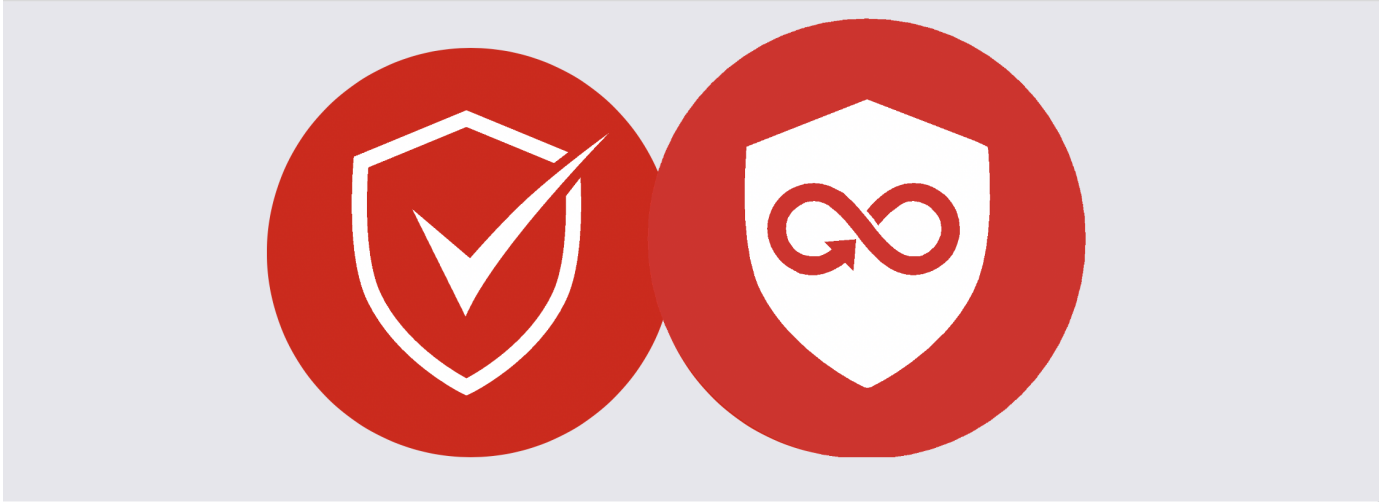
DevSecOps - Automating Security In DevOps



DevSecOps – Automating Security in DevOps

© Copyright 2020 NotSoSecure Global Services Limited - A Claranet Group Company, All Rights Reserved.

Agenda

Introduction to DevOps

- Lab Setup
- Challenges with Traditional IT
- What is DevOps?
- Lab: DevOps using Jenkins Pipeline and Docker

Introduction to DevSecOps

- Challenges for Security in DevOps
- DevOps Threat Model
- DevSecOps – Why, What and How?
- Vulnerability Management

Continuous Integration

- Pre-Commit Hooks
- Secrets Management

Continuous Delivery

- Software Composition Analysis (SCA)
- Static Analysis Security Testing (SAST)
- Dynamic Analysis Security Testing (DAST)

Infrastructure As Code

- Vulnerability Assessment (VA)
- Container Security (CS)
- Compliance as Code (CaC)

Continuous Monitoring

- Alerting and Monitoring

DevSecOps in AWS

- DevOps on Cloud Native AWS
- AWS Threat Landscape
- DevSecOps in Cloud Native AWS

DevSecOps Challenges and Enablers

- Challenges with DevSecOps
- Building DevSecOps Culture
- Security Champions
- Case Studies

Disclaimer

- We will be listing a lot of tools, It’s not an exhaustive list
- We don't endorse or recommend any specific tool / vendor
- Every environment is different: Test and validate before implementing any ideas

Lab Instances

- IDE http://user0-ide.appsecops.dev/
- Jenkins https://user0-jenkins.appsecops.dev
- Archerysec https://user0-archerysec.appsecops.dev/dashboard/
- Mail https://user0-mail.appsecops.dev/
- Vault https://user0-vault.appsecops.dev/
- Kibana https://user0-kibana.appsecops.dev/
- Staging http://user0-staging.appsecops.dev/
- Production https://user0-production.appsecops.dev/prod/home.action
- OpenVAS https://user0-openvas.appsecops.dev/login/login.html
- OWASP ZAP https://user0-zap.appsecops.dev/

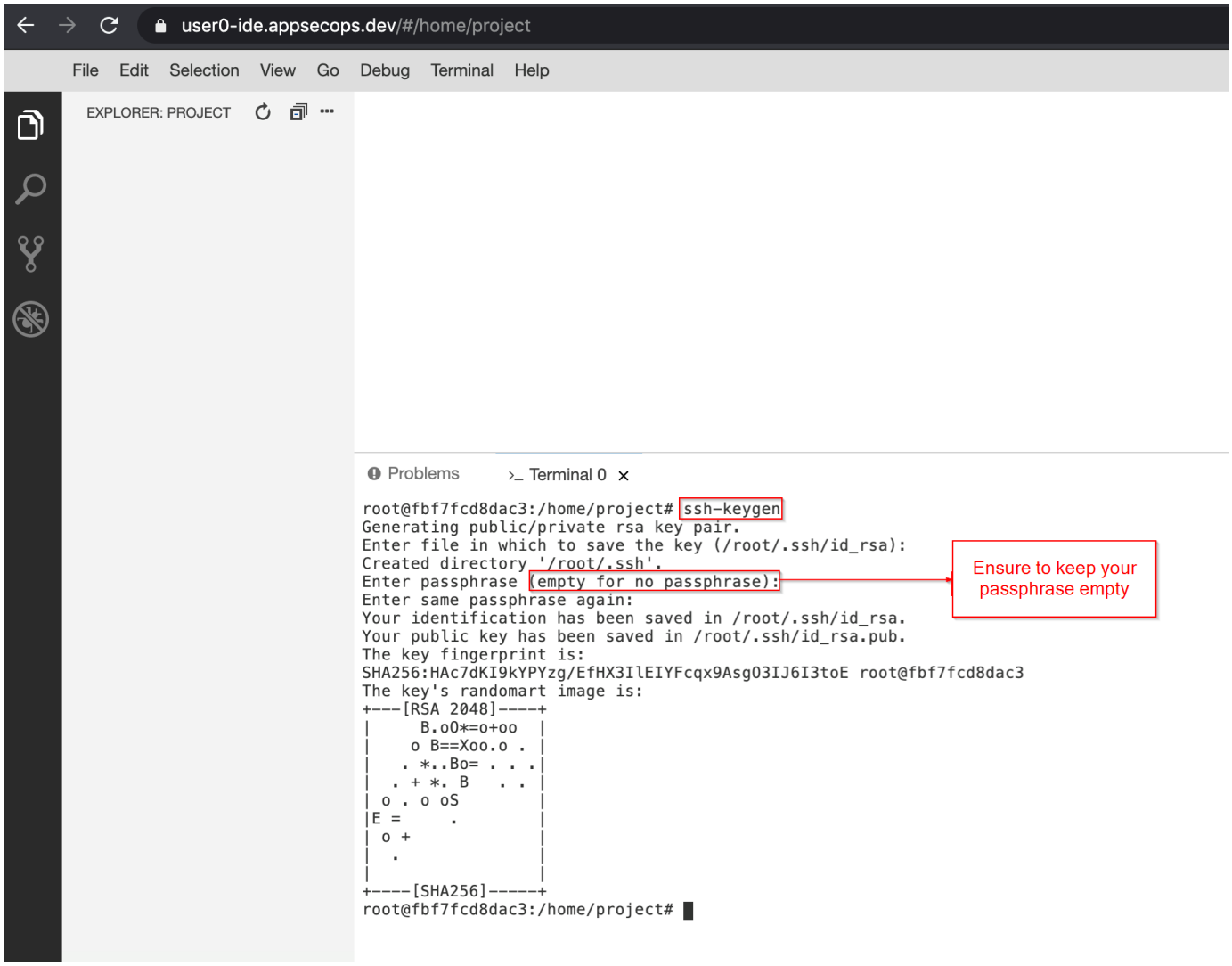
Setup SSH connection

SSH Key gen

Type below command and generate ssh key

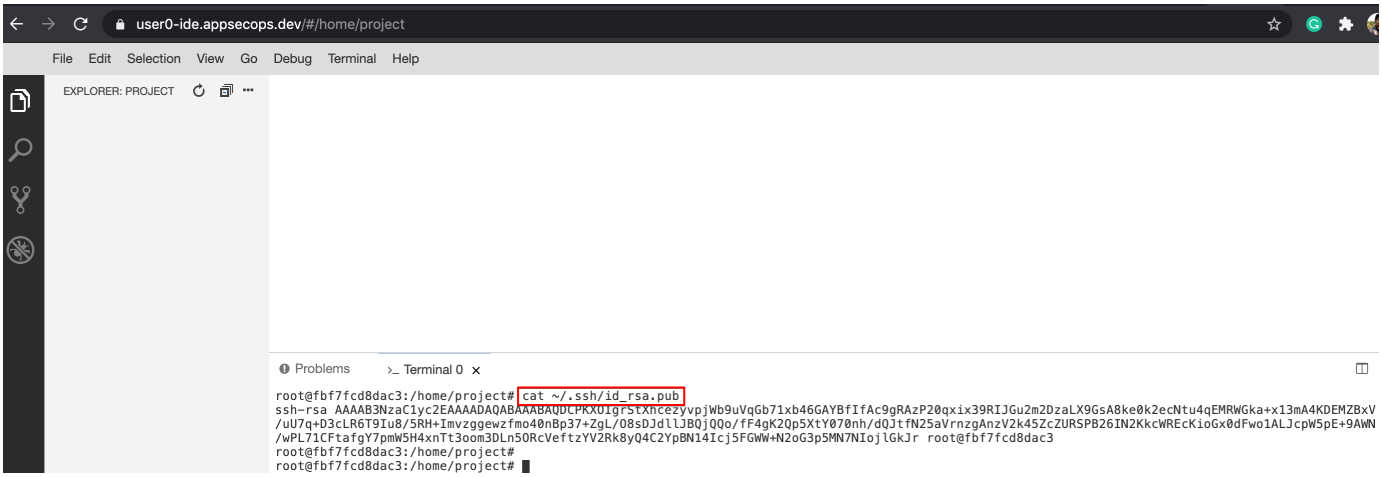
```
ssh-keygen
```

> Note : Do not add passphrase to the public key generated and leave it as blank

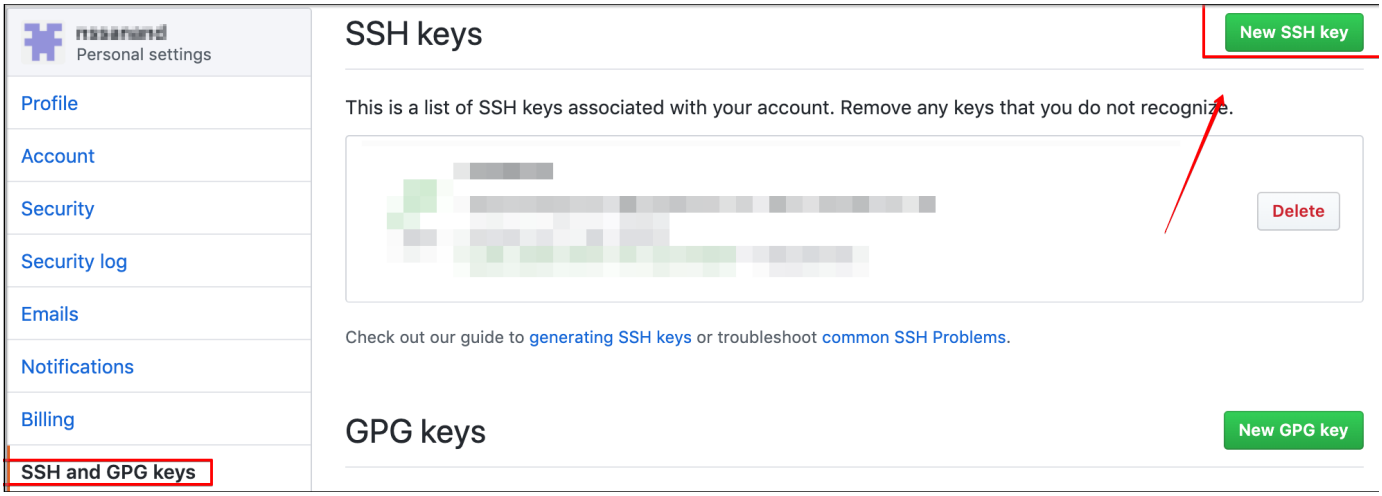


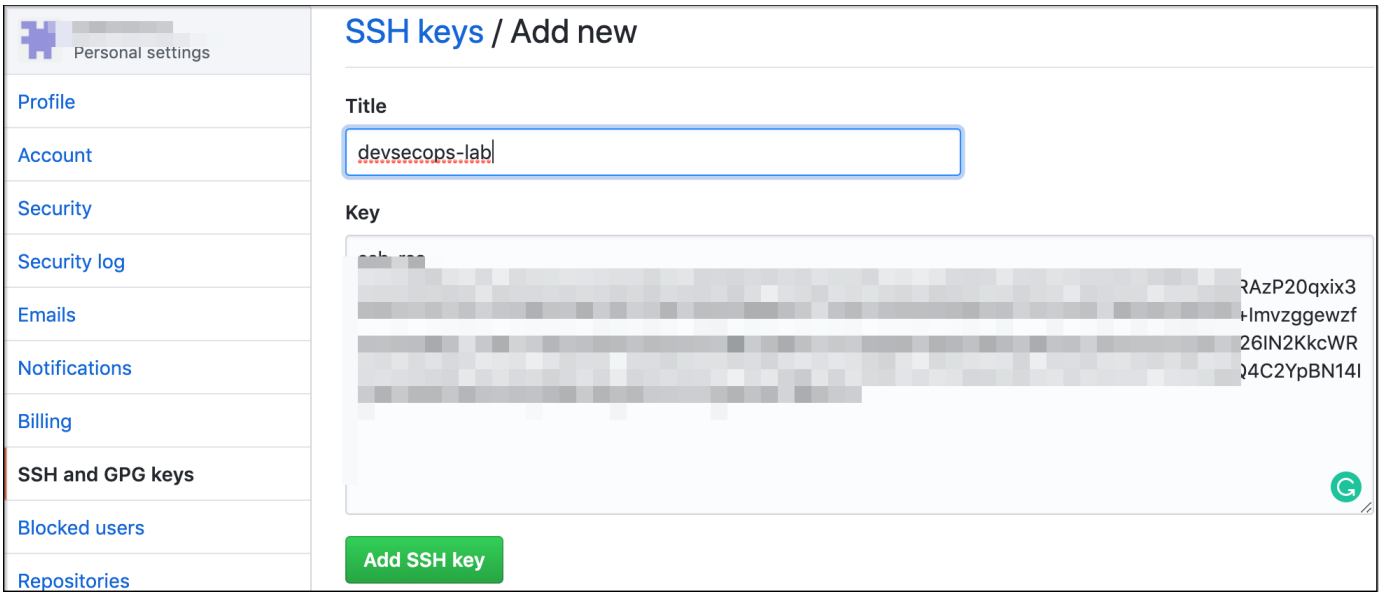
Copy ssh Public Key

```
cat ~/.ssh/id_rsa.pub
```



