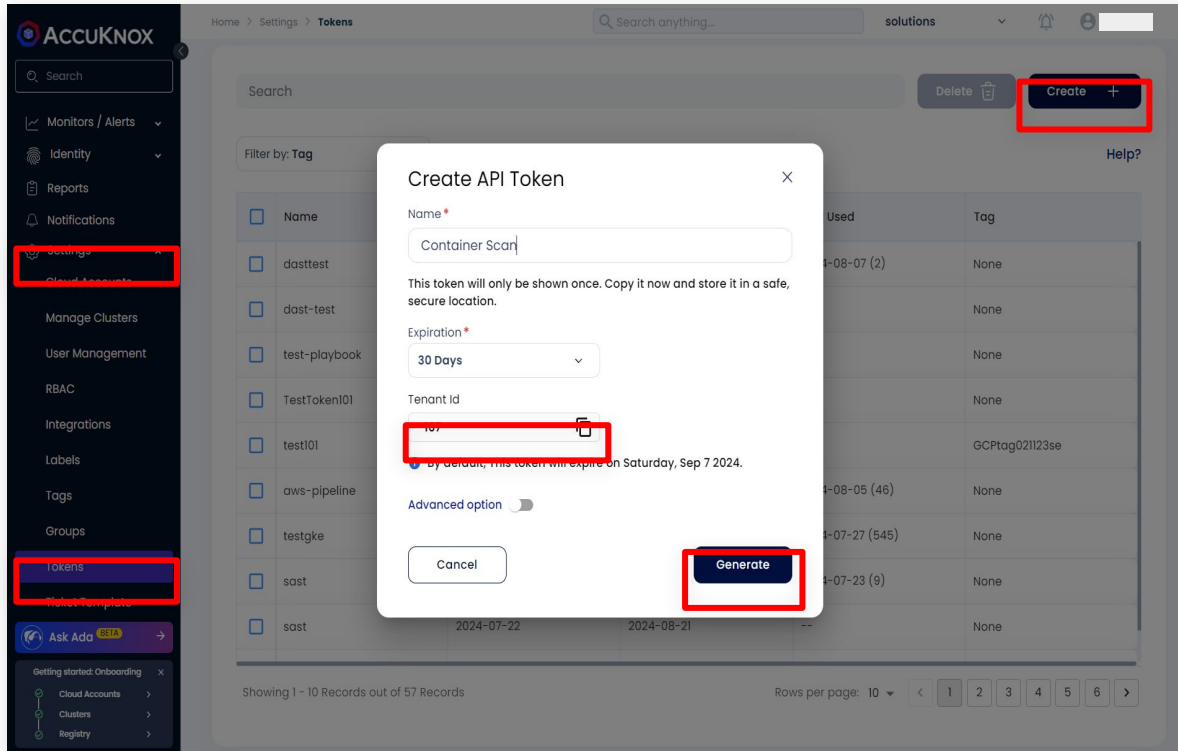


Container Scan

ASPM Integration

Generate AccuKnox API token for CI/CD pipeline

- To generate a token, open AccuKnox and navigate to Settings > Tokens > Create.
- Copy the token and tenant ID, then configure them as secrets in your CI/CD pipeline.



The screenshot shows the AccuKnox web interface. On the left sidebar, the 'Settings' menu item is highlighted with a red box. In the main content area, the 'Tokens' page is displayed, and the 'Create' button in the top right corner is highlighted with a red box. A 'Create API Token' dialog box is open in the center, with the following details:

- Name:** Container Scan
- Expiration:** 30 Days
- Tenant Id:** [Redacted]

The dialog box also includes a warning: "This token will only be shown once. Copy it now and store it in a safe, secure location." and a note: "By default, this token will expire on Saturday, Sep 7 2024." The 'Generate' button at the bottom right of the dialog is highlighted with a red box.

Step 1: Add [AccuKnox Container Scan](#) to Your GitHub Workflow

- Open your GitHub repository and navigate to your workflow file (typically `.github/workflows/your-workflow.yml`).
- After the build step, add the AccuKnox container scan GitHub Action.

Step 2: Run the Workflow

- Push your changes to trigger the workflow, or manually run it from the "Actions" tab in your repository.

Step 3: Review Findings in AccuKnox

- Log in to your AccuKnox dashboard and navigate to the Issues section.
- Go to the "Findings" tab and select Container Image Findings.
- Click on any finding that interests you to view detailed information and recommendations.

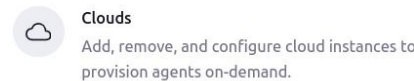
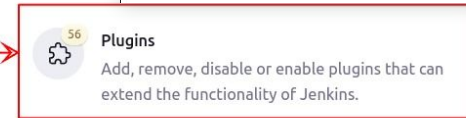
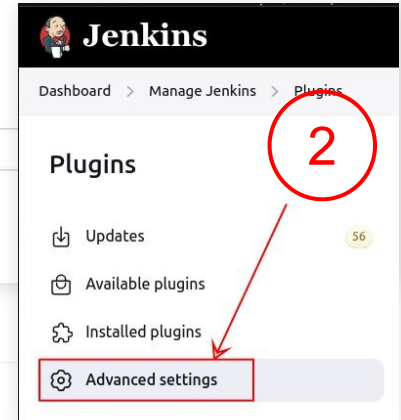
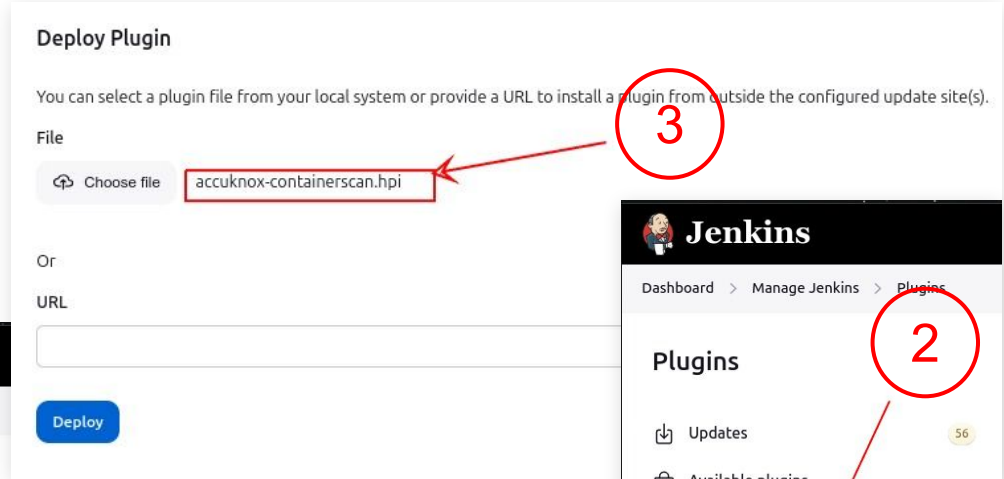
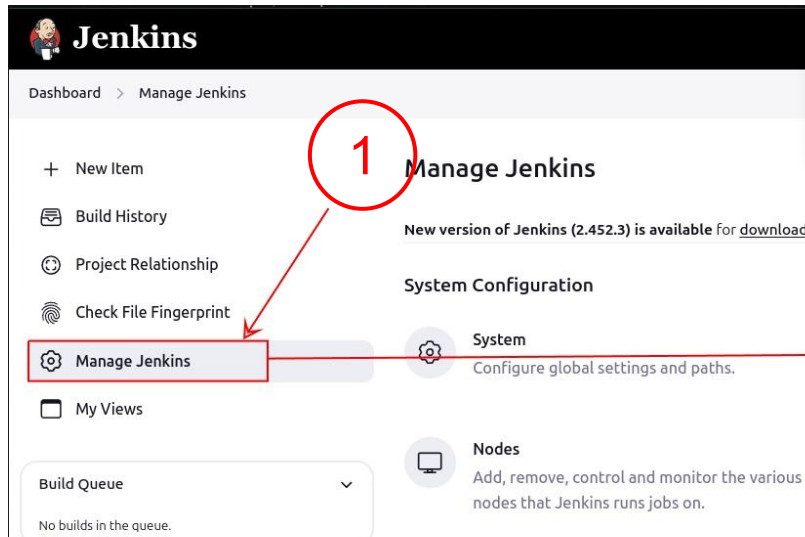
```
jobs:
  accuknox-cicd:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@main

      - name: Build Docker image
        run: |
          docker buildx build .

      - name: AccuKnox Container Scan
        uses: accuknox/container-scan-action@v0.0.1
        with:
          token: ${ secrets.TOKEN }
          tenant_id: ${ secrets.TENANT_ID }
```

Configuring the container scan in Jenkins

- Download the plugin in .hpi format from [here](#).
- Navigate to the Jenkins dashboard.
- Go to Manage Jenkins > Plugins > Advance settings.
- Deploy the plugin.