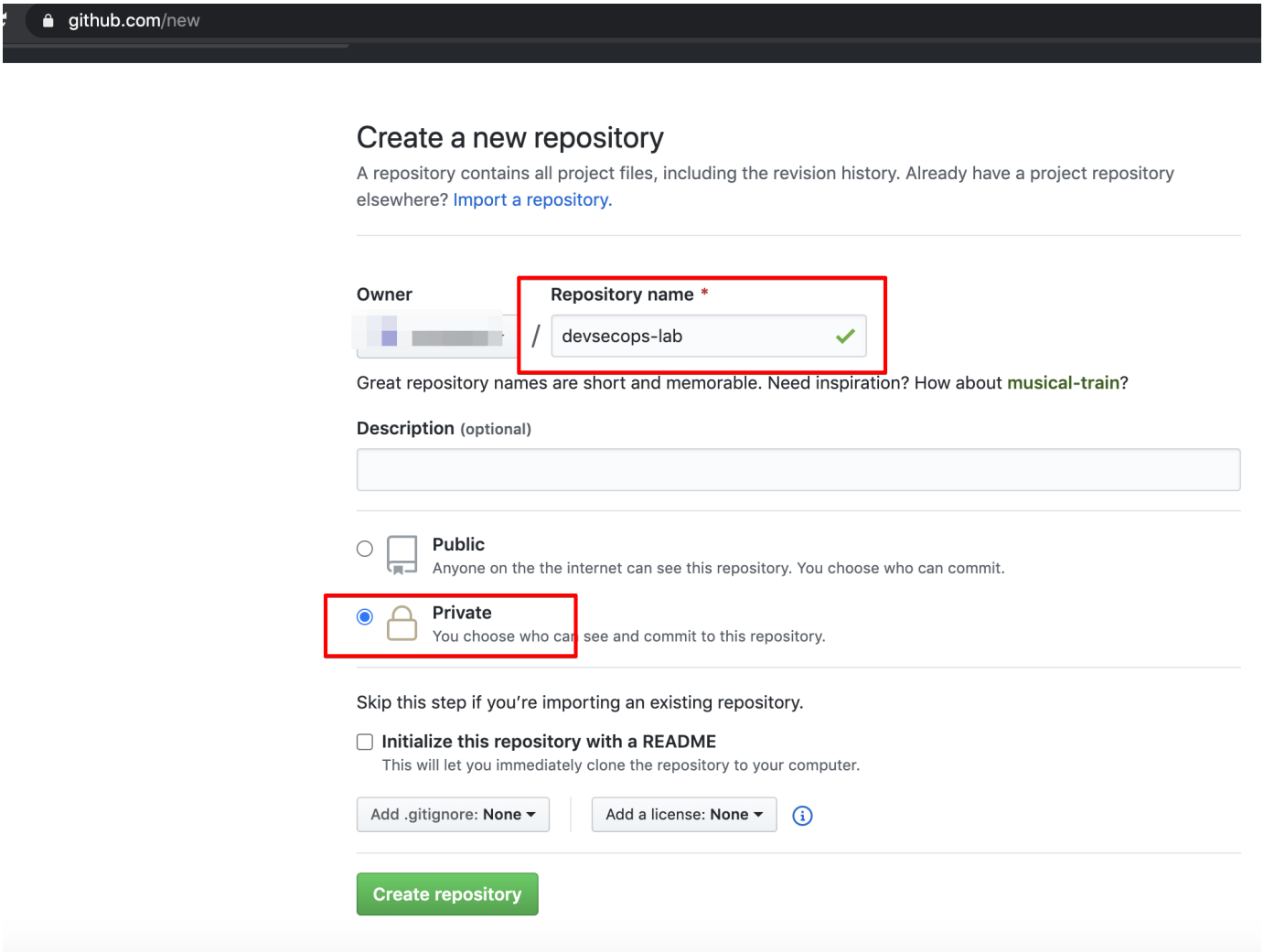
- Go to Github setting page.
- Select SSH and PGP key
- Create New SSH Key
- Upload SSH public key





Create New Repository in GitHub

- Go to your Github account
- Create new repository
- Name as devsecops-lab
- Select as Private repository



Clone Github Repository into IDE

- Go into you IDE
- type below code to clone it

```
git clone git@github.com-YOUR-REPO-NAME
```

```
root@fbf7fcd8dac3:/home/project# git clone git@github.com:[redacted]lab.git
Cloning into 'test_lab'...
The authenticity of host 'github.com (140.82.112.3)' can't be established.
RSA key fingerprint is [redacted]8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,140.82.112.3' (RSA) to the list of known hosts.
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 508 (delta 9), reused 12 (delta 4), pack-reused 490
Receiving objects: 100% (508/508), 2.32 MiB | 0 bytes/s, done.
Resolving deltas: 100% (195/195), done.
root@fbf7fcd8dac3:/home/project#
```

Initial Commit

Provide Vault mysqldb secret into Jenkinsfile

- Replace the Vault mysqldb secret into Jenkinsfile as shown below

```
pipeline {
    agent any
    environment {
      DOCKER_REGISTRY = "localhost:6000"  // Docker registry
      VAULT_PATH_MYSQL="kv/mysql/db"
      VAULT_TOKEN_MYSQL="s.3ZqSnsDurzBOPL17DOBfSI2U" // Replace with generated token
      MYSQL_STAGING_URL="stgmysqldb:3306"
      MYSQL_PRODUCTION_URL="prodmysqldb:3306"
      MYSQL_DB_NAME="test"
      MYSQL_DB_PASSWORD="test"
      MYSQL_DB_USER="test"
      MYSQL_DB_ROOT="tooor"
      ARCHERYSEC_HOST ="http://localhost:8000" // ArcherySec URL
      TARGET_URL="http://localhost:8050" //Staging applicaiton URL
      TARGET_IP="http://localhost:8050"
      OPENVAS_HOST="localhost"
      OPENVAS_USER="admin"
      OPENVAS_PASS="admin"
      OPENVAS_PORT="9390"
    }
    stages {
        stage('Build') {
```

Run below commands

```
git add .
```

```
git config --global user.email "you@example.com"
```

```
git config --global user.name "Your Name"
```

```
git commit -am "initial_commit"
```

```
git push
```

Configure Lab

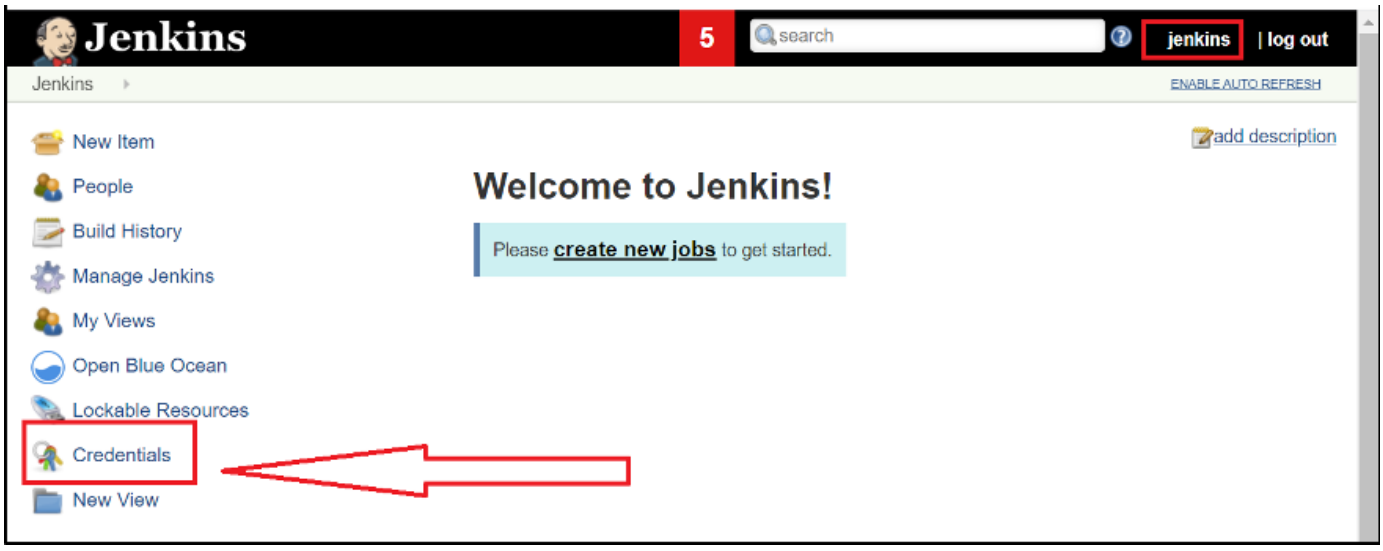
Setup Jenkins Credentials

Login into jenkins server

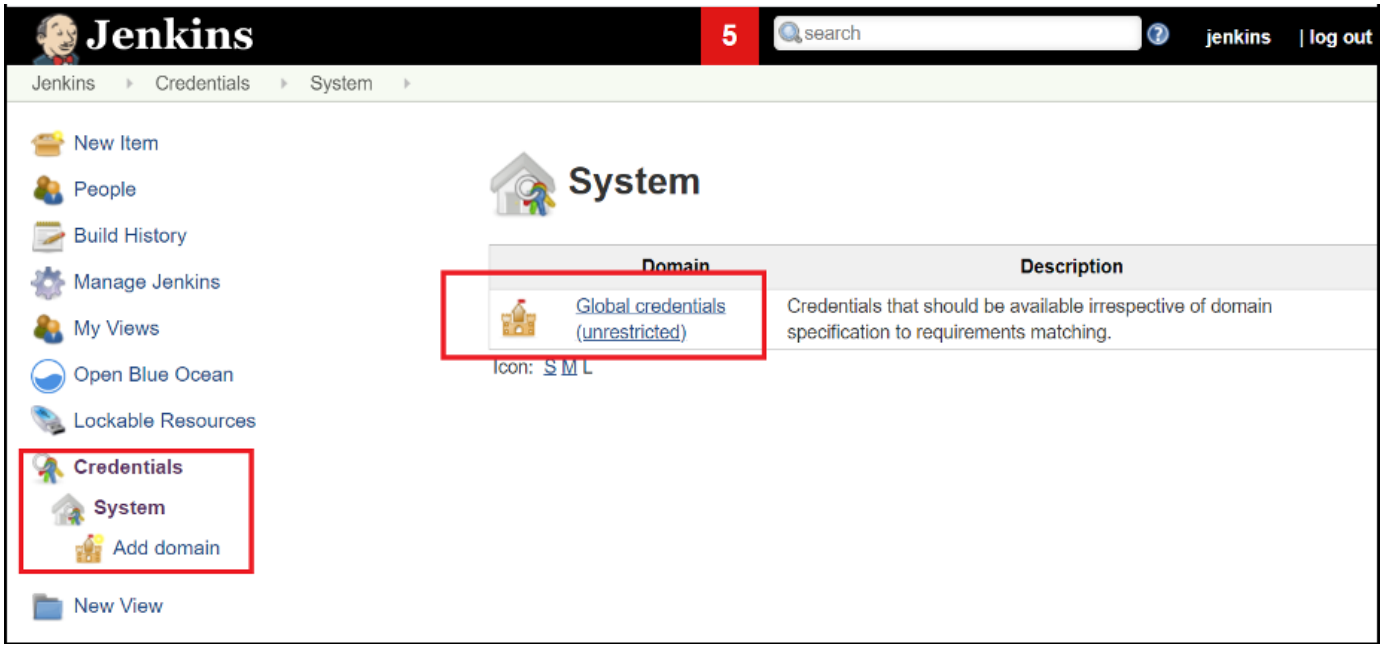
Go to Credentials

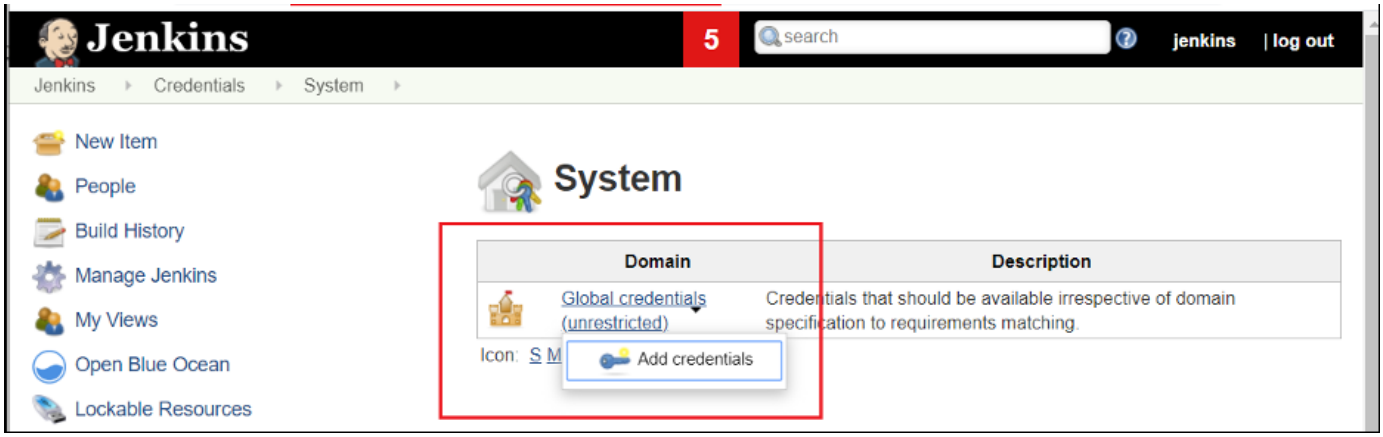
1. Enter the credentials “jenkins:jenkins” and proceed to ‘Sign In’

2. On successful login you would be able to see the dashboard. Click on “Credentials” .

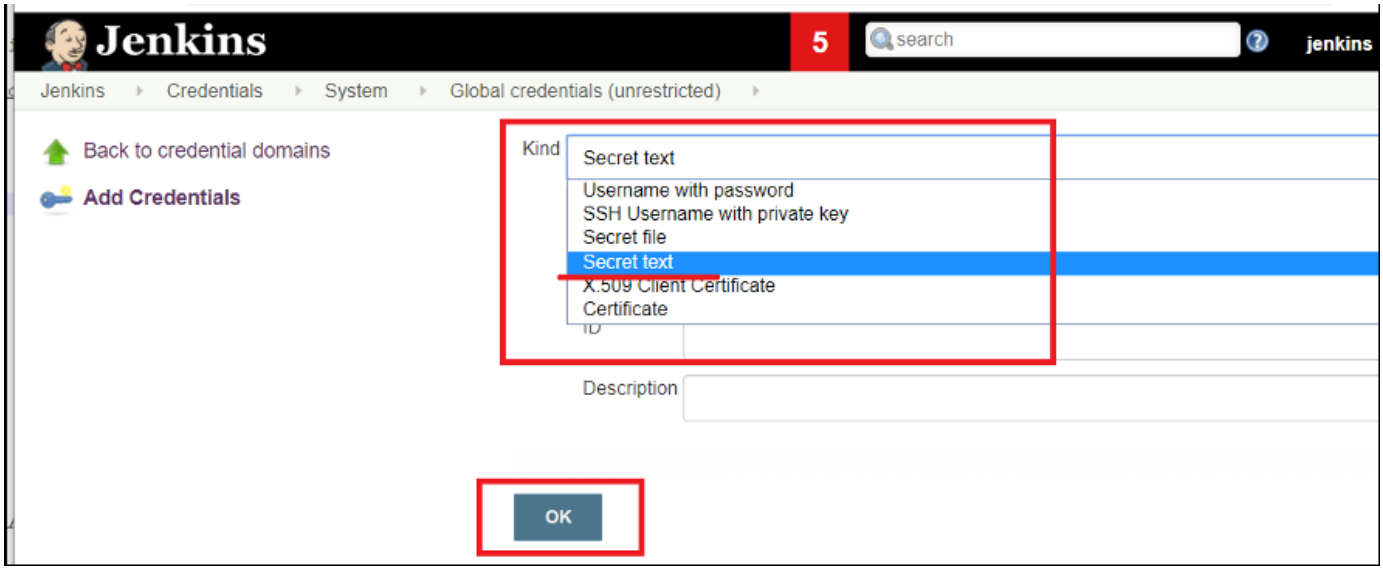


3. Goto “Credentials” > “System” and right-click “Global credentials(unrestricted)” to “Add Credentials”

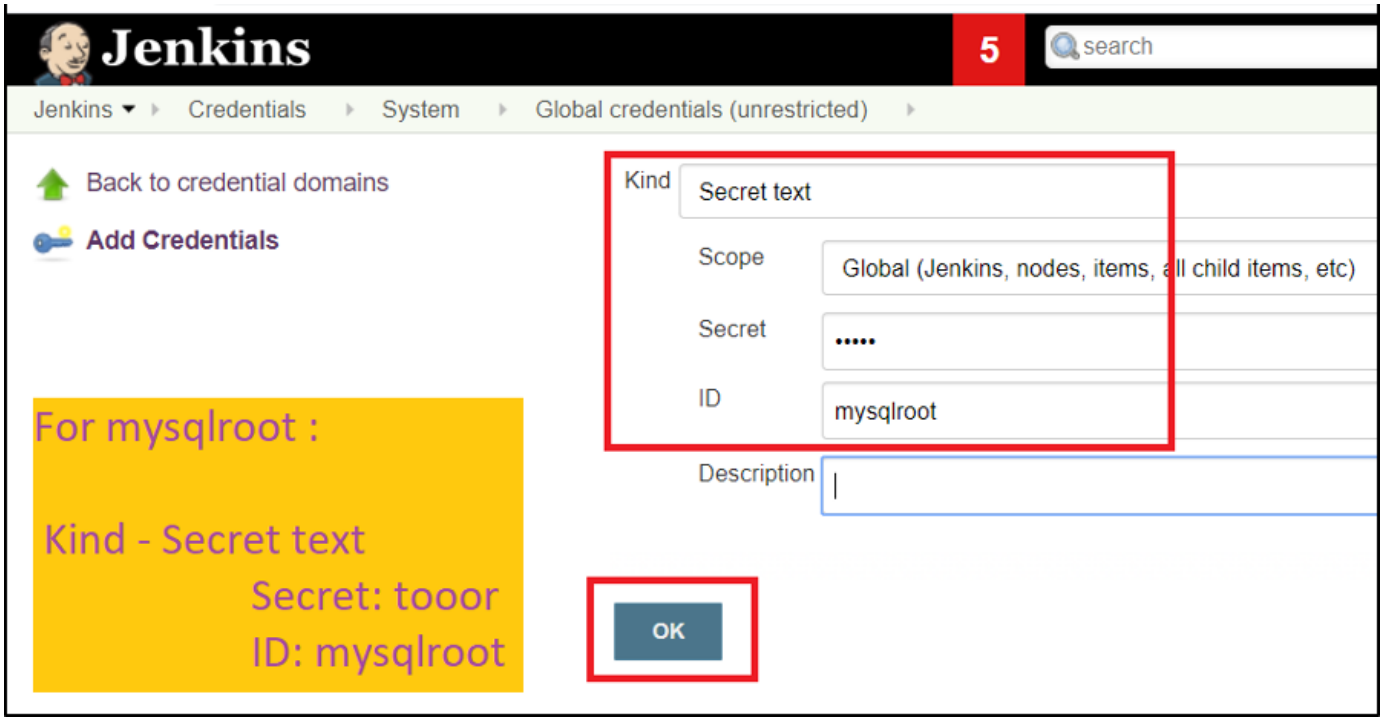




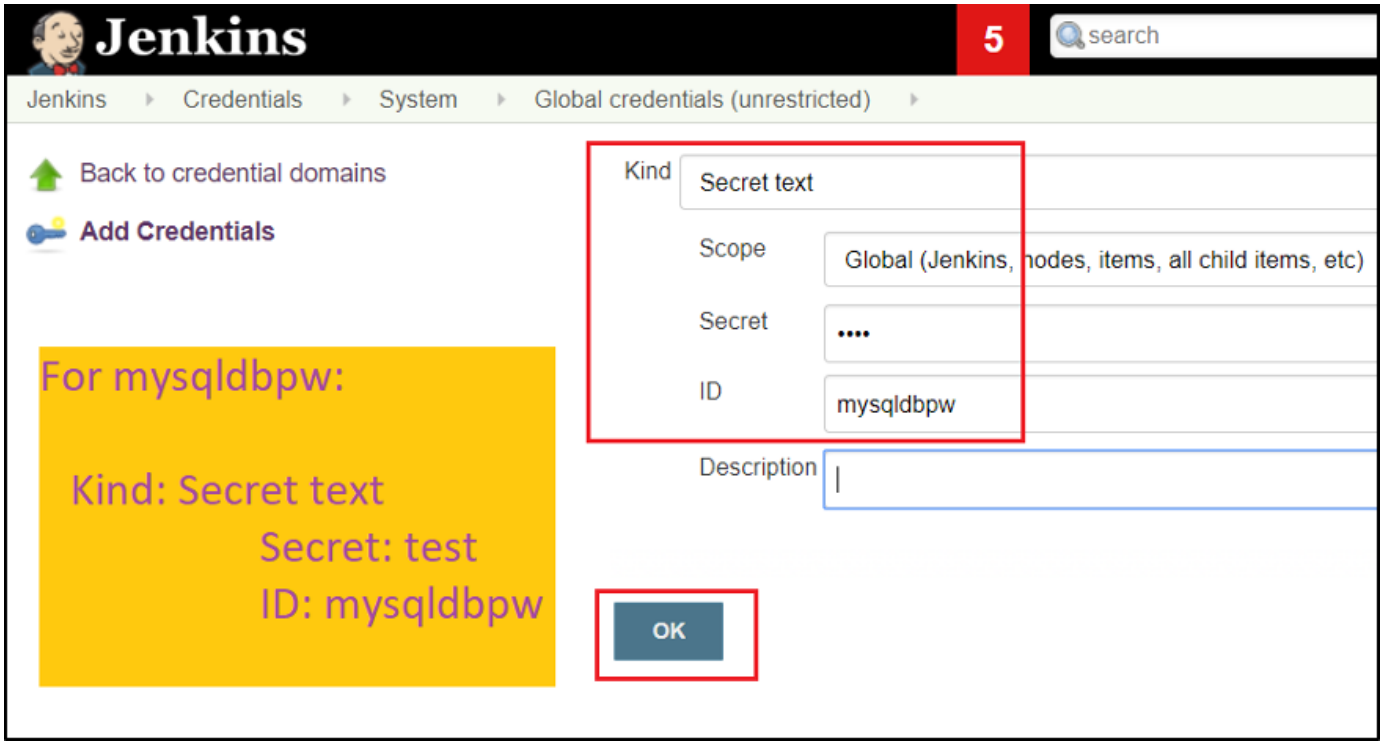
4. Select “Secret text” from the drop-down menu and click “OK” to proceed.

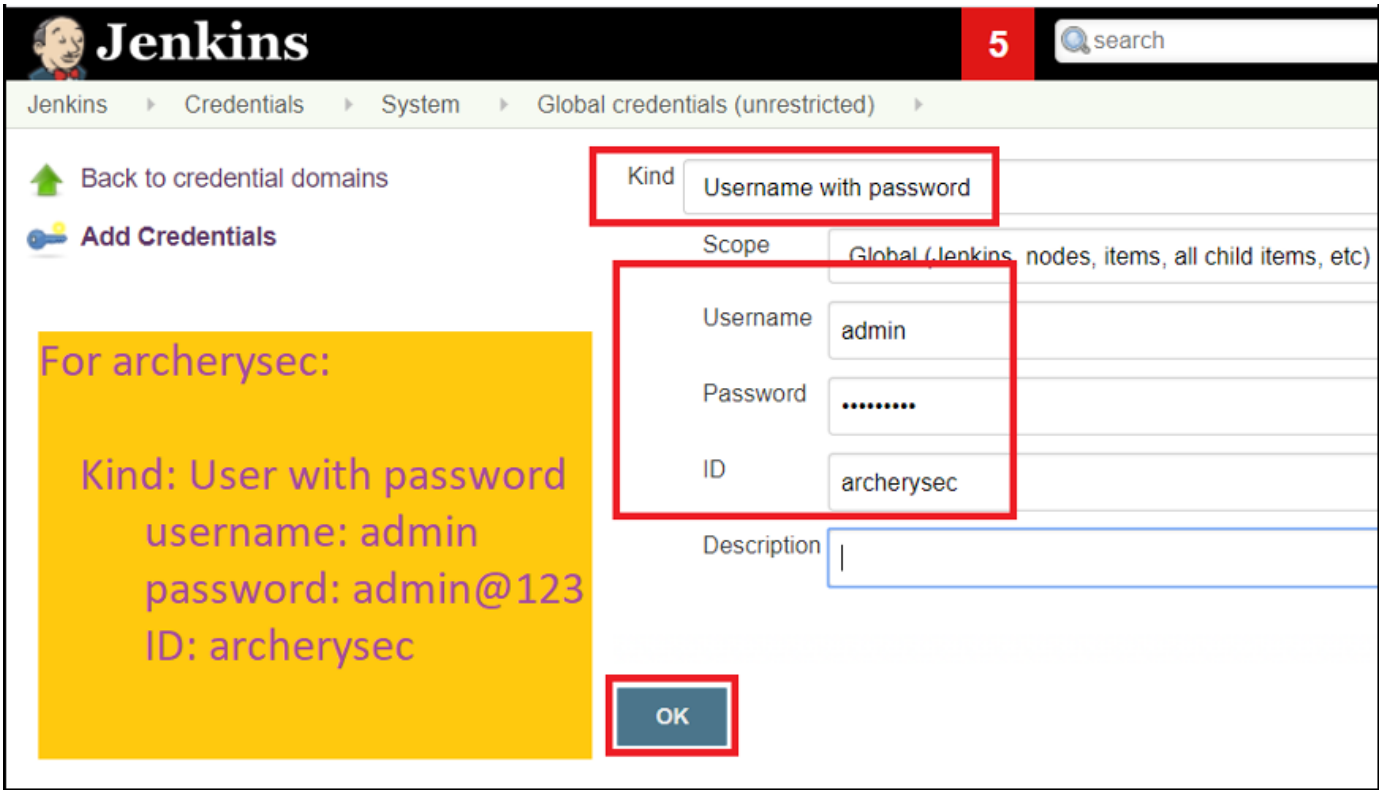


5. Enter Secret: tooor , ID: mysqlroot and Click “OK”.



6. Similarly add credentials for 'mysqldbpw' and 'archerysec' with the config details mentioned in the yellow box of the below screenshots.





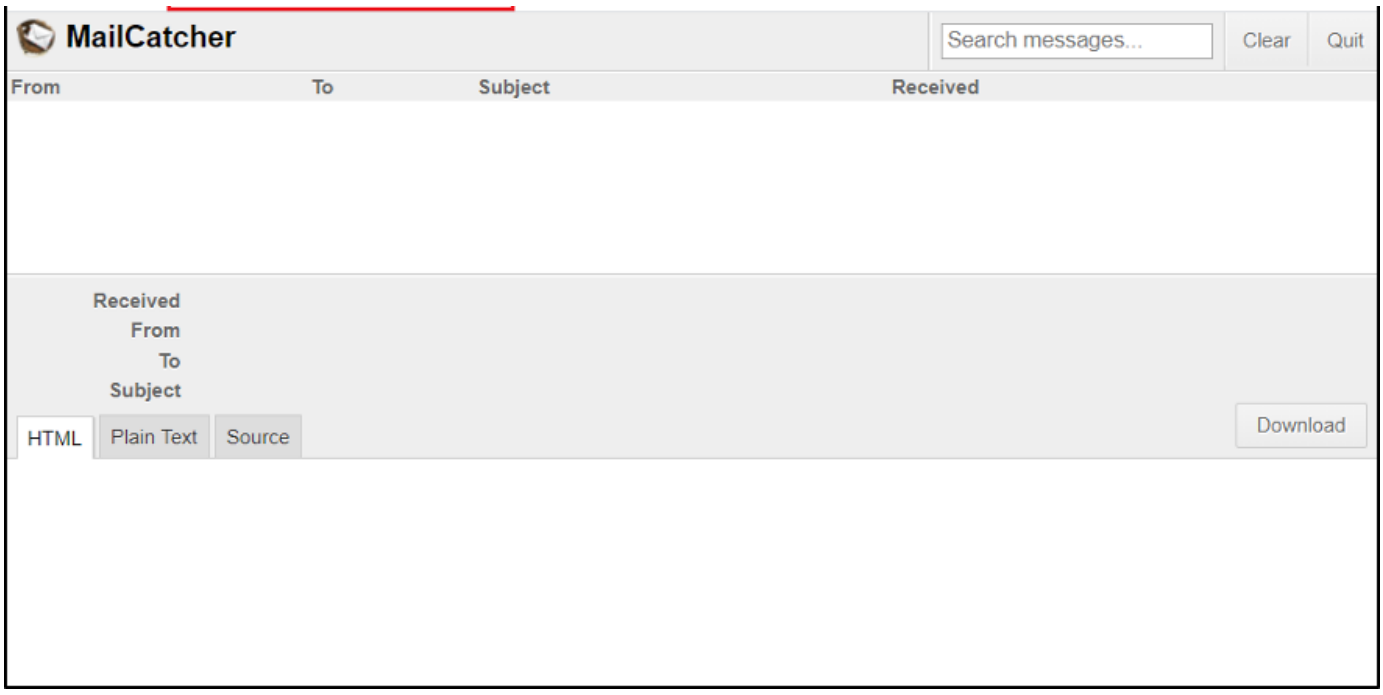
7. Three new entries will be seen as below.

Git SSH setup in Jenkins

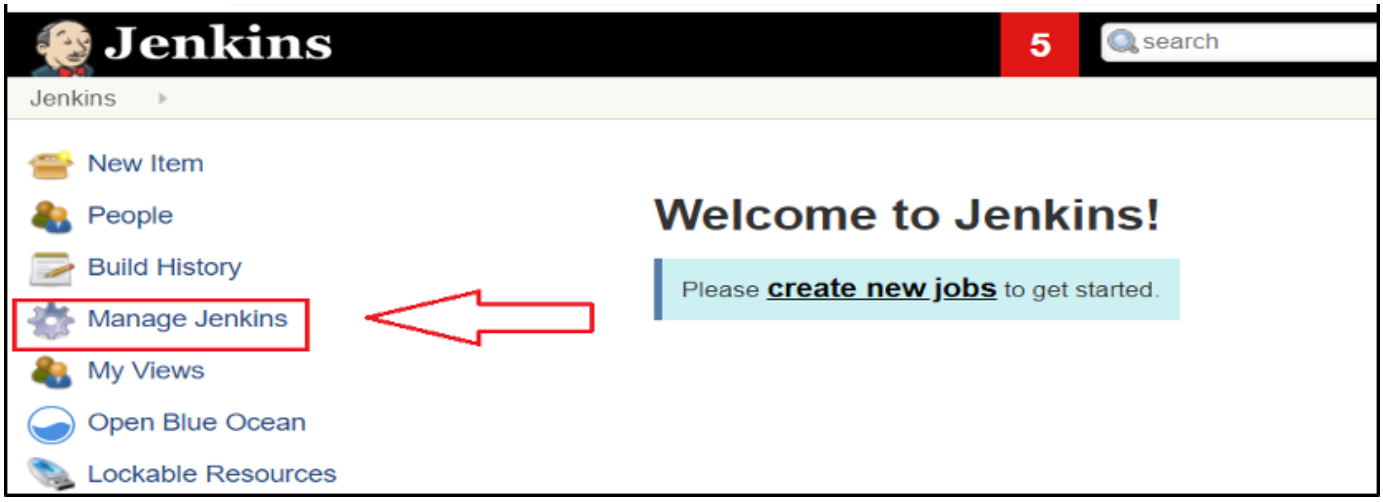
1. Add Credentials
2. Select SSH Username with private key
3. Put Username as your Github Username
4. Private Key Enter directly
5. Copy you private SSH key from SSH folder