- Open the configuration page of your Jenkins job.
- Under the Build section, click on Add build step and select scan image with AccuKnox.
- Fill all the required parameters and trigger the pipeline.
- Review the findings in AccuKnox > Issues > Findings > Container scan.

Dashboard > test > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Steps**
- Post-build Actions

Scan Image with AccuKnox

Image Name

Image Tag

Exit Code

Severity

AccuKnox Token

Tenant ID

AccuKnox Label

Save Apply

5

Dashboard > test > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Steps**
- Post-build Actions

Source Code Management

None

Build Triggers

Build after other projects are built ?

Build periodically ?

Poll SCM ?

Build Steps

Add build step -

- Execute Windows batch command
- Execute shell
- Invoke top-level Maven targets
- Scan Image with AccuKnox**
- Add post-build action -

Save Apply

6

Dashboard > test >

- Status
- Changes
- Workspace
- Build Now
- Configure**
- Delete Project
- Rename

4

6

View Findings in registry scan page

- You can see the findings on the registry scan page.
- Click any of the findings to get more details.

The screenshot displays the AccuKnox dashboard with the 'Registry Scan' menu item highlighted in the left sidebar. The main content area shows a list of repositories with a table of scan results. A red arrow points from a finding in the table to a detailed view window.

Registry Scan Page:

- Navigation: Home > Issues > Registry Scan > Image Details
- Repository: `accuknox-container-scan-example:8185438405`
- Findings: 0 Critical, 36 High, 0 Medium, 0 Low, 0 Info

Image Details View:

- Architecture: amd64
- Content Digest: sha256:2f42c9c325700516bd7f217f7b56c63499a7696e33bac7d9848040b625b740f2
- Created: 03/07/2024 02:28 PM
- Docker Digest: sha256:2f42c9c325700516bd7f217f7b56c63499a7696e33bac7d9848040b625b740f2
- Docker ID: sha256:2f42c9c325700516bd7f217f7b56c63499a7696e33bac7d9848040b625b740f2
- Environment:

```
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
LANG=C.UTF-8
GPG_KEY=0D96DF4D4110E5C43FBFB17F2D347EA6AA6542ID
PYTHON_VERSION=3.6.15
PYTHON_PIP_VERSION=21.2.4
PYTHON_SETUPTOOLS_VERSION=57.5.0
PYTHON_GET_PIP_URL=https://github.com/pypa/get-pip/raw/3cb888cc2869620f57d5d2da64da38f516078c7/public/get-pip.py
PYTHON_GET_PIP_SHA256=c518250e91a70d7b20cceb15272209a4ded2a0c2630e5776f129e0d9b5674309
```
- Operating System: linux (alpine)

Vulnerability Scan Details:

- Image: `accuknox-container-scan-example:8185438405`
- Created: 7 day(s) ago
- Total: 36
- Summary: 0 Critical, 36 High, 0 Medium, 0 Low, 0 Info

View Findings in Findings page

- Alternatively, you can view the findings on the **Findings** page. Select the **Container Image Findings** to access the relevant details.

The screenshot shows the ACCUKNOX interface. On the left is a dark sidebar with navigation items: Dashboard, Inventory, Issues, Findings (highlighted in red), Registry Scan, Compliance, Runtime Protection, Remediation, Monitors / Alerts, Identity, Reports, Notifications, and Settings. Below the sidebar is a 'Ask Ada BETA' button and an 'Onboarding' section with links for Cloud Accounts, Clusters, and Registry.

The main content area is titled 'Home > Issues > Findings'. It features a search bar, a dropdown menu for 'Container Image Findings' (highlighted in red), and filters for 'Asset' and 'Group by'. Below these are icons for edit, view, print, download, and refresh.

	Identification numbers	Name	Assetname	Risk factor	Pkg name
<input type="checkbox"/>	CVE-2017-7475, CWE-476	cairo: NULL pointer dere...	sujoy13/php:latest	Low	libcairo2
<input type="checkbox"/>	CVE-2024-26792, No CWE	kernel: btrfs: fix double f...	sujoy13/php:latest	Medium	linux-libc-dev
<input type="checkbox"/>	CVE-2024-40941, No CWE	kernel: wifi: iwlfwif: mvm:...	sujoy13/php:latest	Medium	linux-libc-dev
<input type="checkbox"/>	CVE-2022-48862, No CWE	kernel: vhost: fix hung th...	sujoy13/php:latest	Medium	linux-libc-dev
<input type="checkbox"/>	CVE-2020-35501, CWE-863	kernel: audit not loggin...	sujoy13/php:latest	Low	linux-libc-dev
<input type="checkbox"/>	CVE-2024-26700, No CWE	kernel: drm/amd/displa...	sujoy13/php:latest	Medium	linux-libc-dev
<input type="checkbox"/>	CVE-2023-39615, CWE-119	libxml2: crafted xml can...	sujoy13/php:latest	Medium	libxml2
<input type="checkbox"/>	CVE-2021-3864, CWE-284	kernel: descendant's du...	sujoy13/php:latest	High	linux-libc-dev
<input type="checkbox"/>	CVE-2021-47448, No CWE	kernel: mptcp: fix possib...	sujoy13/php:latest	Medium	linux-libc-dev

Total Records: 50373

- Use case 1: Dependency analysis - scanning for supply chain vulnerabilities
- Use case 2: Scan for sensitive data exposure
- Use case 3: Authentication Vulnerabilities
- Use case 4: Remote Code Execution (RCE) vulnerabilities
- Use case 5: Denial of Service (DoS) vulnerabilities

Use case 1: Dependency analysis - scanning for supply chain vulnerabilities

- This jetty server 9.2.26 is vulnerable to XSS
- AccuKnox container scan identifies this vulnerability
- Provides you the solution



jetty: using specially formatted URL against DefaultServlet or ResourceHandler leads to XSS conditions: (org.eclipse.jetty:jetty-server@7.6.0.v20120127) Medium

Description	Result	Solution	References	Source Code
In Eclipse Jetty version 9.2.26 and older, 9.3.25 and older, and 9.4.15 and older, the server is vulnerable to XSS conditions if a remote client USES a specially formatted URL against the DefaultServlet or ResourceHandler that is configured Show More...				
Finding for in resource <code>Container rajvanshi/storm:latest</code>				
Failing since about 1 month ago, on 23/07/2024				

Details [+ Create Ticket](#)

Asset
rajvanshi/storm:latest

Asset Type
Container

Status 
Active

Ignored
 No

Severity 
Medium

Tickets
0

Notes 
Add Comments and Press Ctrl + Enter to Submit



jetty: using specially formatted URL against DefaultServlet or ResourceHandler leads to XSS conditions: (org.eclipse.jetty:jetty-server@7.6.0.v20120127) Medium

Description	Result	Solution	References	Source Code
Upgrade to version 9.2.27.v20190403, 9.3.26.v20190403, 9.4.16.v20190411 of the package org.eclipse.jetty:jetty-server				

Details [+ Create Ticket](#)

Asset
rajvanshi/storm:latest

Asset Type

Use case 2: Scan for sensitive data exposure

- This container image have multiple RSA private keys
- AccuKnox container scans for the sensitive data exposure and reports it.

rajvanshi/juice-shop:latest

Overview Vulnerabilities Resources **Sensitive Data** Scan History Layers

RSA private Key

File Name	Full Path
last-login-ip.component.spec.ts	/juice-shop/frontend/src/app/last-login-ip/last-login-ip.component.spec.ts
app.guard.spec.ts	/juice-shop/frontend/src/app/app.guard.spec.ts
insecurity.js	/juice-shop/build/lib/insecurity.js
insecurity.ts	/juice-shop/lib/insecurity.ts

- The apache derby is a JDBC driver
- In this case it's vulnerable to a broken authentication
- AccuKnox proposes a solution to upgrading it to version 10.14.3

Finding ×

A cleverly devised username might bypass LDAP authentication checks. I ...: (org.apache.derby:derby@10.10.2.0) critical

Description

A cleverly devised username might bypass LDAP authentication checks. In LDAP-authenticated Derby installations, this could let an attacker fill up the disk by creating junk Derby databases. In LDAP-authenticated Derby installations, this [Show More...](#)

Compliance Frameworks

No compliance found

Solution

Upgrade to version 10.14.3, 10.15.2.1, 10.16.1.2, 10.17.1.0 of the package org.apache.derby:derby

- Jackson is a popular Java library used for processing JSON, here it's vulnerable to code execution.
- AccuKnox identifies the vulnerability and suggests to updating Jackson to version 2.9.10.4

Finding



A deserialization flaw was discovered in jackson-databind through 2.9. ...: (com.fasterxml.jackson.core:jackson-databind@2.6.3)

High

Description

A deserialization flaw was discovered in jackson-databind through 2.9.10.4. It could allow an unauthenticated user to perform code execution via ignite-jta or quartz-core:
org.apache.ignite.cache.jta.jndi.CacheJndiTxLookup,
org.apache.ignite.cache.jta.jndi.CacheJndiTxFactory, and
org.quartz.utils.JNDIConnectionProvider.

[Show Less...](#)

Solution

Upgrade to version 2.9.10.4 of the package
com.fasterxml.jackson.core:jackson-databind

- Netty codec is a java is a java library, here it's vulnerable to a Denial of Service attack via a memory leakage.
- AccuKnox identifies the issue and reports it.

netty-codec: Bzip2Decoder doesn't allow setting size restrictions for decompressed data: (io.netty:netty-codec@4.1.30.Final)

High



Description

Result

Solution

References

Source Code

The Bzip2 decompression decoder function doesn't allow setting size restrictions on the decompressed output data (which affects the allocation size used during decompression). All users of Bzip2Decoder are affected. The malicious input can trigger an OOME and so a DoS attack

[Show Less...](#)

Details

[+ Create Ticket](#)

Asset

rajvanshi/storm:latest

Asset Type

Container

Status 

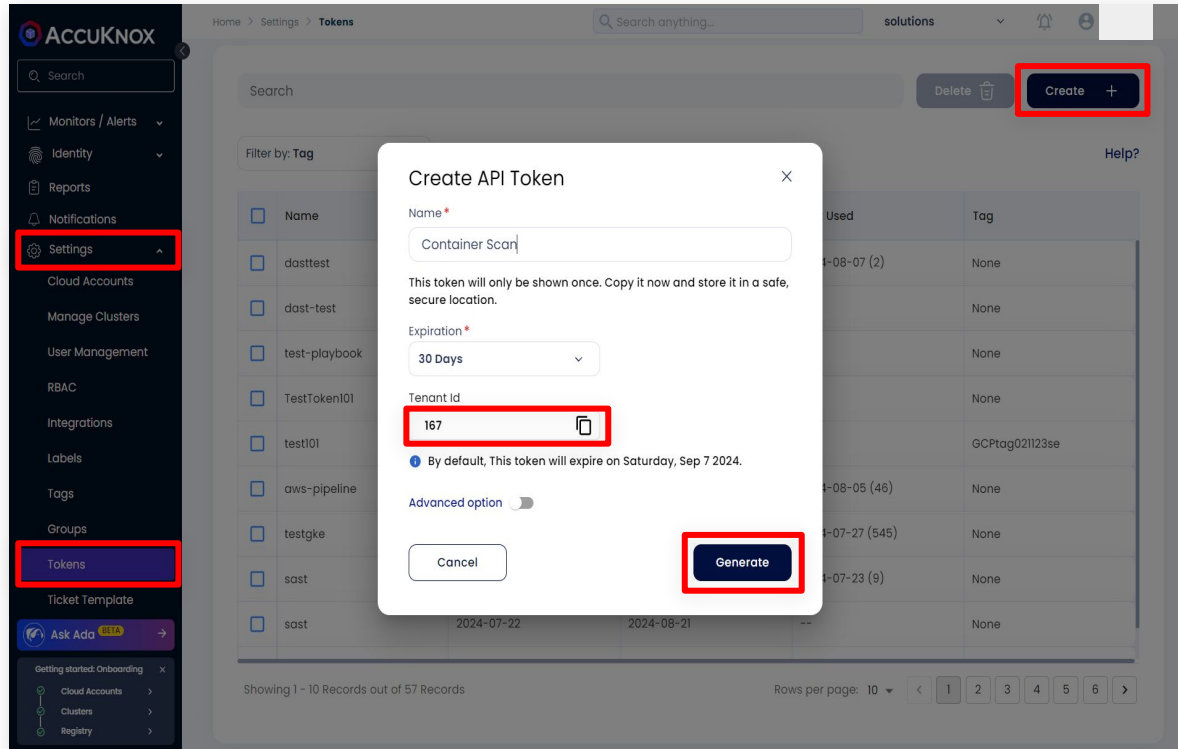


IaC Scan

ASPM Integration

Generate AccuKnox API token for CI/CD pipeline

- To generate a token, open AccuKnox and navigate to **Settings > Tokens > Create**.
- Copy the token and tenant ID, then configure them as secrets in your CI/CD pipeline.



Step 1: Add [AccuKnox IAC scan GitHub action](#) to your workflow like this image.

Step 2: Run the Workflow

- Push your changes to trigger the workflow, or manually run it from the "Actions" tab in your repository.

Step 3: Review Findings in AccuKnox

- Log in to your AccuKnox dashboard and navigate to the Issues section.
- Go to the "Findings" tab and select IaC Findings.
- Click on any finding that interests you to view detailed information and recommendations.

```
jobs:
  tests:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@main

      - name: Run IaC scan
        uses: accuknox/iac-scan-action@v0.0.1
        with:
          directory: ./
          output_file_path: ./results
          token: ${{ secrets.TOKEN }}
          endpoint: ${{ secrets.ENDPOINT }}
          tenant_id: ${{ secrets.TENANT_ID }}
          quiet: "true"
          soft_fail: "true"
```

Configuring the IaC scan in Jenkins

- Download the plugin in .hpi format from [here](#).
- Navigate to the Jenkins dashboard.
- Go to Manage Jenkins > Plugins > Advance settings.
- Deploy the plugin.