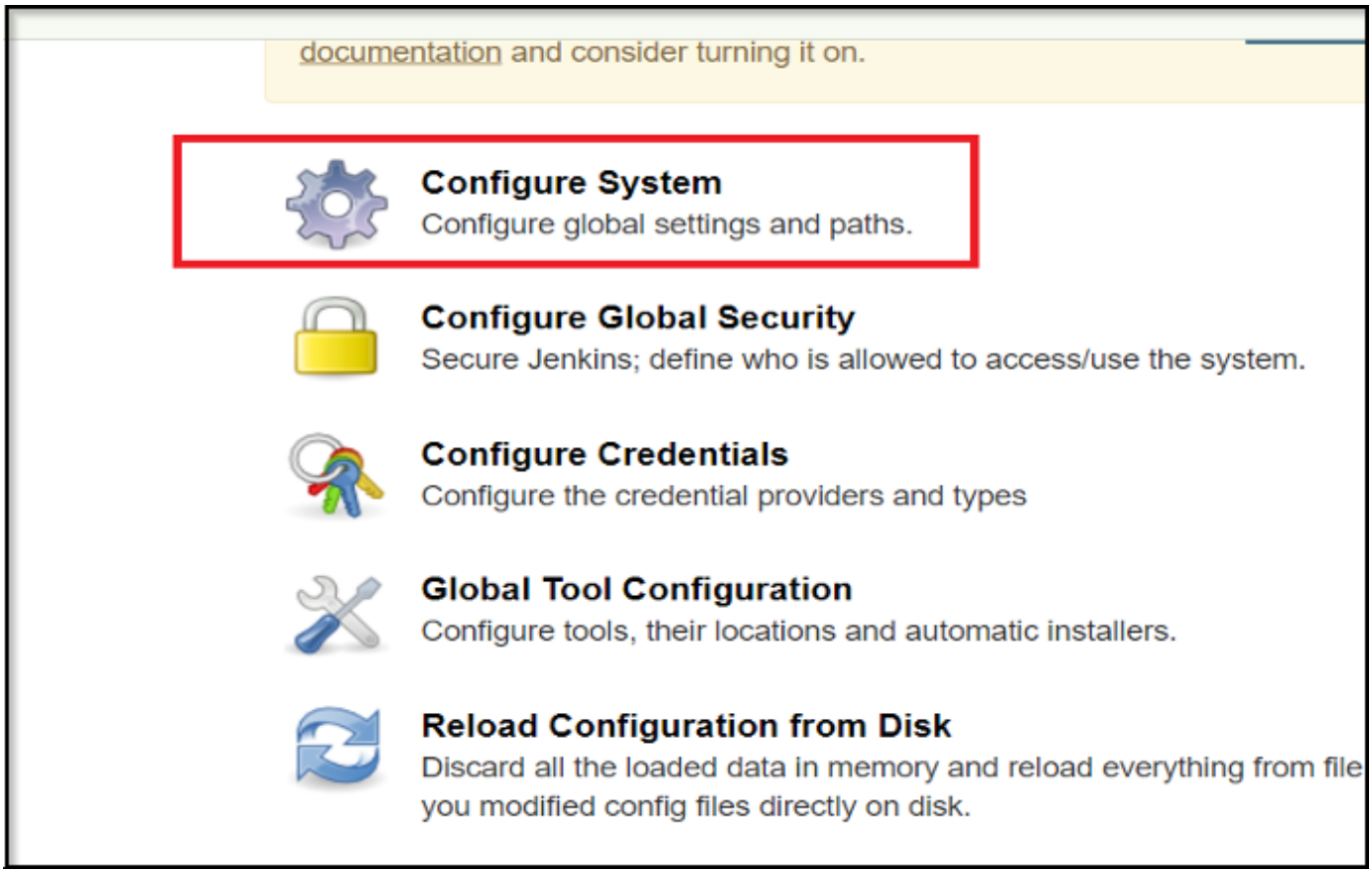
```
cat ~/.ssh/id_rsa
```

6. Open your browser and open MailCatcher Server.

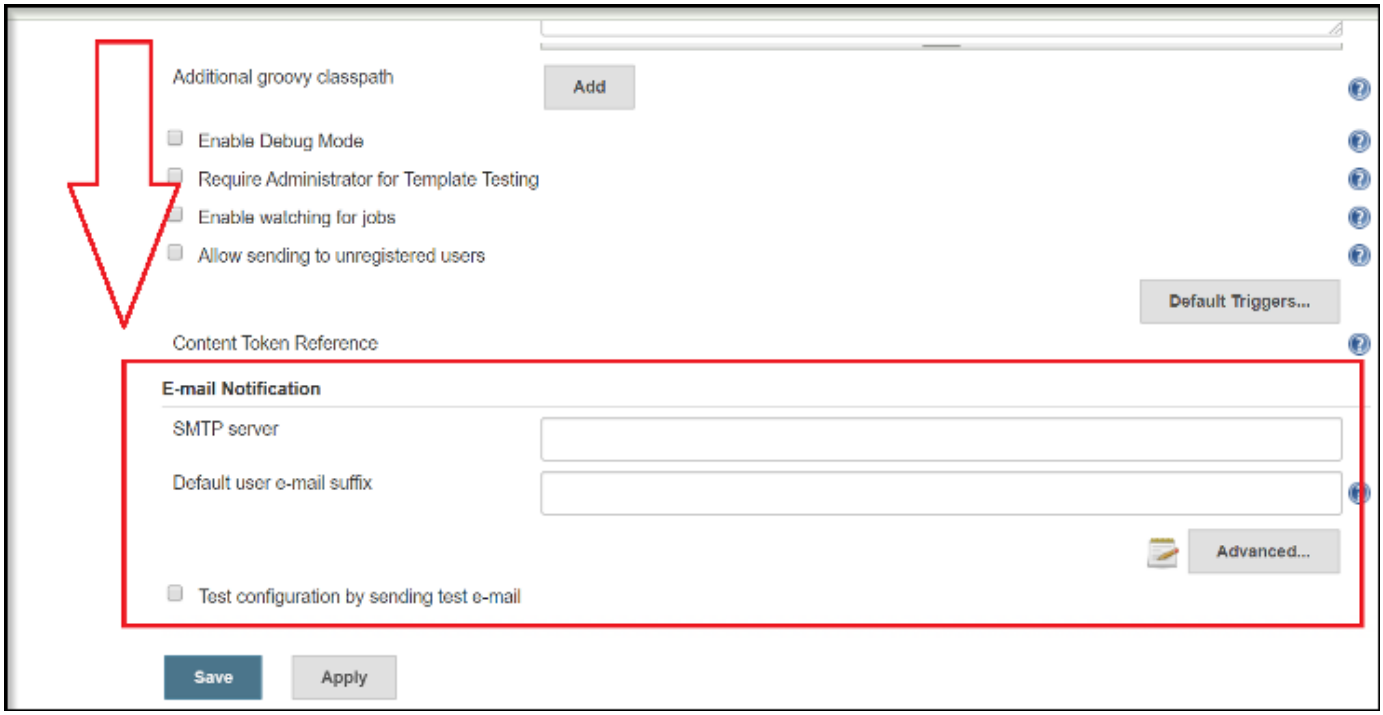


7. To configure the Mail, goto the Jenkins Dashboard and click “Manage Jenkins”, then scroll down to “Configure System”

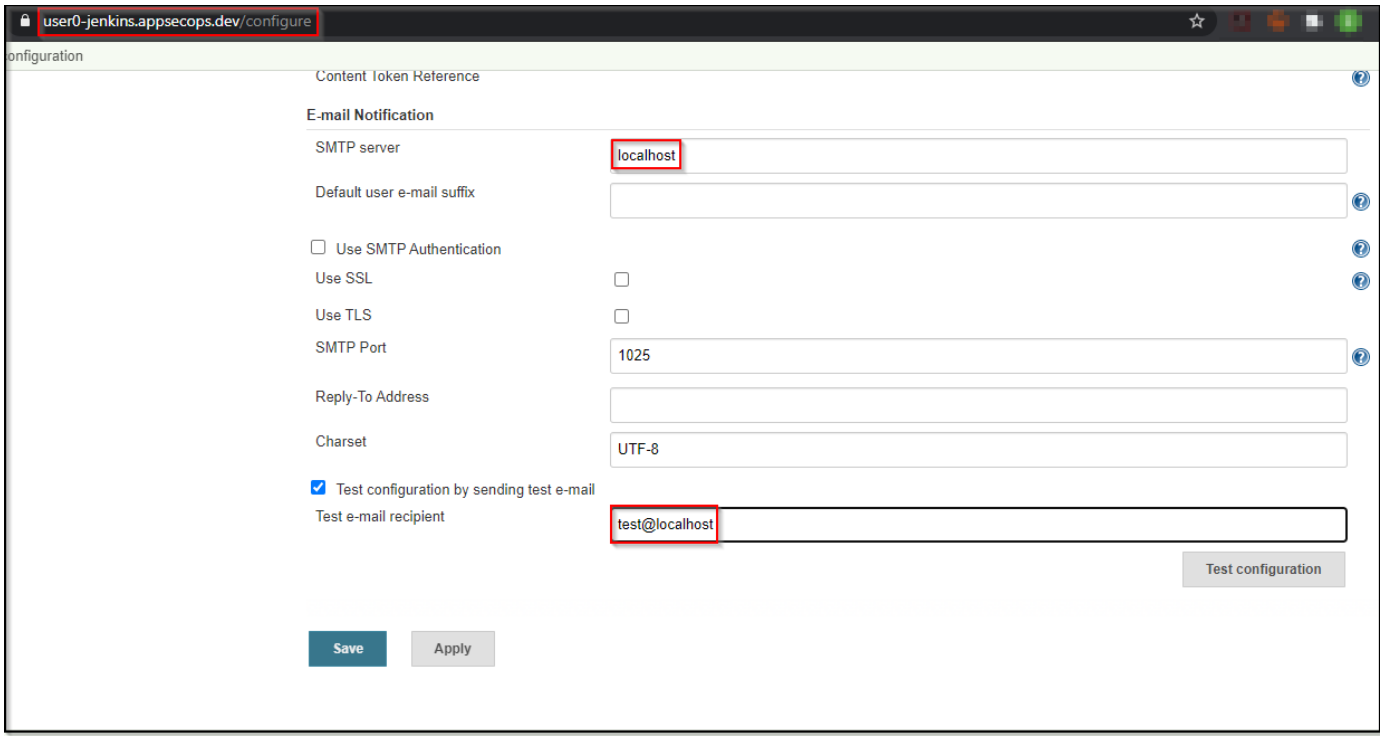


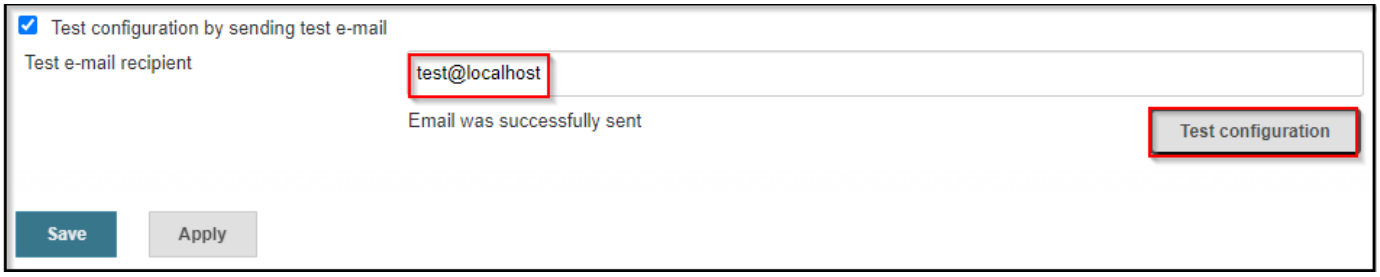


8. Scroll down to “E-mail Notification” section.

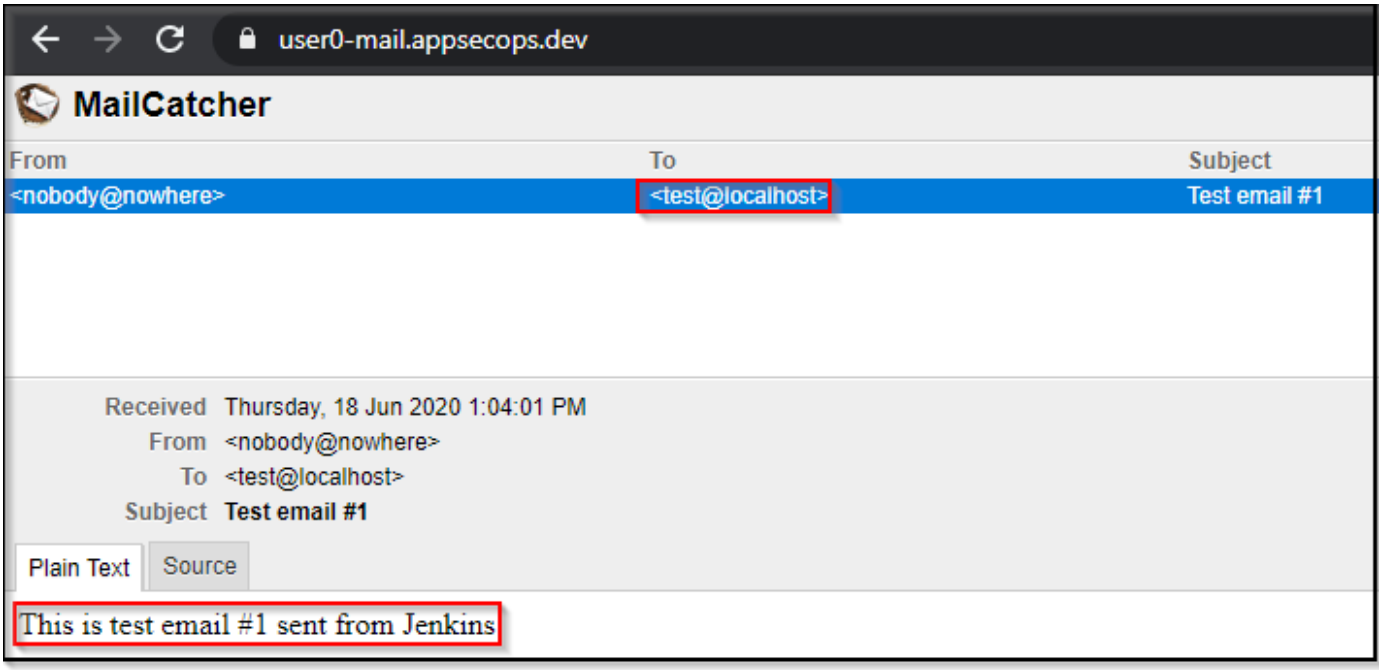


9. Click “Advanced” and enter the details accordingly as seen below. Click “Test configuration” to test the config





Go back to MailCatcher and you will see a test email sent from Jenkins.



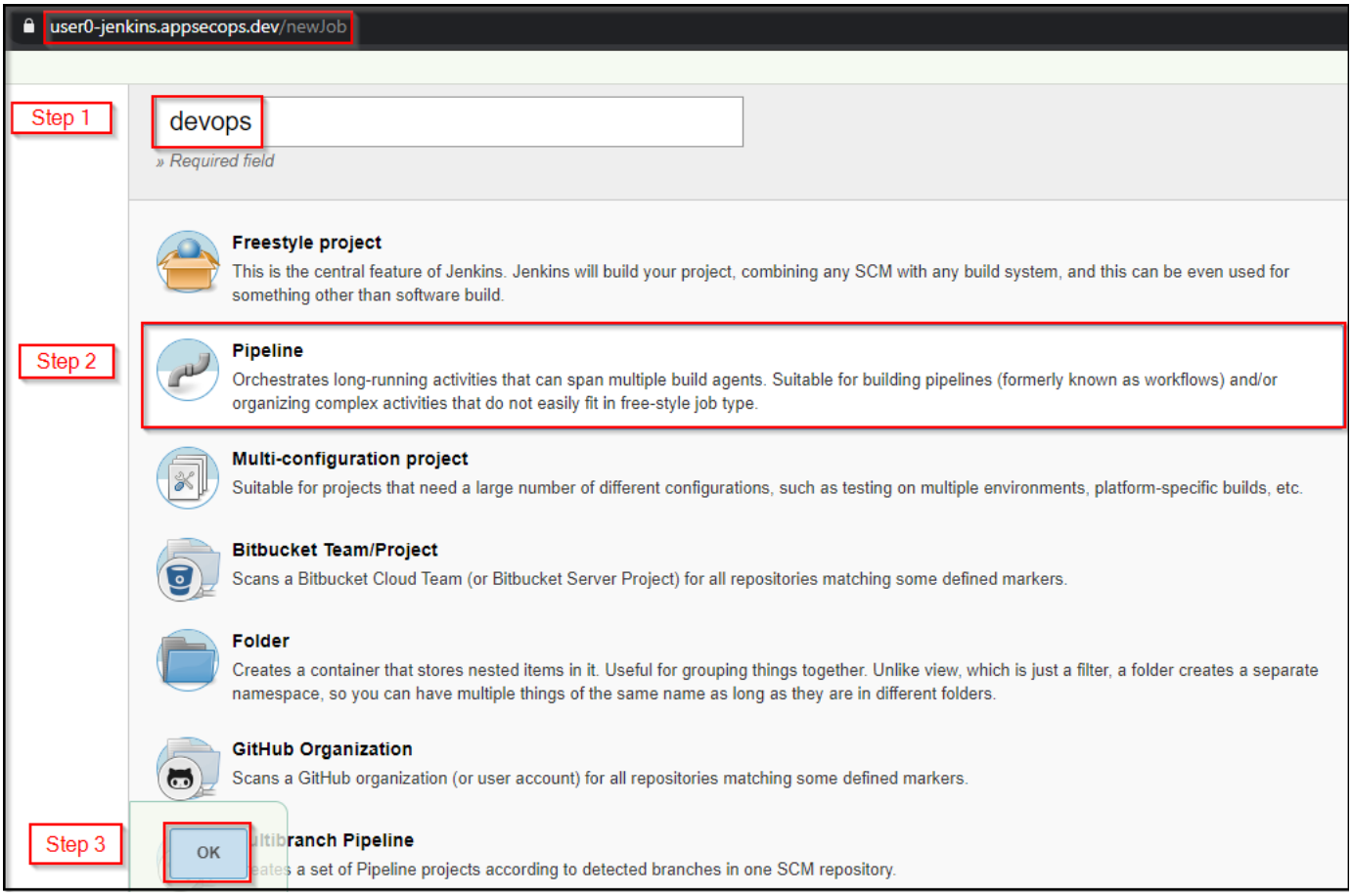
Instructions to create your first DevSecOps Pipeline