The image shows two screenshots from the Jenkins web interface. The left screenshot is the 'Manage Jenkins' page, with a red circle '1' around the 'Manage Jenkins' link in the left sidebar. A red arrow points from this link to the 'Plugins' section in the main content area, which is also circled with a red circle '2'. Within the 'Plugins' section, the 'Advanced settings' link is highlighted with a red box. The right screenshot is the 'Deploy Plugin' page, with a red circle '3' around the 'File' input field. A red arrow points from this field to the text 'accuknox-containerscan.hpi' which is also highlighted with a red box. A blue 'Deploy' button is visible at the bottom of the 'Deploy Plugin' page.

- Open the configuration page of your Jenkins job.
- Under the Build section, click on Add build step and select AccuKnox IaC scan.
- Fill all the required parameters.
- Review the findings in AccuKnox > Issues > Findings > Container scan.

Dashboard > accuknox-iac-scan > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Build Steps

AccuKnox IaC Scan

Directory

File

Soft Fail
true

Framework
all

Repository

Branch

AccuKnox Token

Tenant ID

Add build step ^

Save Apply

Dashboard > accuknox-iac-scan > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Prepare SonarQube Scanner environment ?
- Set GitHub commit status with custom context an
- Terminate a build if it's stuck
- With Ant ?

Build Steps

Add build step ^

Filter

AccuKnox IaC Scan

Dashboard > test >

Jenkins

- Status
- Changes
- Workspace
- Build Now
- Configure**
- Delete Project
- Rename

- Go to the AccuKnox > Findings and select the IAC Scan from the drop down menu

The screenshot shows the AccuKnox interface. On the left is a dark sidebar with a search bar and a list of navigation items: Dashboard, Inventory, Issues, Findings (highlighted with a red box), Registry Scan, Risk-based Prioritization, Compliance, Runtime Protection, Collectors, Remediation, Monitors / Alerts, and Reports. Below Reports is a button for 'Ask Ada BETA'. At the bottom of the sidebar is a 'Getting started: Onboarding' section with 'Cloud Accounts' and 'Clusters'.

The main content area is titled 'Findings' and has a breadcrumb 'Home > Issues > Findings'. It features a filter bar with 'IAC Scan' selected in a dropdown menu (highlighted with a red box), an 'Asset' dropdown, and three more dropdowns for 'Risk factor', 'Status', and 'Asset Type'. A 'Reset' button is on the left and an 'Edit' button is on the right.

Below the filter bar is a 'Default Configuration' section with a search bar and a 'Hide Default Config' button. It lists 'Static Code Analysis', 'Cloud Misconfiguration', and 'CIS Kubernetes Benchmarks v1.23' with expand/collapse icons. There are also 'Ticket Configuration' and 'Group by' dropdowns, and a row of icons for actions like edit, share, tag, etc.

A 'COLUMNS' button is visible above a table. The table has the following structure:

<input type="checkbox"/>	Last seen	Findings	Asset	Status	Data type	Exploit
<input type="checkbox"/>	2024-03-10	Ensure DB instan...	chirag8680006/t...	Active	IAC Scan	False

- To get a detailed view of a finding click on the finding and then click on the arrow icon.

The image shows a security finding detail page on the left and a configuration modal on the right. A red arrow points from the modal back to the main page.

Main Page:

- Home > Issues > Findings > Details
- Create Collectors
- Ticket Confi... | Create Ticket +
- Ensure the S3 bucket has access logging enabled**
- Severity: Not_available
- Status: Active
- Exploitability: False
- Discovered: 2 Minutes Ago
- Description: <https://github.com/accuknox/cloud-docs/tree/main/docs/en/enterprise-edition/policy-reference/aws-policies/s3-policies/s3-13-enable-logging.adoc>
- Solution: <https://github.com/accuknox/cloud-docs/tree/main/docs/en/enterprise-edition/policy-reference/aws-policies/s3-policies/s3-13-enable-logging.adoc>
- Ticket Comments: 0 comments available
- Show comments

Configuration Modal:

- Ensure the S3 bucket has access logging enabled
- Asset: terraform-aws-example:None
- Asset Type: github-repository
- Location: terraform-aws-example/blob/None/main.tf#L1-L10
- Status: Active
- Ignored: No
- Tickets: 0
- Severity: Not Available
- Ticket Confi... | Save

- This security group is misconfigured and allows ingress connections from any IP in the world to the port 80

Ensure no security groups allow ingress from 0.0.0.0:0 to port 80 Low


Description Result Solution References Source Code

Allowing ingress from 0.0.0.0/0 to port 80 (i.e. the HTTP port) can expose your Amazon Web Services (AWS) resources to potential security threats. This is because 0.0.0.0/0 represents all IP addresses, and allowing traffic from all IP addresses to port 80 can make it easier for attackers to access your resources. By ensuring that your AWS security groups do not allow ingress from 0.0.0.0/0 to port 80, you can help protect your resources from potential attacks and unauthorized access. Instead, you should specify the IP addresses or ranges of IP addresses that are allowed to access your resources, and only allow traffic from those sources. [Show Less...](#)

Details [+ Create Ticket](#)

Asset
Testing_Script.t


Asset Type
IaC_IAC-Repository

Status 

● **Active**

Ignored

No

Severity 

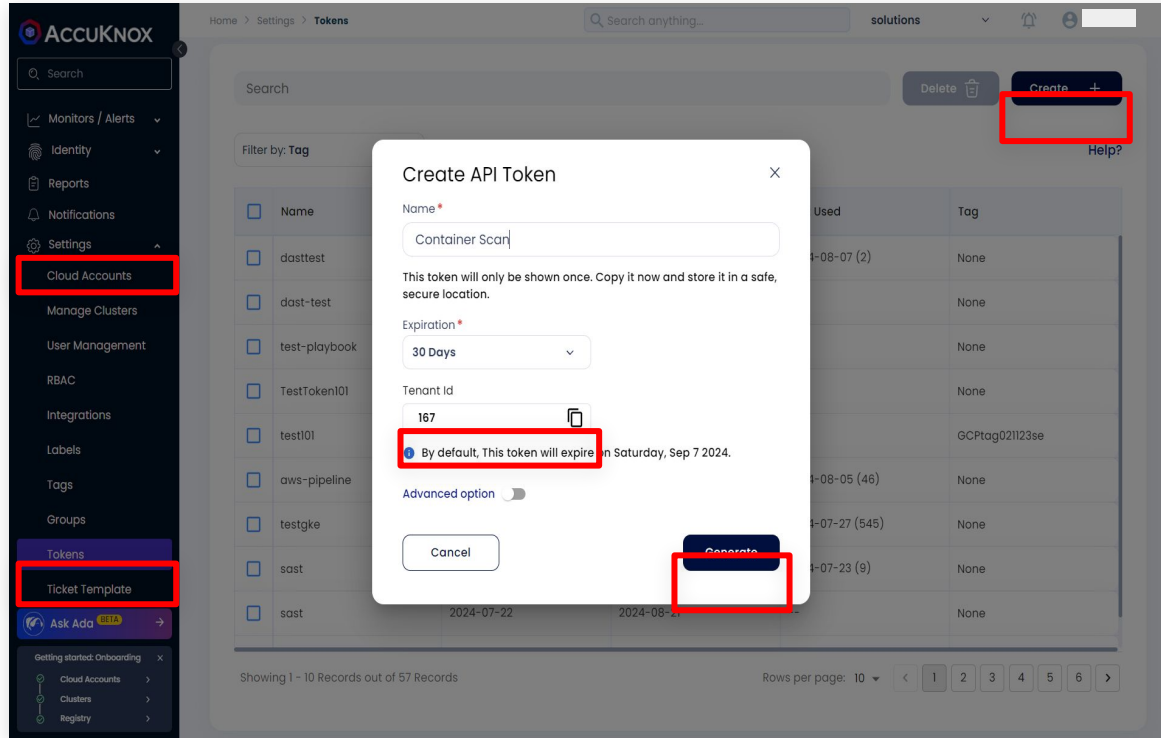


SAST Scan

ASPM Integration

Generate AccuKnox API token for CI/CD pipeline

- To generate a token, open AccuKnox and navigate to **Settings > Tokens > Create**.
- Copy the token and tenant ID, then configure them as secrets in your CI/CD pipeline.

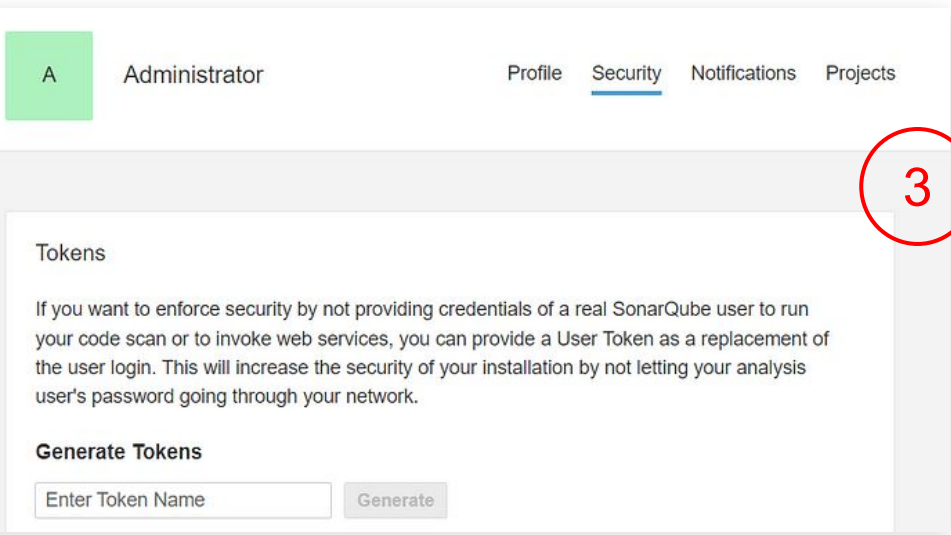
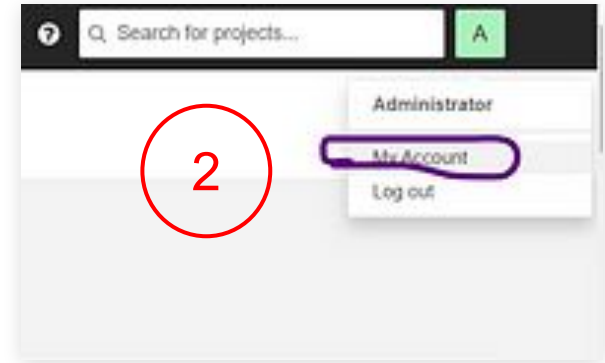


The screenshot displays the AccuKnox web interface. On the left sidebar, the 'Settings' menu is expanded, with 'Cloud Accounts' and 'Ticket Template' highlighted by red boxes. The main content area shows the 'Tokens' page, where a 'Create +' button is also highlighted with a red box. A modal dialog titled 'Create API Token' is open in the center. The dialog contains the following fields and options:

- Name***: A text input field containing 'Container Scan'.
- Expiration***: A dropdown menu set to '30 Days'.
- Tenant Id**: A text input field containing '167', with a copy icon to its right.
- A warning message: 'By default, This token will expire on Saturday, Sep 7 2024.' (The entire message is highlighted with a red box).
- Advanced option**: A toggle switch currently turned off.
- Buttons: 'Cancel' and 'Generate' (the 'Generate' button is highlighted with a red box).

The background shows a table of existing tokens with columns for Name, Used, and Tag. The table lists several tokens, including 'dasstest', 'test-playbook', 'TestTokenID1', 'test101', 'aws-pipeline', 'testgke', 'sast', and 'sast'.

- Deploy a SonarQube VM. Refer this [guide here](#).
- Create a `sonar-project.properties` file into your GitHub repository.
- To generate a SonarQube token, go to SonarQube > My Account > security and click on generate button.



```
# Required metadata
sonar.projectKey=github_sonar_example
sonar.projectName=GitHub Sonar Example
sonar.projectVersion=1.0
# Path to source directories (required)
sonar.sources=.
# Encoding of the source files
sonar.sourceEncoding=UTF-8
# Additional settings
sonar.host.url=<http://your_sonarqube_server:9000>
sonar.login=your_project_token
```

A red circle with the number "1" is drawn around the first line of the code block.

Configuring the SAST scan in GitHub actions

Step 1: Add these steps to your GitHub workflow.

Step 2: Configure these Parameters as GitHub Secrets `SONAR_TOKEN`, `SQ_URL`, `SQ_PROJECTS`, `AK_URL`, `TENANT_ID`, `ACCUKNOX_TOKEN`

Step 3: Run the Workflow

- Push your changes to trigger the workflow, or manually run it from the "Actions" tab in your repository.

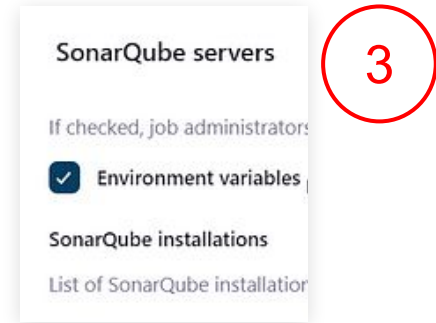
Step 4: Review Findings in AccuKnox

- Log in to your AccuKnox dashboard and navigate to the Issues section.
- Go to the "Findings" tab and select SAST Findings.
- Click on any finding that interests you to view detailed information and recommendations.

```
jobs:
  sonarqube_sast:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
      - uses: sonarsource/sonarqube-scan-action@master
        env:
          SONAR_TOKEN: ${ secrets.SONAR_TOKEN }
          SONAR_HOST_URL: ${ secrets.SQ_URL }
        - name: Run AccuKnox SAST job
          run: |
            docker run --rm \
              -e SQ_URL=${ secrets.SQ_URL } \
              -e SQ_AUTH_TOKEN=${ secrets.SONAR_TOKEN } \
              -e REPORT_PATH=/app/data/ \
              -e SQ_PROJECTS="^github_sonar_example$" \
              -v $PWD:/app/data/ \
              accuknox/sastjob:latest
        - name: Upload SAST reports
          env:
            run: |
              cd ${GITHUB_WORKSPACE}
              for file in `ls -1 SQ-*.json`; do
                curl --location --request POST "<https://$AK_URL/api/v1/artifact/?tenant_id=${ secrets.tenat_id }&data_type=SQ&save_to_s3=false>" \
                  --header "Tenant-Id: ${ secrets.tenat_id }" \
                  --header "Authorization: ${ secrets.accuknox_token }" \
                  --form "file=@\"$file\"
              done
```

Configuring the SAST scan in Jenkins

- Go to Manage Jenkins > Tools and add the SonarQube installation details.
- Create SonarQube credentials.
- Go to Manage Jenkins > System Configurations. Select the check-box of Injecting Environment variables and add the details of the SonarQube Server