1. Open your browser and Login to Jenkins using the credentials

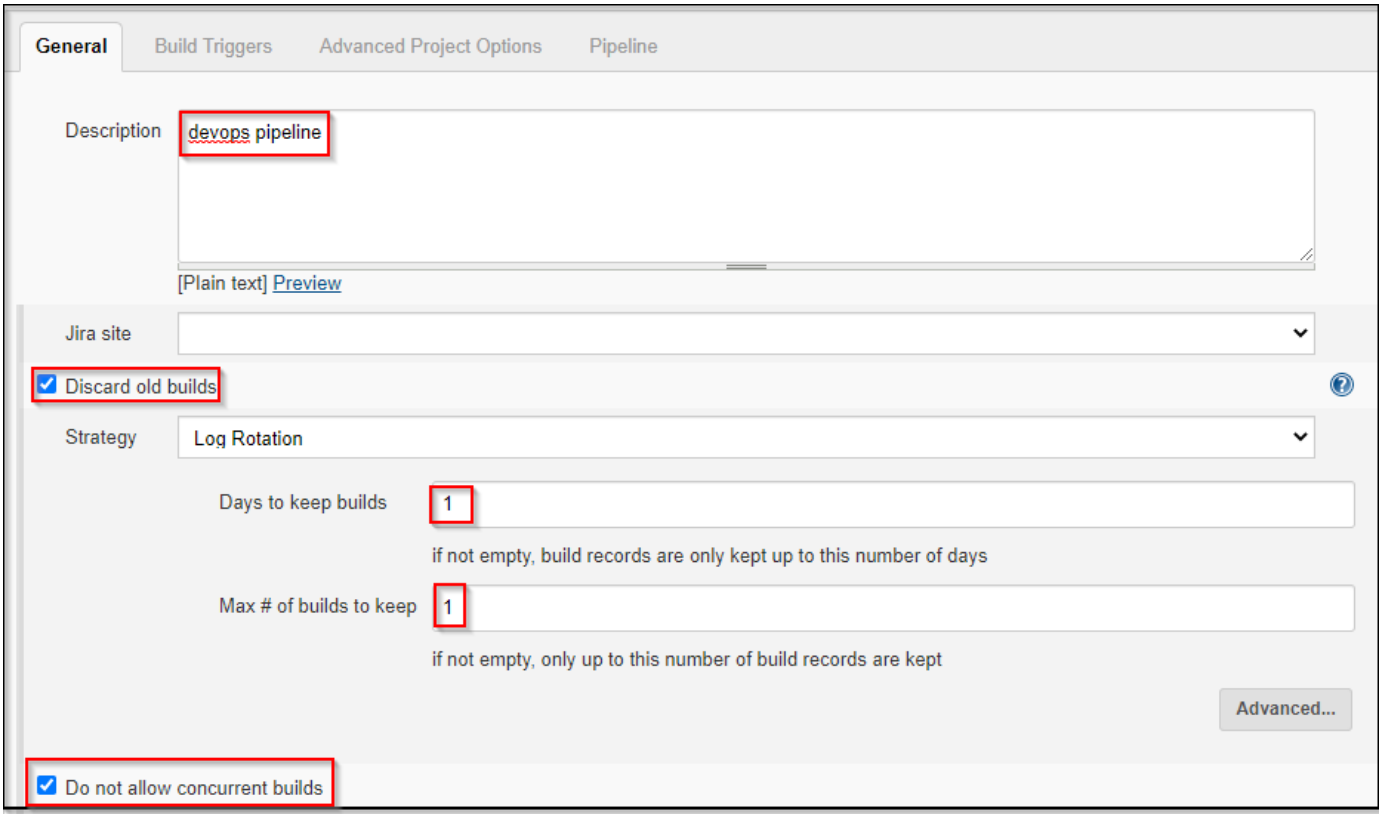
`jenkins:jenkins`

2. Click on “Create new jobs” to create a new job.

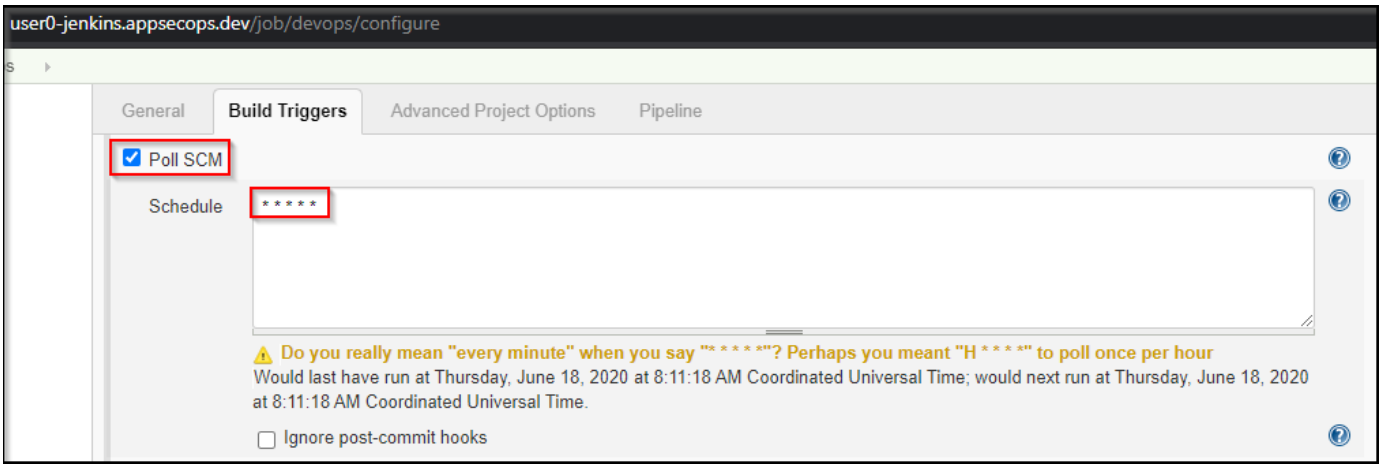
3. Next enter some name (eg. 'devops') in the "Enter an item name" section. Click "Pipeline" and then click "OK" at the bottom.



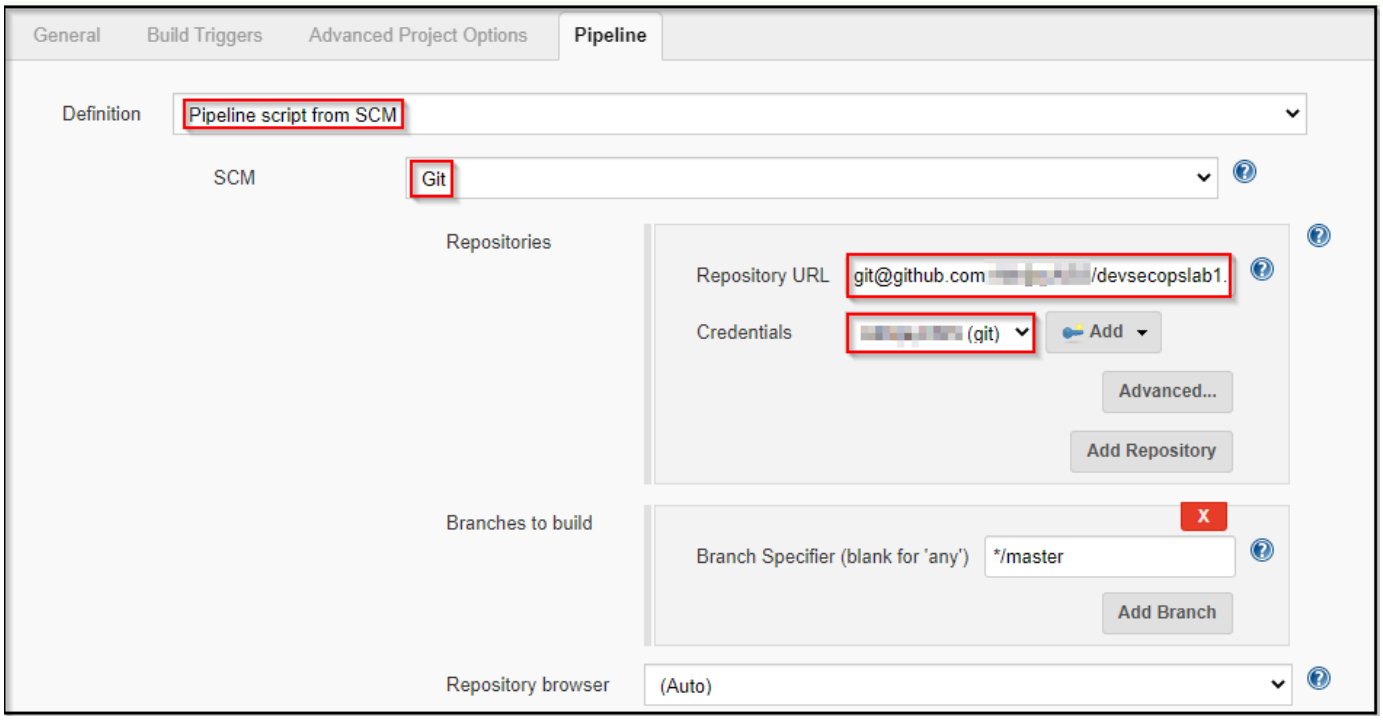
4. Enter the Configuration details as specified below.



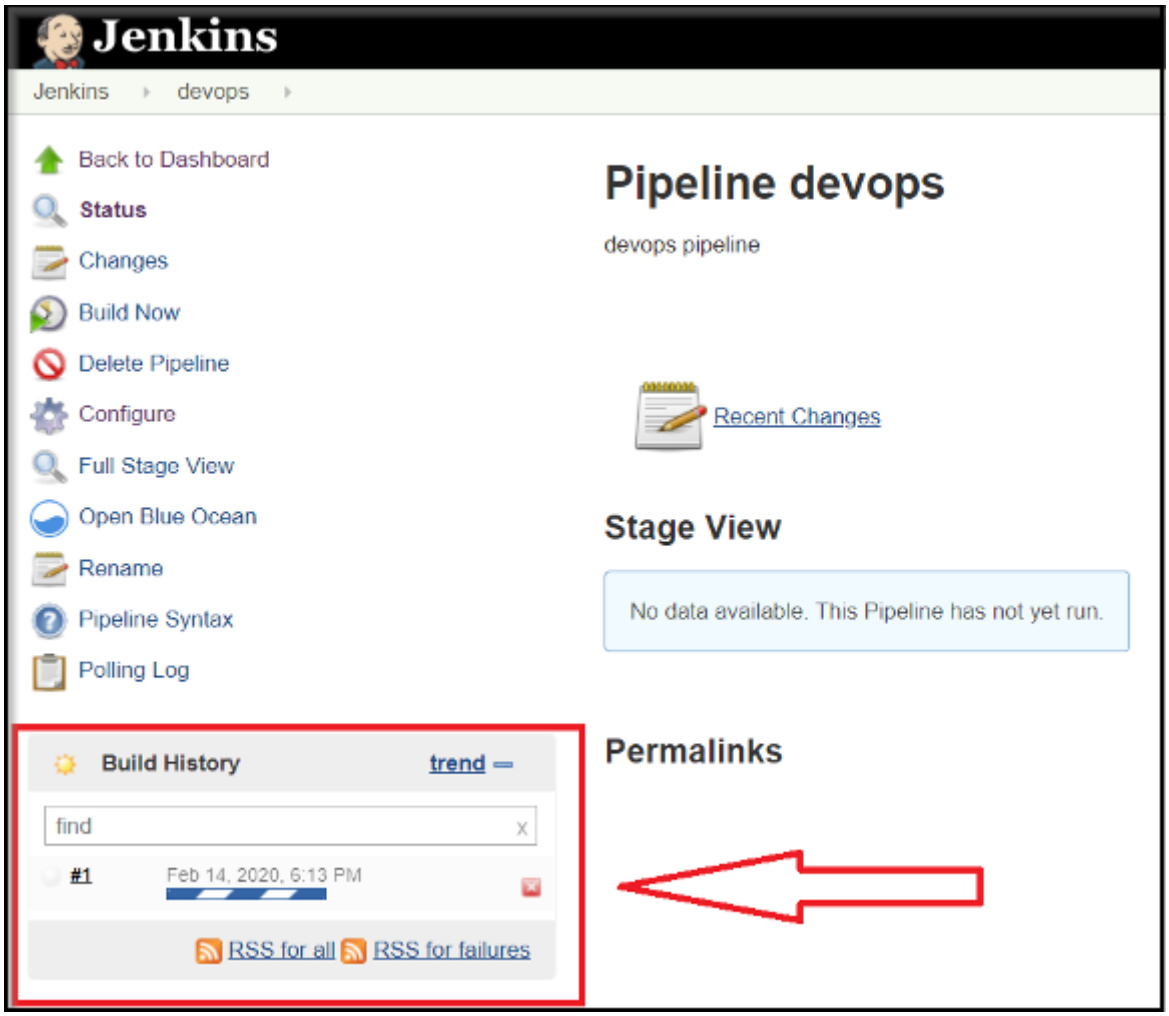
Poll SCM : * * * * *



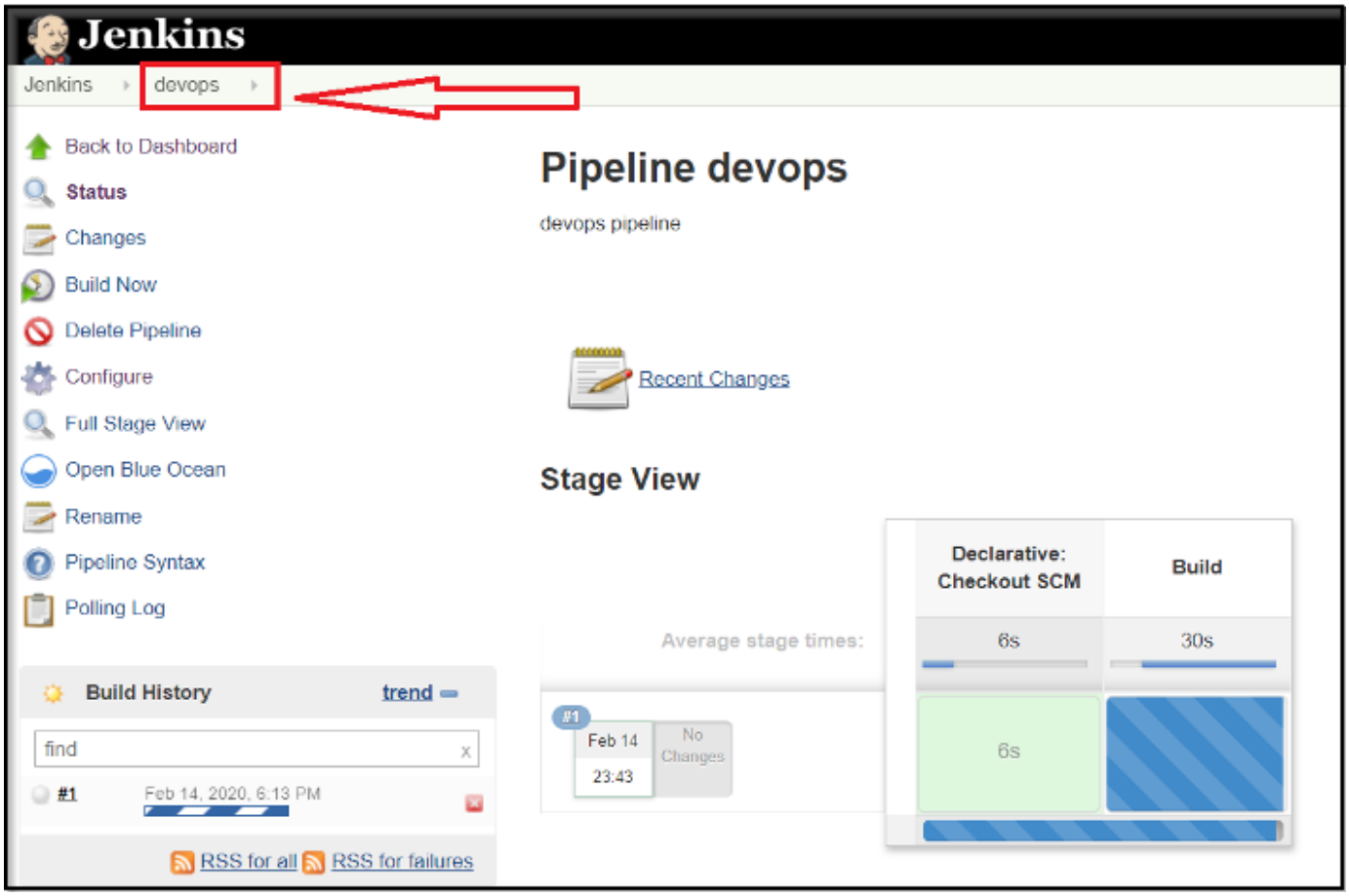
Copy your git Repo ssh URL and provide into Jenkins