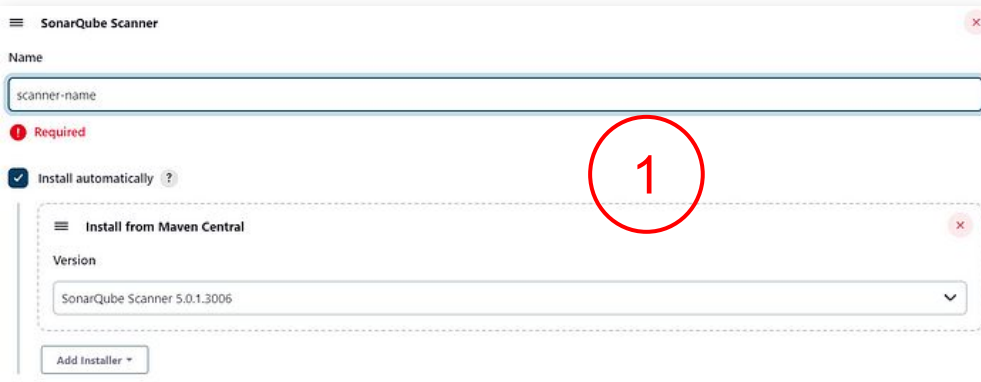
SonarQube servers **3**

If checked, job administrators

Environment variables

SonarQube installations

List of SonarQube installation



SonarQube Scanner

Name

scanner-name **1**

Required

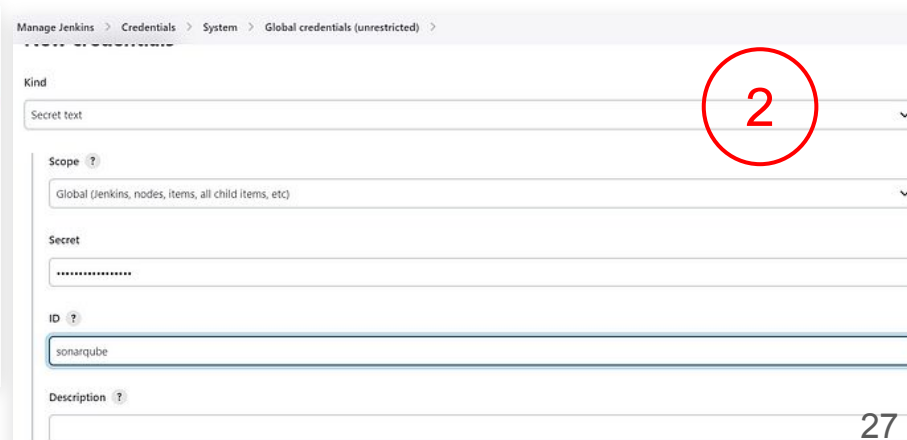
Install automatically ?

Install from Maven Central

Version

SonarQube Scanner 5.0.1.3006

Add Installer



Manage Jenkins > Credentials > System > Global credentials (unrestricted)

Kind **2**

Secret text

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID ?

sonarqube

Description ?

- Go to Manage Jenkins > Tools and add the SonarQube installation details.
- Create SonarQube credentials.
- Go to Manage Jenkins > System Configurations. Select the check-box of Injecting Environment variables and add the details of the SonarQube Server

```

pipeline {
  agent any
  environment {
    SONAR_TOKEN = credentials('sonar-token') // Replace with your Jenkins credential ID
  }
  for SonarQube token
  for SonarQube URL
  for SonarQube URL
  for AccuKnox URL
  for Tenant ID
  for AccuKnox token
  stages {
    stage('Checkout Code') {
      steps {
        checkout([$Class: 'GitSCM', branches: [[name: '*/main']],
doGenerateSubmoduleConfigurations: false, extensions: [[Class: 'CloneOption', depth: 0]],
userRemoteConfigs: [[url: 'YOUR_GIT_REPO_URL']]]) // Replace with your Git repository URL
      }
    }
    stage('SonarQube Analysis') {
      steps {
        withSonarQubeEnv('SonarQube') { // Replace 'SonarQube' with your SonarQube
server configuration name in Jenkins
          sh """
            sonar-scanner \
            -Dsonar.projectKey=your_project_key \
            -Dsonar.sources=. \
            -Dsonar.host.url=${SQ_URL} \
            -Dsonar.login=${SONAR_TOKEN}
          """
        }
      }
    }
    stage('Run AccuKnox SAST') {
      steps {
        sh """
          docker run --rm \
          -e SQ_URL=${SQ_URL} \
          -e SQ_AUTH_TOKEN=${SONAR_TOKEN} \
          -e REPORT_PATH=/app/data/ \
          -e SQ_PROJECTS="^AccuKnox-Sonarqube-Usecase$" \
          -v \$(pwd):/app/data/ \
          accuknox/sastjob:latest
        """
      }
    }
    stage('Upload SAST Reports') {
      steps {
        script {
          def files = sh(script: 'ls -1 SQ-*.json', returnStdout:
true).trim().split('\n')
          for (file in files) {
            sh """
              curl --location --request POST
              "https://${AK_URL}/api/v1/artifact/?tenant_id=${TENANT_ID}&data_type=SQ&save_to_s3=false" \
              --header "Tenant-Id: ${TENANT_ID}" \
              --header "Authorization: Bearer ${AK_TOK}" \
              --form "file=@${file}"
            """
          }
        }
      }
    }
  }
}

```

- Trigger the workflow, go to the AccuKnox > Findings and select the Static Code Analysis findings here.

The screenshot displays the AccuKnox interface. On the left is a dark sidebar with navigation options: Dashboard, Inventory, Issues, Findings (highlighted in purple), Registry Scan, Compliance, Runtime Protection, Remediation, Monitors / Alerts, Identity, Reports, Notifications, and Settings. At the bottom of the sidebar is a 'Getting started: Onboarding' section with links for Cloud Accounts, Clusters, and Registry. The main content area is titled 'Home > Issues > Findings'. A search bar is at the top with the text 'Search anything...'. Below the search bar are filters for 'Asset' and 'Group by'. A dropdown menu is open, showing 'Static Code Analysis Findings' selected and highlighted with a red box. The main table lists findings with columns: Assetname, Name, Risk factor, Description, and Status. The table contains several rows of findings, including 'Use the opposite opera...', 'Using http protocol is in...', and 'Define a constant inste...'. At the bottom, it shows 'Total Records: 348' and a pagination control with page numbers 1 through 18.

	Assetname	Name	Risk factor	Description	Status	
	udit-uniyal_Awesome...	Use the opposite opera...	Low	Why is this an issue? It is n...	Active	
	udit-uniyal_Awesome...	Using http protocol is in...	Low	Clear-text protocols such c...	Active	
	udit-uniyal_Awesome...	Define a constant inste...	Critical	Why is this an issue? Duplic...	Active	
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Make sure that using thi...	Medium	Using pseudorandom num...	Active
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Using http protocol is in...	Low	Clear-text protocols such c...	Active
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Using http protocol is in...	Low	Clear-text protocols such c...	Active
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Using http protocol is in...	Low	Clear-text protocols such c...	Active
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Using http protocol is in...	Low	Clear-text protocols such c...	Active
<input type="checkbox"/>	2024-07-23 09:15:47	udit-uniyal_Awesome...	Make sure that using thi...	Medium	Using pseudorandom num...	Active

- Use case 1: Privilege escalation
- Use case 2: XML External Entity (XXE) injection
- Use case 3: Hard coded password
- Use case 4: Cross Site Request Forgery (CSRF)
- Use case 5: Remote Code Execution (RCE)

Use case 1: Privilege escalation

- An IAM policy in this code is vulnerable to privilege escalation attack.
- AccuKnox identifies this vulnerability and suggests a solution.

This policy is vulnerable to the "EC2" privilege escalation vector. Remove permissions or restrict the set of resources they apply to. Critical [🔗](#) ✕

Description	Result	Solution	References	Source Code
<p><p>Within IAM, identity-based policies grant permissions to users, groups, or roles, and enable specific actions to be performed on designated resources. When an identity policy inadvertently grants more privileges than intended, certain us</p> <p>Show More...</p>				

Details [+ Create Ticket](#)

Asset
gitlab-sast-testing

Asset Type
static_code_Software

Status [✎](#)

Use case 2: XML External Entity (XXE) injection

- This code have an XML parsing vulnerability XML External Entity injection.
- AccuKnox identifies the vulnerability and proposes a solution.

Disable access to external entities in XML parsing. critical [🔗](#) ✕

Description	Result	Solution	References	Source Code
<p><p>This vulnerability allows the usage of external entities in XML.</p> <h2>Why is this an issue?</h2> <p>External Entity Processing allows for XML parsing with the involvement of external entities. However, when this functionality is enabl Show More...</p>				

Details [+ Create Ticket](#)

Asset
test-vulnerable-code-snippets

Asset Type
static_code_Software

Status [✎](#)

Use case 3: Hard coded password

- There is a hardcoded password in the source code.
- AccuKnox identifies the vulnerability, and proposes a solution.

Detected 'password' in this variable name, review this potentially hardcoded credential. Critical


Description	Result	Solution	References	Source Code
-------------	--------	----------	------------	-------------

```
2 // Exercise - 1
3 // Author: @TheXC3LL
4 // Website: Tarlogic.com
5 class login {
6     public $username = "X-C3LL";
7     public $password = "Insanity";
8     public $role = "MUGGLE";
9 }
10 $one = new login();
11 $a = serialize($one);
12 echo "Example of an object:\n$a\n\n";
13 echo "FLAG: \n";
14 $test = unserialize($argv[1]);
15 $check = $test->role - 1337;
16 if ($check == "ADMIN") {
```

Details [+ Create Ticket](#)


Asset
test-vulnerable-code-snippets

Asset Type
static_code_Software

Status 

● Active

Ignored
 No

Severity 

Use case 4: Cross Site Request Forgery (CSRF)

- This is a CSRF vulnerability. Here an attacker can send arbitrary HTTP or HTTPS requests behalf of this web server.
- AccuKnox identifies this critical vulnerability and proposes a solution.

Make sure disabling CSRF protection is safe here. Critical [🔗](#) ✕

Description	Result	Solution	References	Source Code
<p>A cross-site request forgery (CSRF) attack occurs when a trusted user of a web application can be forced, by an attacker, to perform sensitive actions that he didn't intend, such as updating his profile or sending a message, more generally anything that can change the state of the application.</p> <p>The attacker can trick the user/victim to click on a link, corresponding to the privileged action, or to visit a malicious web site that embeds a hidden web request and as web browsers automatically include cookies, the actions can be authenticated and sensitive.</p> Show Less...				

Details [+ Create Ticket](#)

Asset
test-vulnerable-code-snippets

Asset Type
static_code_Software

Status [✎](#)

● Active

Ignored

No

Use case 5: Remote Code Execution (RCE)

- This code uses the `eval()` function, which is vulnerable to RCE.
- AccuKnox identifies the vulnerability within the source code and suggests a solution.

Make sure that this dynamic injection or execution of code is safe.

Medium



Description Result Solution References Source Code

```
7 $empty = 'No variable given';
8
9 // pass the variable name into an eval block, making it
10 // vulnerable to Remote Code Execution (rce). This RCE
11 // is NOT blind.
12 eval('echo $' . $variable . ';');
13
```

Details

[+ Create Ticket](#)

Asset

test-vulnerable-code-snippets

Asset Type

static_code_Software

Status 

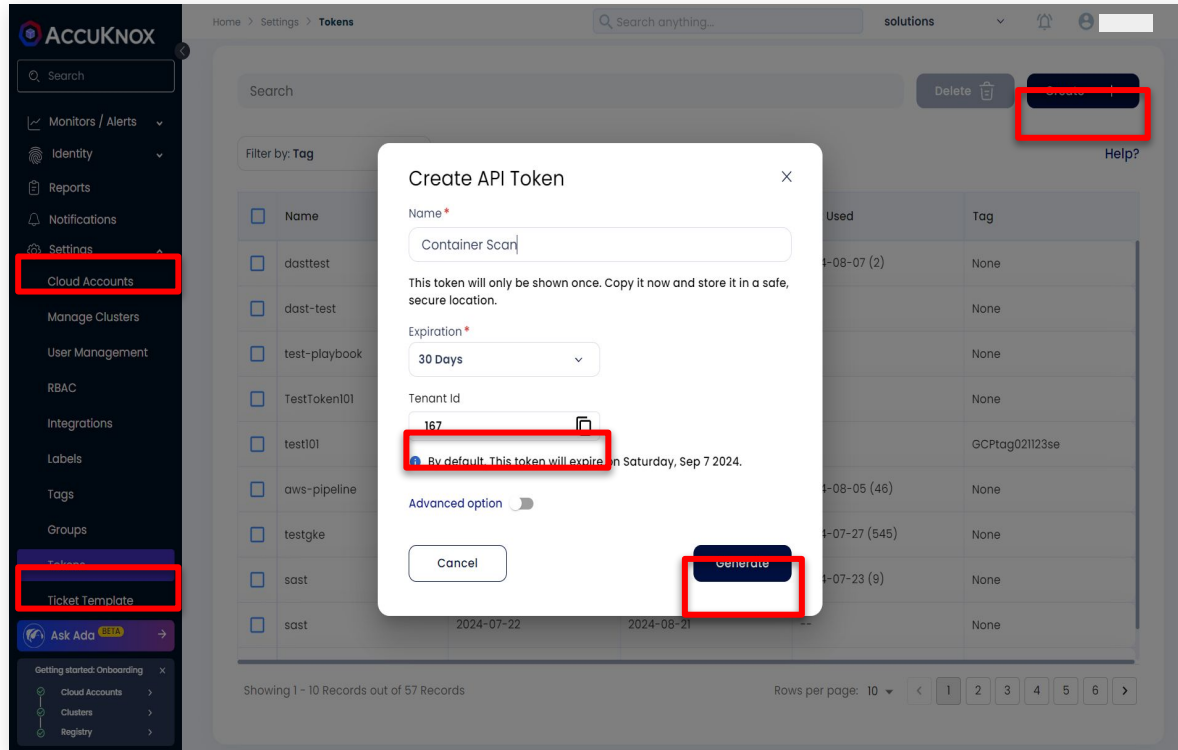


DAST Scan

ASPM Integration

Generate AccuKnox API token for CI/CD pipeline

- To generate a token, open AccuKnox and navigate to **Settings > Tokens > Create**.
- Copy the token and tenant ID, then configure them as secrets in your CI/CD pipeline.



The screenshot displays the AccuKnox web interface. On the left sidebar, the 'Settings' menu is expanded, and 'Cloud Accounts' is highlighted with a red box. In the main content area, the 'Tokens' page is visible, and the 'Create' button is highlighted with a red box. A modal dialog titled 'Create API Token' is open in the center. The dialog contains the following fields and options:

- Name ***: Container Scan
- Expiration ***: 30 Days
- Tenant Id**: 167

A warning message below the Tenant Id field states: "By default, this token will expire on Saturday, Sep 7 2024." The 'Generate' button at the bottom right of the dialog is highlighted with a red box. The background shows a table of existing tokens with columns for Name, Used, and Tag.

Configuring the DAST scan in GitHub actions

Step 1: Add these steps to your GitHub workflow.

Step 2: Run the Workflow

- Push your changes to trigger the workflow, or manually run it from the "Actions" tab in your repository.

Step 3: Review Findings in AccuKnox

- Log in to your AccuKnox dashboard and navigate to the Issues section.