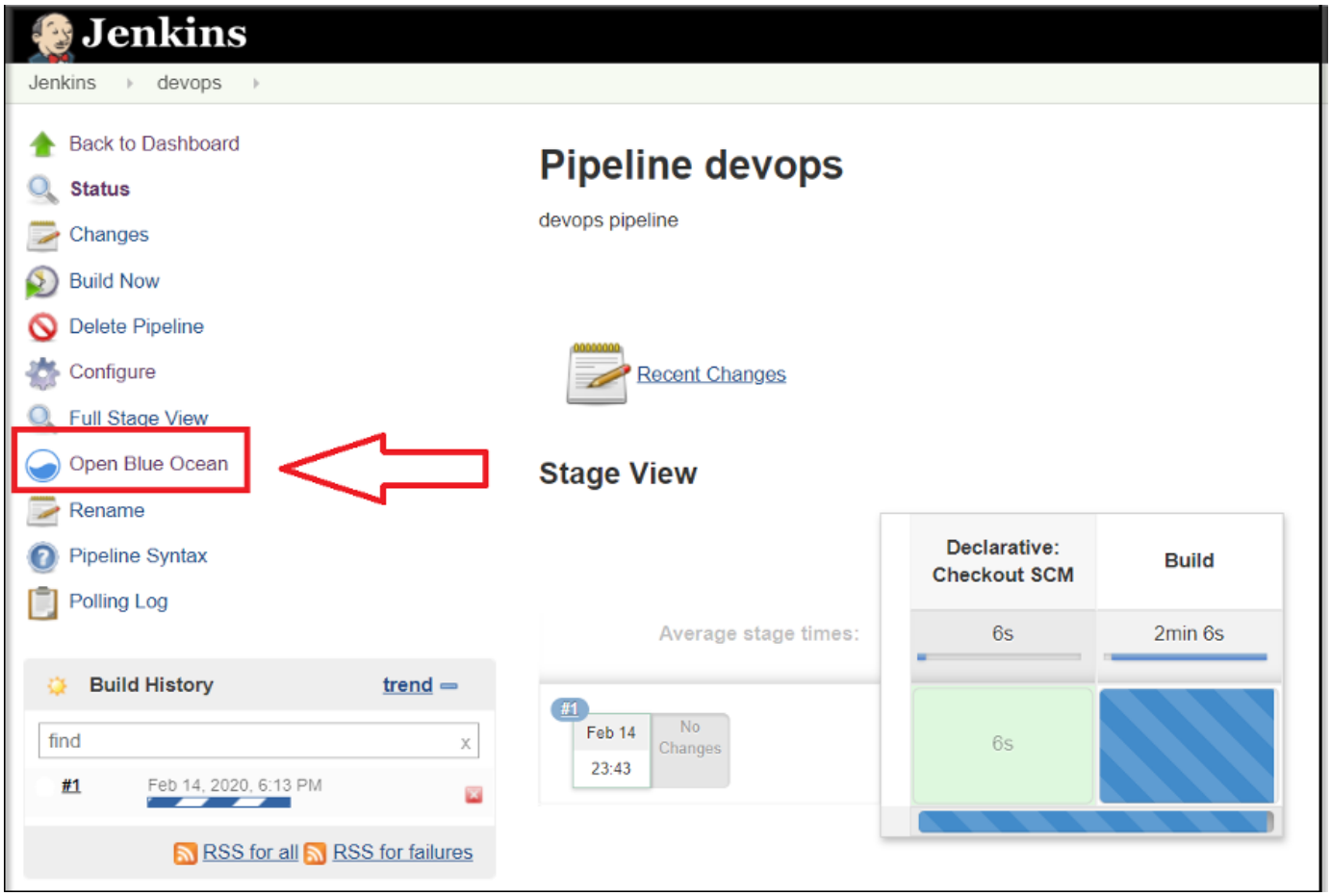
5. Once the configuration is saved, a build will run.

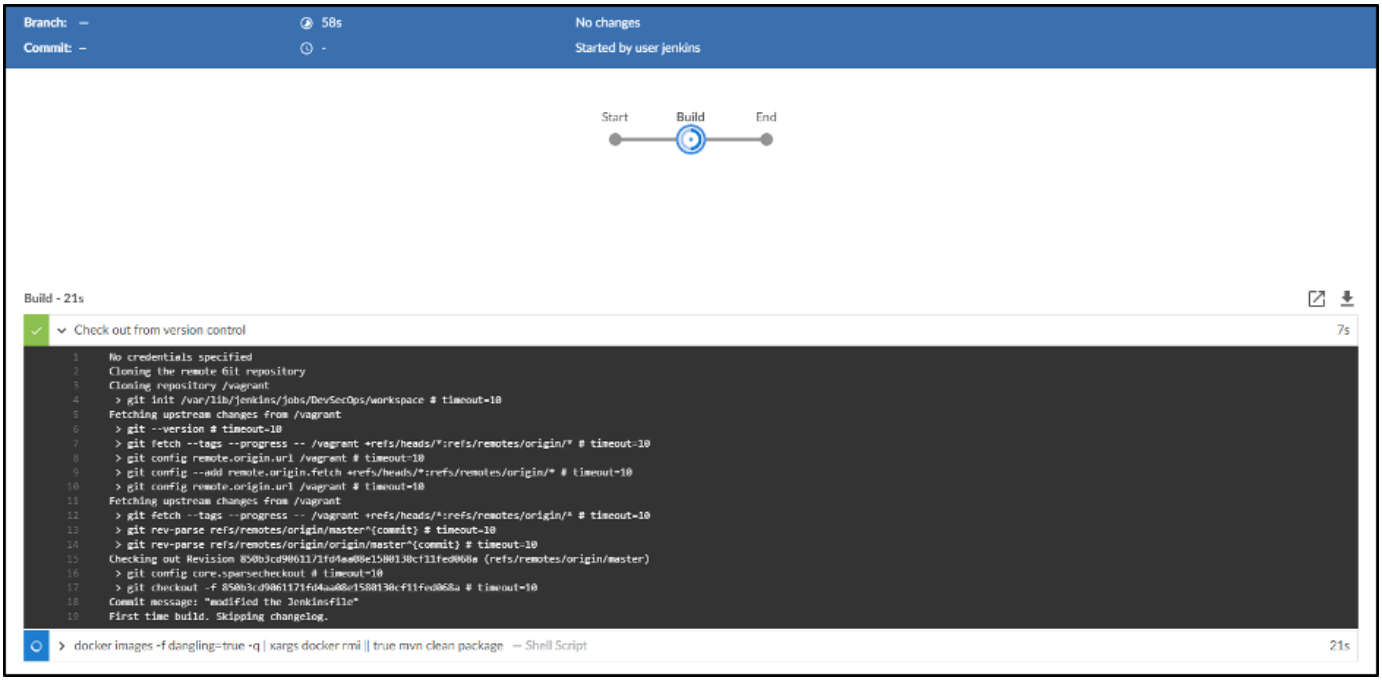


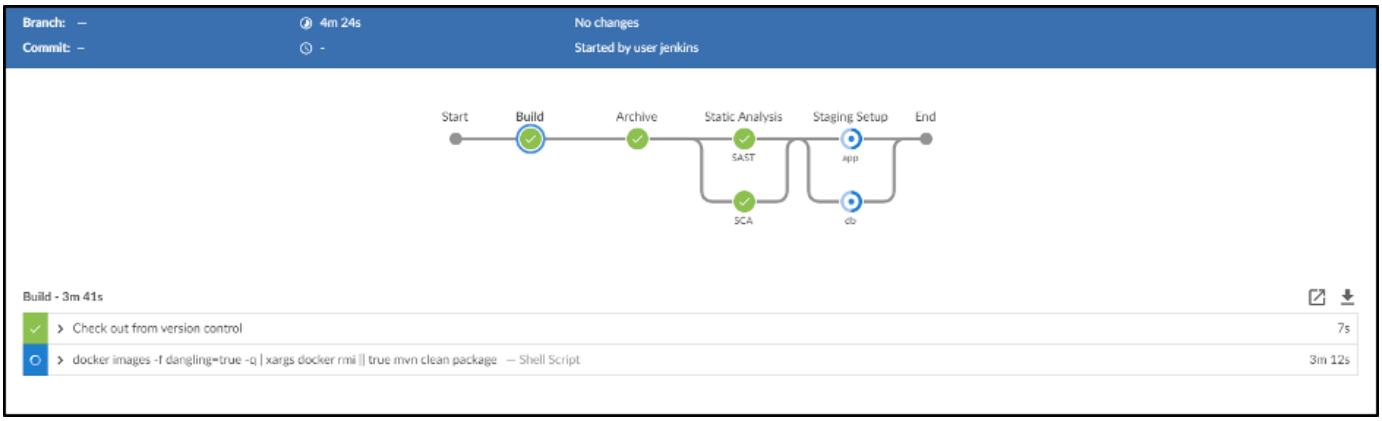
6. Click to open “Blue Ocean”.



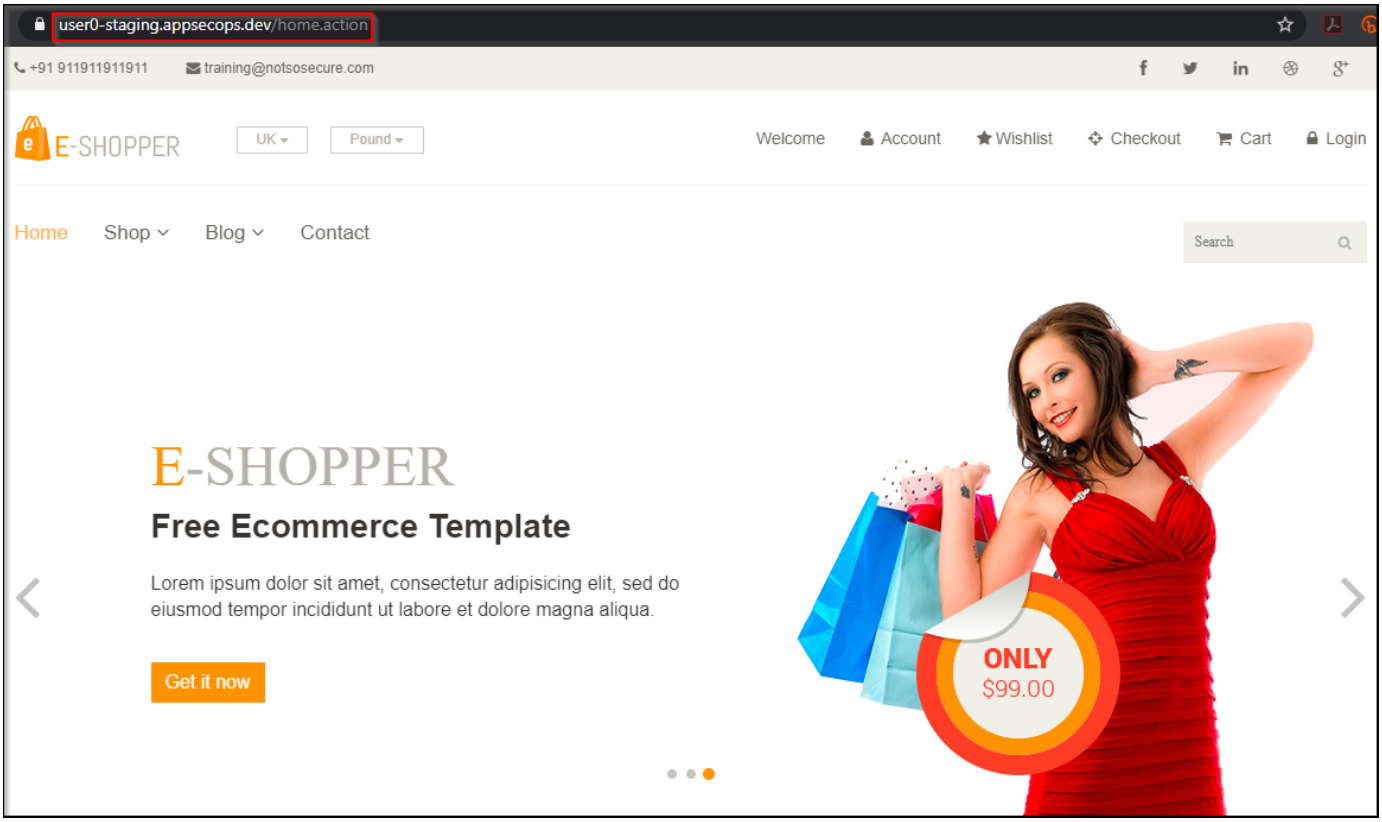
7. Below is the progress as represented by Blue Ocean plugin.





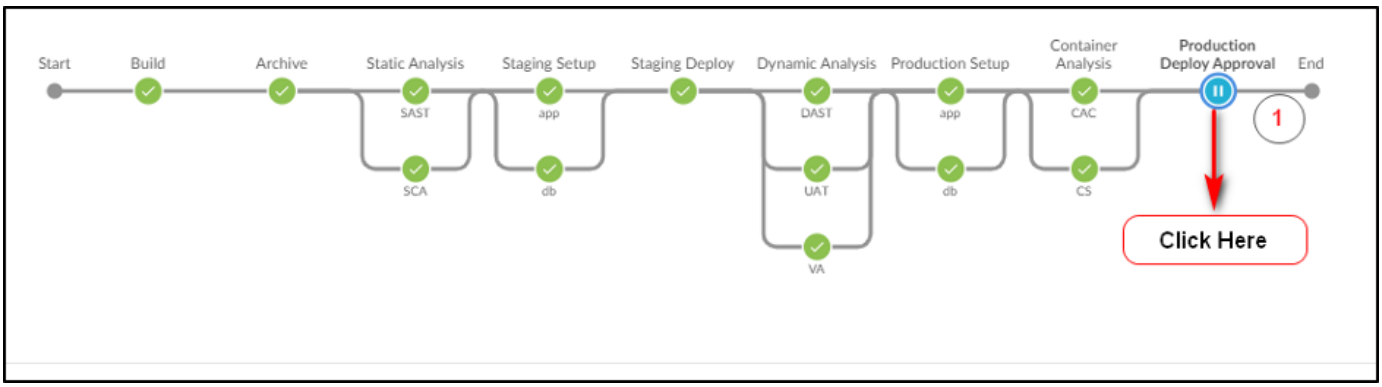


By the end of the 'Staging Deploy' stage your staging application should be up and running as shown below

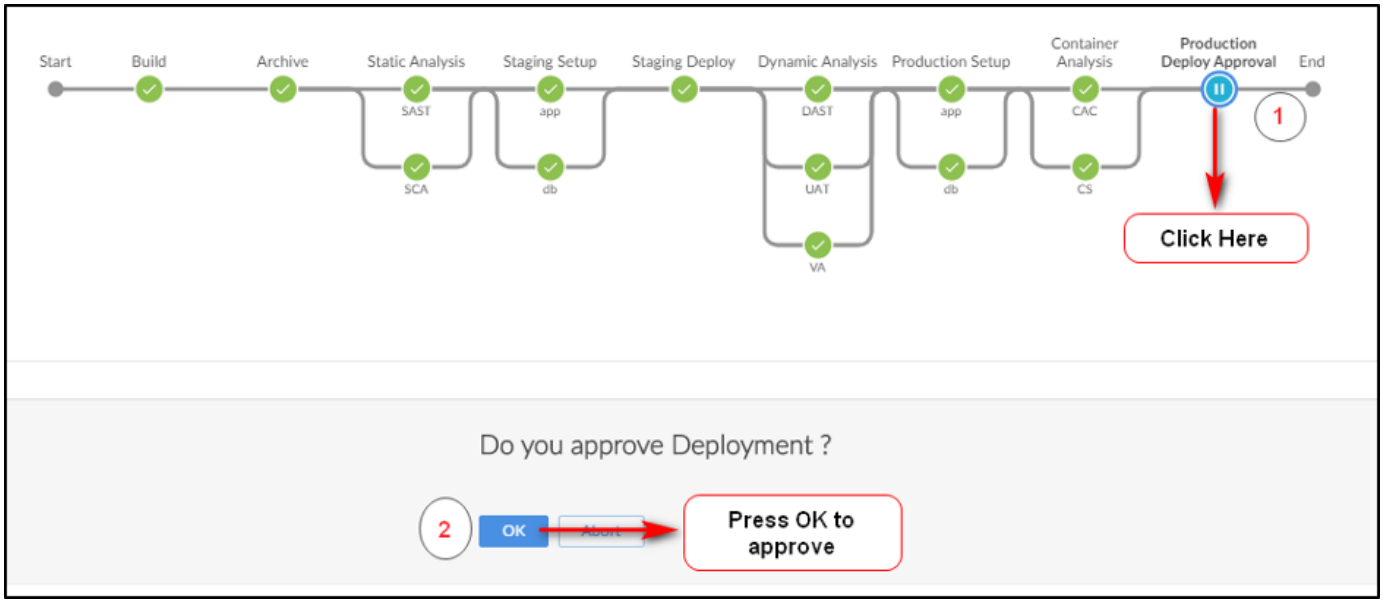


Click On the 'Production Deploy Approval'

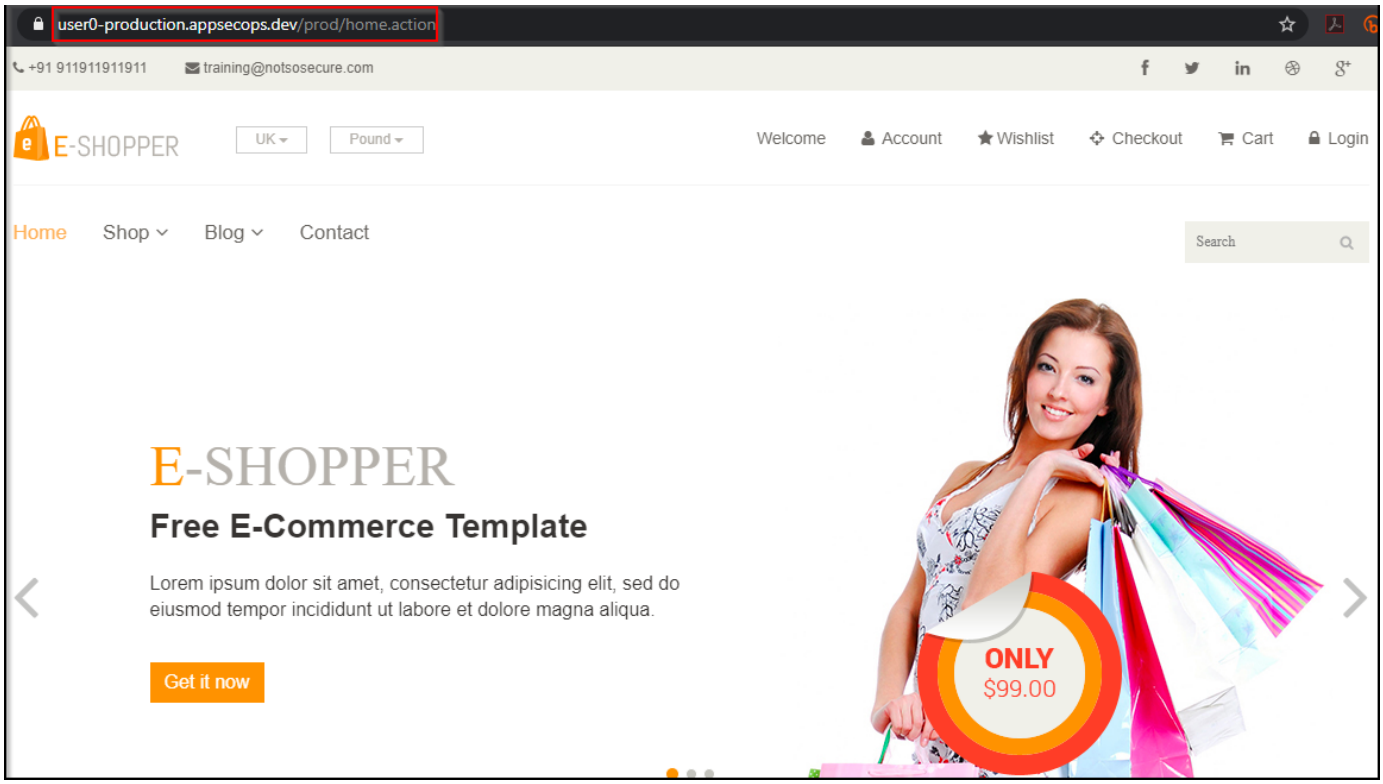
Click OK to deploy



After approving the deployment our pipeline would be complete as shown below



Once the pipeline is complete please access the production application deployed as shown below



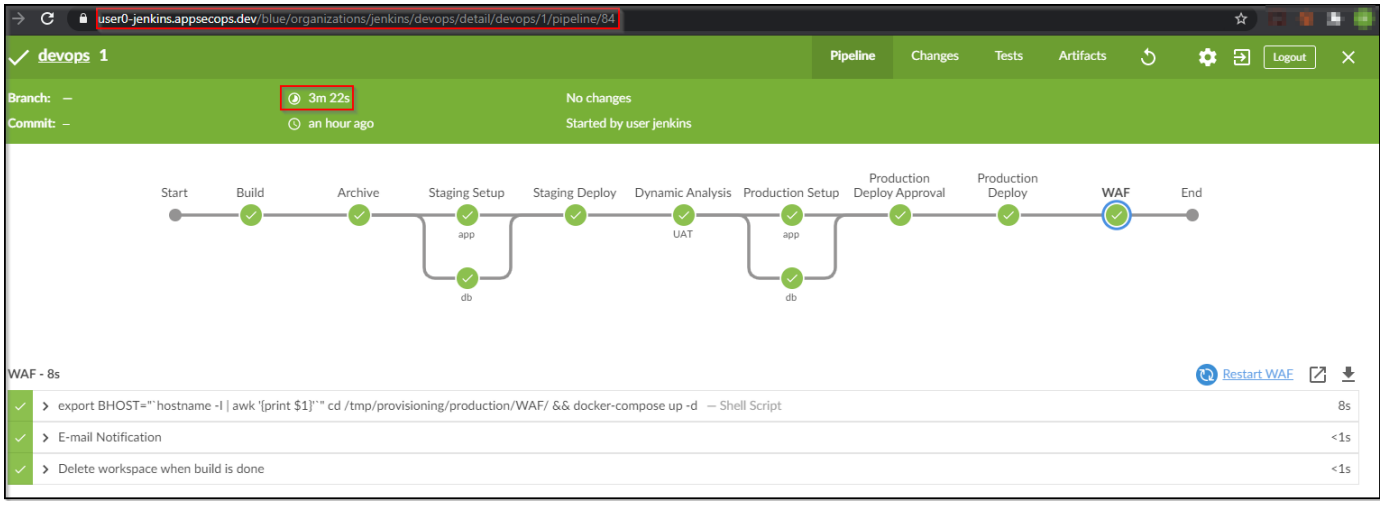
DevOps Pipline

1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in Jenkinsfile.DevOps
2. Copy Contents of `Jenkinsfile.DevOps`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

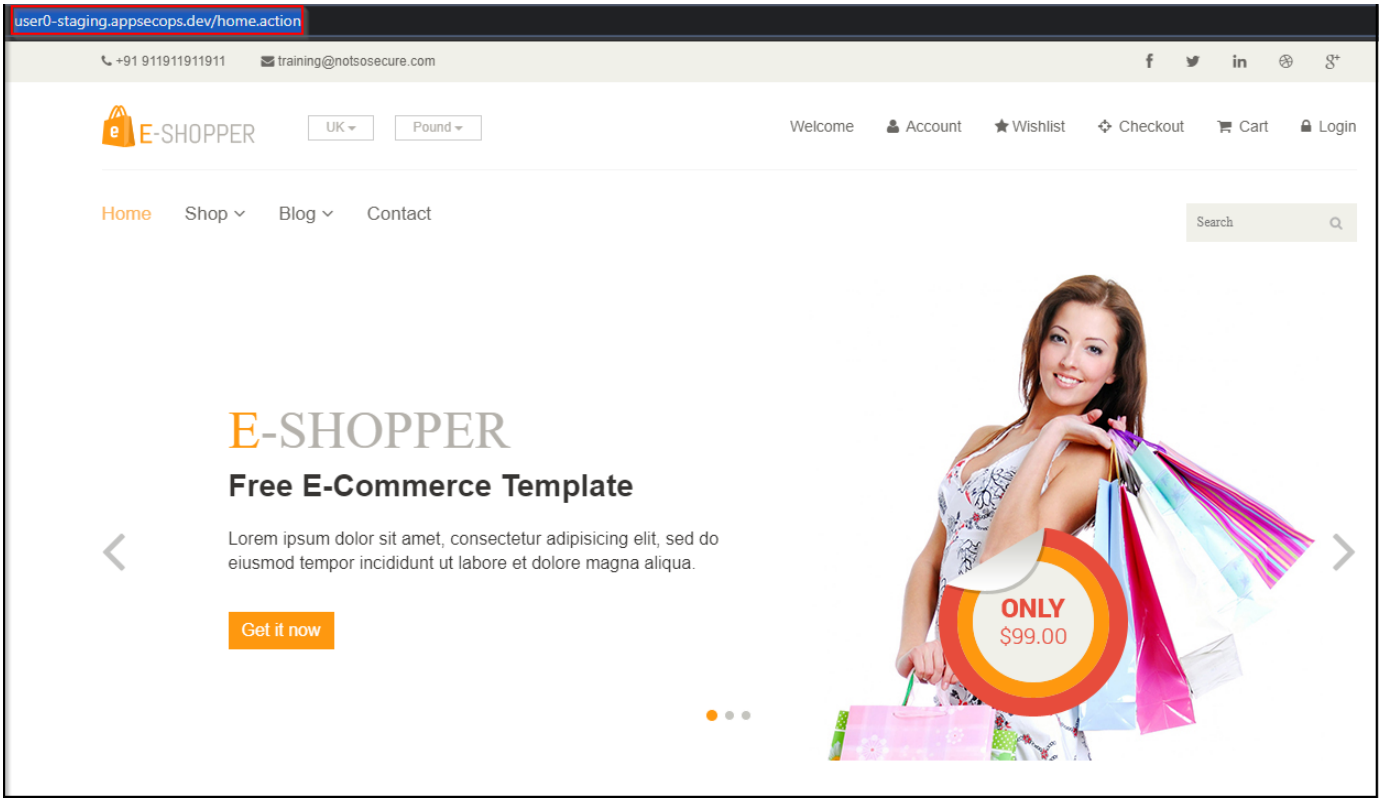
```
git commit -am "devops pipeline"
```