- Go to the "Findings" tab and select DAST Findings.
- Click on any finding that interests you to view detailed information and recommendations.

```
jobs:
  zap:
    runs-on: ubuntu-latest
    steps:
      - name: set permissions
        run: sudo chmod -R 777 .

      - name: workspace permissions
        run: sudo chmod -R 777 ${github.workspace}

      - name: zap scan
        run: |
          docker run --rm -v ${github.workspace}:/zap/wrk/:rw -t zaproxy/zap-stable zap-
          baseline.py \
            -t <url that you want to scan> \
            -J report.json
            -I

      - name: zap
        run: |
          cd ${GITHUB_WORKSPACE}
          curl --location --request POST 'https://${secrets.accuknox_url}
          }}/api/v1/artifact/?tenant_id=${secrets.tenant_id}
          }}&data_type=ZAP&label_id=dasttest&save_to_s3=false' \
            --header 'Tenant-Id: ${secrets.tenant_id}' \
            --header 'Authorization: Bearer ${secrets.token}' \
            --form 'file=@\report.json'
```

Configuring the DAST scan in Jenkins

Create a New Pipeline:

- Go to the Jenkins dashboard.
- Click on "New Item" to create a new pipeline.
- Name your pipeline and select "Pipeline" from the list, then click "OK."

Configure the Pipeline:

- In the pipeline configuration page, scroll down to the "Pipeline" section.
- Under "Definition," select "Pipeline script" from the dropdown.

Add the Script:

- In the script box that appears, paste your pipeline script.

Save the Configuration:

- After adding the script, click "Save" or "Apply."

Run the Pipeline:

- Go to the newly created pipeline and click "Build Now" to run the script.

```
pipeline {
  agent any
  environment {
    TENANT_ID = credentials('bearer-tenant_id') // Replace with your Jenkins credentials
    ID for the tenant_id
    BEARER_TOKEN = credentials('bearer-token') // Replace with your Jenkins credentials
    ID for the Bearer token
  }
  stages {
    stage('Checkout') {
      steps {
        checkout([$class: 'GitSCM', branches: [[name: '*/main']],
doGenerateSubmoduleConfigurations: false, extensions: [], userRemoteConfigs: [[url:
'YOUR_GIT_REPO_URL']]]) // Replace with your Git repository URL
      }
    }
    stage('Set Workspace Permissions') {
      steps {
        sh "sudo chmod -R 777 $PWD"
      }
    }
    stage('List Files') {
      steps {
        sh 'ls'
      }
    }
    stage('Run ZAP Scan') {
      steps {
        sh """
        docker run --rm -v $PWD:/zap/work/:rw -t \
        zapproxy/zap-stable zap-baseline.py \
        -t <your-website> \
        -J report.json \
        -I
        """
      }
    }
    stage('Upload ZAP Report') {
      steps {
        sh """
        curl --location --request POST
        https://cspm.demo.accuknox.com/api/v1/artifact/?
        tenant_id=${TENANT_ID}&data_type=ZAPLabelId=dasttest&save_to_s3=false" \
        --header "Tenant-Id: ${TENANT_ID}" \
        --header "Authorization: Bearer ${BEARER_TOKEN}" \
        --form "file=@report.json"
        """
      }
    }
  }
}
```

- To get the findings, go to the AccuKnox > Findings and select the DAST here.

The screenshot shows the AccuKnox web interface. On the left is a dark sidebar with navigation options: Dashboard, Inventory, Issues, Findings (highlighted with a red box), Registry Scan, Compliance, Runtime Protection, Remediation, Monitors / Alerts, Identity, Reports, Notifications, and Settings. Below the sidebar is a 'Getting started: Onboarding' section with links for Cloud Accounts, Clusters, and Registry. The main content area is titled 'Home > Issues > Findings'. At the top, there is a search bar and a dropdown menu for 'DAST Findings' (highlighted with a red box). Below this are filters for 'Asset' and 'Group by'. A table of findings is displayed with columns: Name, Assetname, Description, Risk factor, and Location. The table contains several rows of findings, including 'Retrieved from Cache', 'Missing Anti-clickjackin...', 'Storable and Cacheabl...', 'Strict-Transport-Securit...', 'Content Security Policy ...', 'Modern Web Application', and 'Strict-Transport-Securit...'. At the bottom of the table, it says 'Total Records: 558' and a pagination control showing page 1 of 28.

Name	Assetname	Description	Risk factor	Location
Retrieved from Cache	https://app.demo.accu...	<p>The content was ret...	Informational	https://app.demo.accu...
Retrieved from Cache	https://app.demo.accu...	<p>The content was ret...	Informational	https://app.demo.accu...
Missing Anti-clickjackin...	https://app.demo.accu...	<p>The response does ...	Medium	https://app.demo.accu...
Storable and Cacheabl...	https://app.demo.accu...	<p>The response conte...	Informational	https://app.demo.accu...
Retrieved from Cache	https://app.demo.accu...	<p>The content was ret...	Informational	https://app.demo.accu...
Strict-Transport-Securit...	https://app.demo.accu...	<p>HTTP Strict Transpor...	Low	https://app.demo.accu...
Content Security Policy ...	https://app.demo.accu...	<p>Content Security Pol...	Medium	https://app.demo.accu...
Modern Web Application	https://app.demo.accu...	<p>The application app...	Informational	https://app.demo.accu...
Strict-Transport-Securit...	https://app.demo.accu...	<p>HTTP Strict Transpor...	Low	https://app.demo.accu...

- Use case 1: Cross Origin Resource Sharing (CORS) misconfiguration
- Use case 2: File inclusion vulnerability
- Use case 3: Cross site scripting vulnerability
- Use case 4: SQL injection vulnerability
- Use case 5: Missing content security policy

Use case 1: Cross Origin Resource Sharing (CORS) misconfiguration

- This web application is vulnerable to CORS.
- CORS can lead to so many issues such as unauthorized access, cross site scripting, session hijacking and many other issues.
- AccuKnox identifies this vulnerability and proposes the solution.

Cross-Domain Misconfiguration Medium

[Description](#) [Result](#) [Solution](#) [References](#) [Source Code](#) [Details](#) [+ Create Ticket](#)

`<p>Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server.</p>`

Asset
https://juice-shop.herokuapp.com

Asset Type

Use case 2: File inclusion vulnerability

- This web application is vulnerable to file inclusion.
- In file inclusion attack, and attacker can access files stored on a web server.
- This can lead to sensitive data leakage and source code leakage.
- AccuKnox identifies this vulnerability, and proposes a solution.

Source Code Disclosure - File Inclusion High


Description Result Solution References Source Code

<p>The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will exec
[Show More...](#)

Details [+ Create Ticket](#)

Asset
http://testphp.vulnweb.com

Asset Type
WebApp

Status 

Use case 3: Cross site scripting vulnerability

- This web application have a cross site scripting vulnerability (XSS)
- XSS allows an attacker to inject arbitrary javascript code into a webpage
- AccuKnox identifies the vulnerability and proposes a solution to it

Source Code Disclosure - File Inclusion High

Description Result Solution References Source Code


<p>The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will exec

[Show More...](#)

Details [+ Create Ticket](#)

Asset
http://testphp.vulnweb.com

Asset Type
WebApp

Status 


Use case 4: SQL injection


- This web application is vulnerable to SQL injection attacks.
- SQL injection vulnerability allows an attacker to access the database of the website.
- AccuKnox identifies the vulnerability and proposes a solution.


SQL Injection - MySQL High

Description Result Solution References Source Code

<p>SQL injection may be possible.</p>

 Finding for in resource WebApp | <http://testphp.vulnweb.com>


 **Failing since** about 1 month ago, on 21/07/2024

 **Last detected** about 1 month ago, on 21/07/2024

Details + Create Ticket

Asset
http://testphp.vulnweb.com

Asset Type
WebApp

Status 
● Active

Use case 5: Missing content security policy

- This web application don't have a content security policy.
- The CSP adds a layer of security to the application.
- AccuKnox identifies this misconfiguration and proposes a solution.

Content Security Policy (CSP) Header Not Set

Medium



Description

Result

Solution

References

Source Code

Details

[+ Create Ticket](#)

<p>Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data thef

[Show More...](#)

Asset

https://juice-shop.herokuapp.com

Asset Type

WebApp