Pipeline



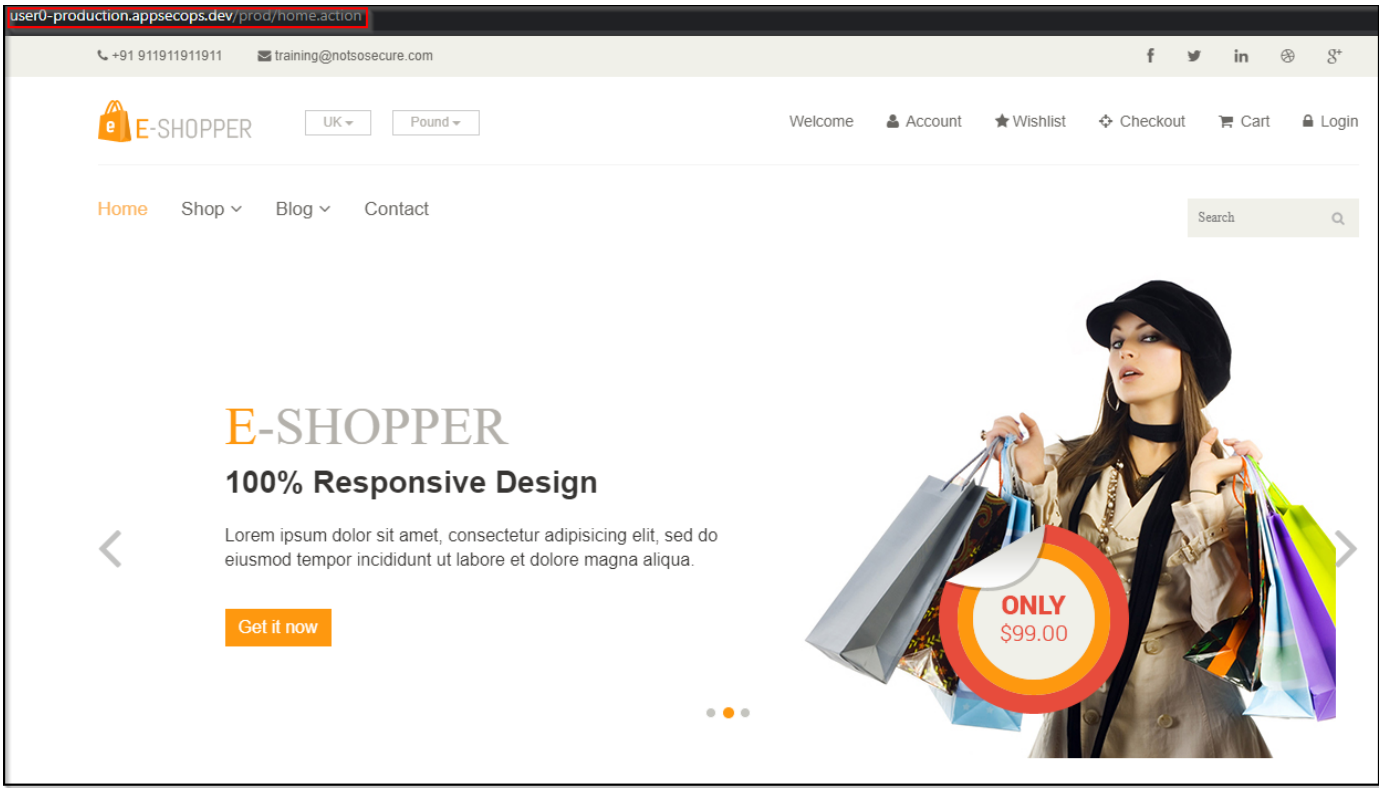
Staging App

https://user0-staging.appsecops.dev/home.action



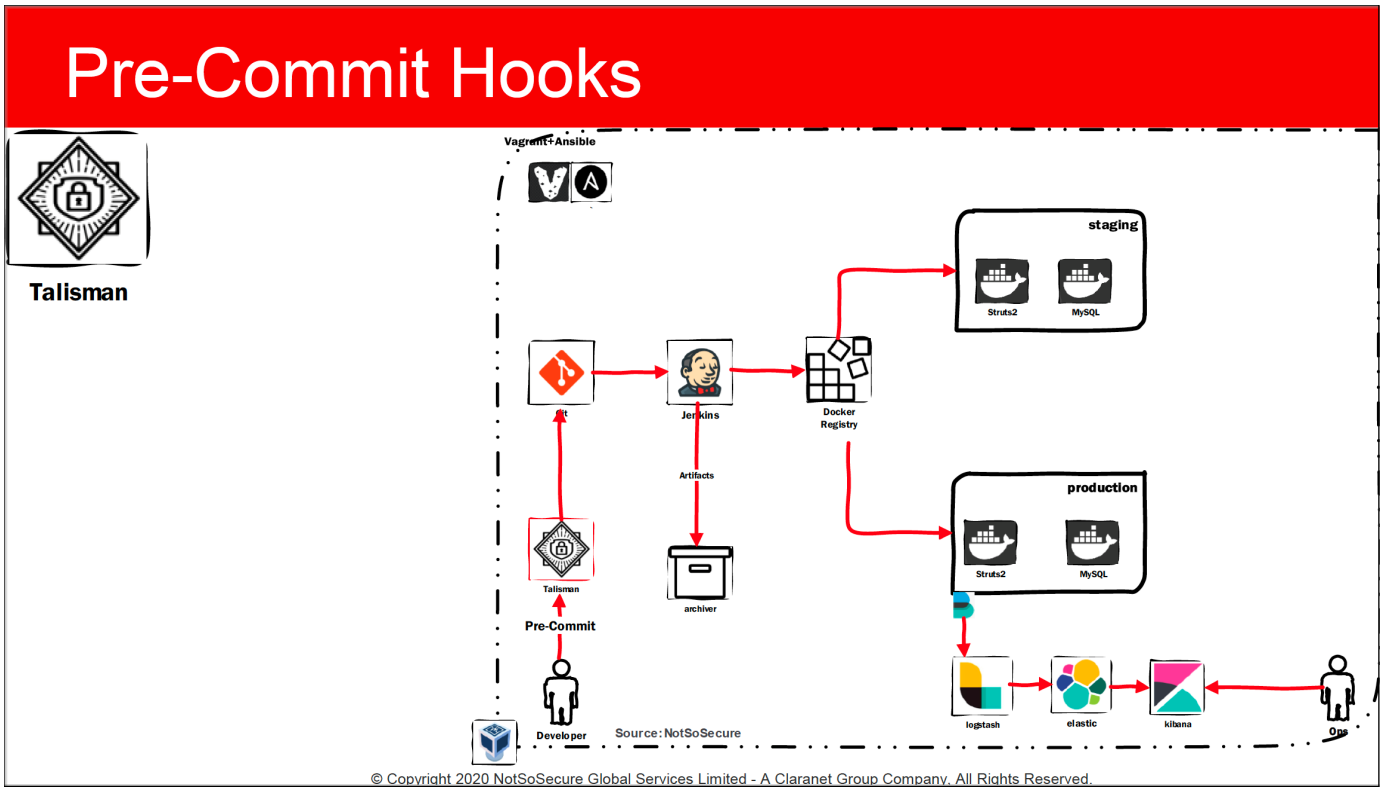
Production App

https://user0-production.appsecops.dev/prod/home.action

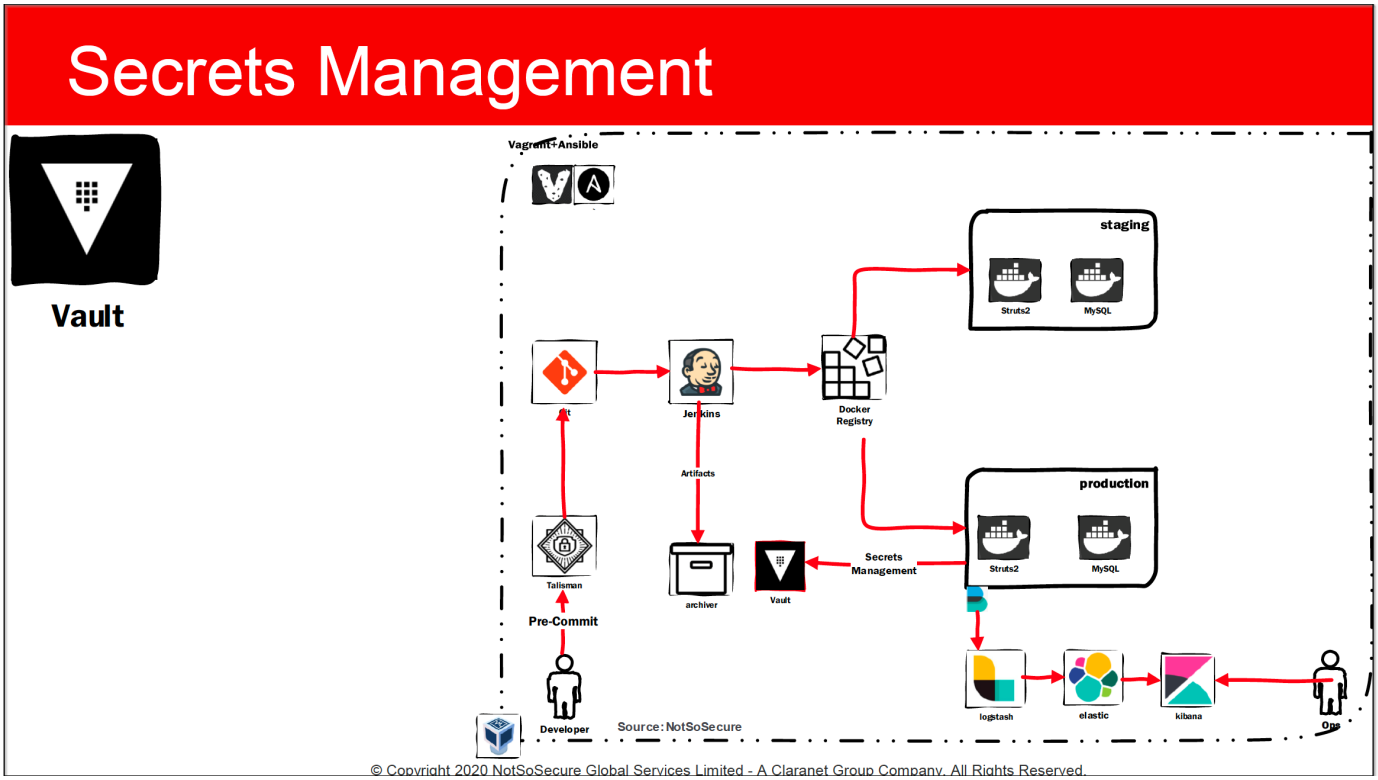


Continuous Integration

Pre-Commit Hooks



Secrets Management


Secrets Management
Vault
Vagrant+Ansible
staging
Struts2
MySQL
Jenkins
Docker Registry
Artifacts
production
Struts2
MySQL
Tailsman
archiver
Vault
Secrets Management
Pre-Commit
Developer
Source: NotSoSecure
logstash
elastic
kibana
Ops
© Copyright 2020 NotSoSecure Global Services Limited - A Claranet Group Company. All Rights Reserved.

Pre-commit Hook

Install Talisman

```
curl
https://raw.githubusercontent.com/thoughtworks/talisman/master/global_install_scr
 > ~/install-talisman.sh
```

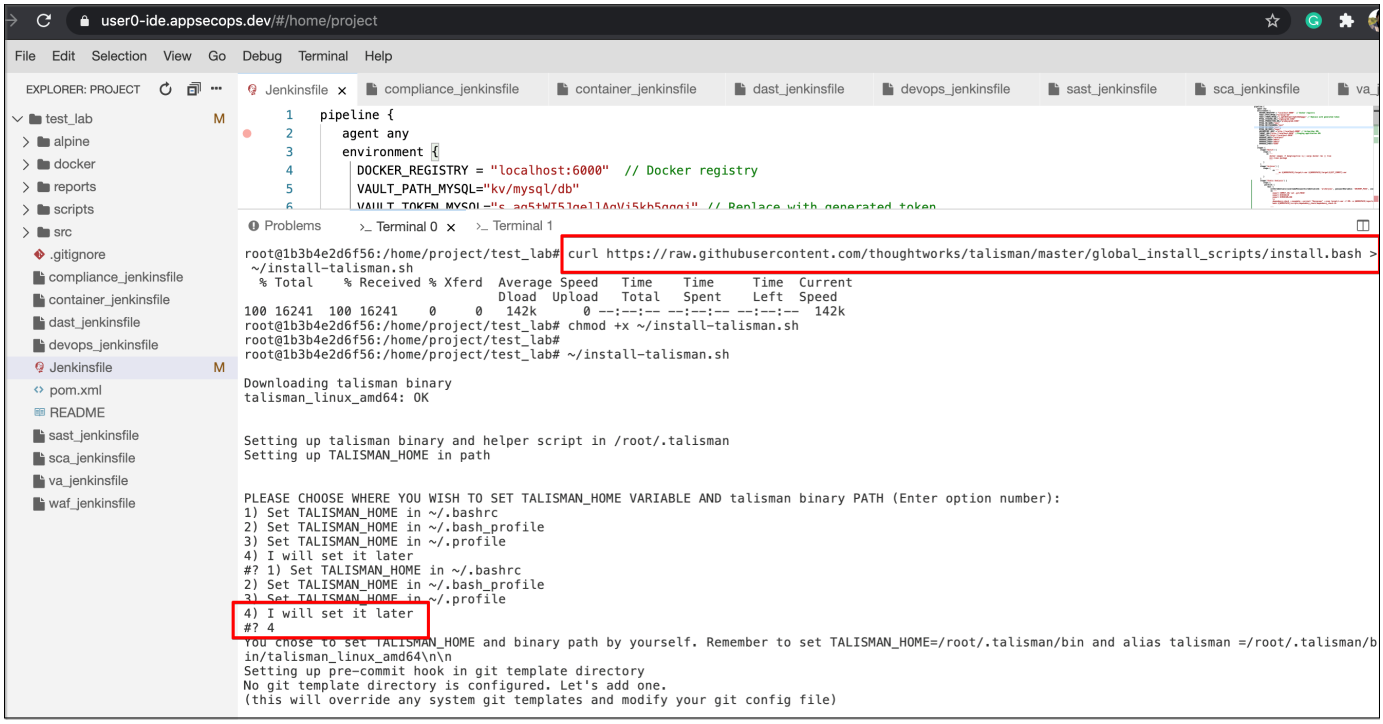
```
chmod +x ~/install-talisman.sh
```

```
~/install-talisman.sh
```

```
PLEASE CHOOSE WHERE YOU WISH TO SET TALISMAN_HOME VARIABLE AND talisman binary
PATH (Enter option number):
1) Set TALISMAN_HOME in ~/.bashrc
2) Set TALISMAN_HOME in ~/.bash_profile
3) Set TALISMAN_HOME in ~/.profile
4) I will set it later
```

```
Please enter root directory to search for git repos (Default: /root):
/home/project/
```

Don't forget to give your repo path

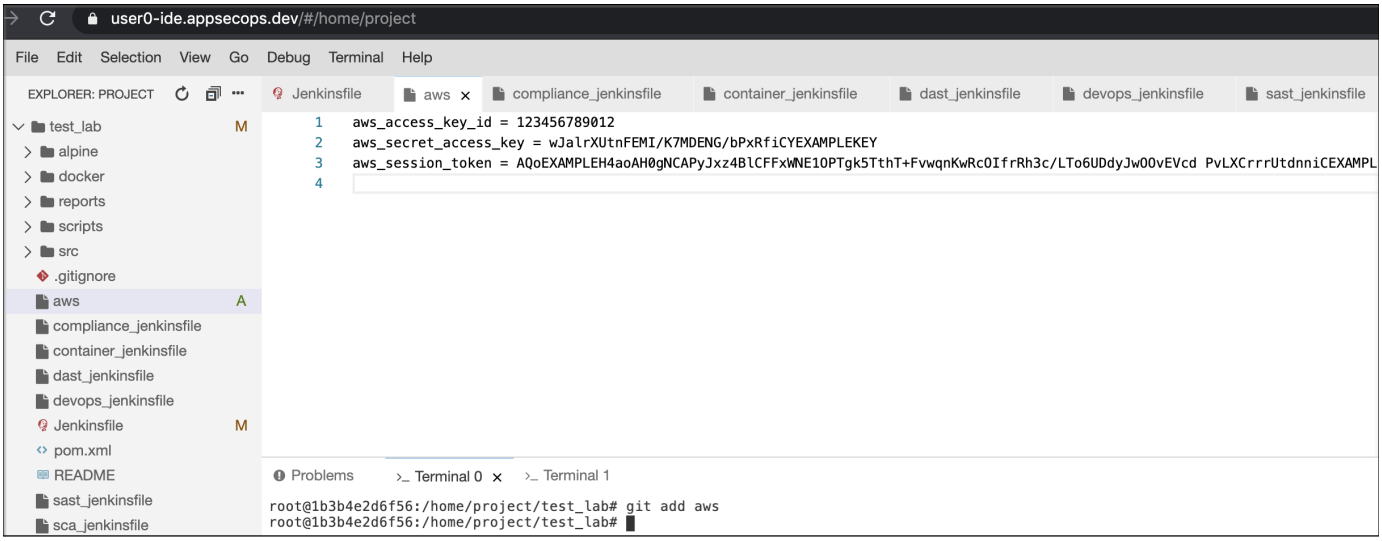


```
Problems    >_ Terminal 0 ×    >_ Terminal 1


Setting up talisman binary and helper script in /root/.talisman
Setting up TALISMAN_HOME in path


PLEASE CHOOSE WHERE YOU WISH TO SET TALISMAN_HOME VARIABLE AND talisman binary PATH (Enter option number):
1) Set TALISMAN_HOME in ~/.bashrc
2) Set TALISMAN_HOME in ~/.bash_profile
3) Set TALISMAN_HOME in ~/.profile
4) I will set it later
#? 1) Set TALISMAN_HOME in ~/.bashrc
2) Set TALISMAN_HOME in ~/.bash_profile
3) Set TALISMAN_HOME in ~/.profile
4) I will set it later
#? 4
You chose to set TALISMAN_HOME and binary path by yourself. Remember to set TALISMAN_HOME=/root/.talisman/bin and alias talisman =/root/.talisman/b
in/talisman_linux_amd64\n\n
Setting up pre-commit hook in git template directory
No git template directory is configured. Let's add one.
(this will override any system git templates and modify your git config file)

Git template directory: (/root/.git-template)

Setting up template pre-commit hook
'/root/.git-template/hooks/pre-commit' -> '/root/.talisman/bin/talisman_hook_script'
Talisman template hook successfully installed.

Setting up talisman hook recursively in git repos
Please enter root directory to search for git repos (Default: /root): /home/project/
        Searching /home/project/ for git repositories
Setting up pre-commit hook in /home/project/test_lab/.git/hooks
'/home/project/test_lab/.git/hooks/pre-commit' -> '/root/.talisman/bin/talisman_hook_script'
DONE
```

Create config file with below data

```
aws_access_key_id = 123456789012
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
aws_session_token =
AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEV

PvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4Olgk
```

```
git add aws
```




```
git commit -m "accidentally adding aws keys"
```

```
Problems    >_ Terminal 0 ×    >_ Terminal 1

root@1b3b4e2d6f56:/home/project/test_lab# git commit -m "accidentally adding aws keys"

Talisman Report:
+------+-------------------------------------------------------+
| FILE |                        ERRORS                         |
+------+-------------------------------------------------------+
| aws  | Expected file to not to contain                       |
|      | base64 encoded texts such as:                         |
|      | wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY              |
+------+-------------------------------------------------------+
| aws  | Expected file to not to contain                       |
|      | base64 encoded texts such as:                         |
|      | AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT...    |
+------+-------------------------------------------------------+
| aws  | Expected file to not to contain                       |
|      | base64 encoded texts such as:                         |
|      | PvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgR...    |
+------+-------------------------------------------------------+
| aws  | Potential secret pattern                              |
|      | : aws_access_key_id =                                 |
|      |                                          123456789012 |
+------+-------------------------------------------------------+
| aws  | Potential secret pattern                              |
|      | : aws_secret_access_key =                             |
|      | wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY              |
+------+-------------------------------------------------------+
| aws  | Potential secret pattern :                            |
|      | aws_access_key_id =                                   |
+------+-------------------------------------------------------+
| aws  | Potential secret pattern :                            |
|      | aws_secret_access_key =                               |
+------+-------------------------------------------------------+
```

To delete the aws credetnail file created use the below command

```
rm -rf aws
```

Navigate to .git folder and type the below commmand to remove talisman hook

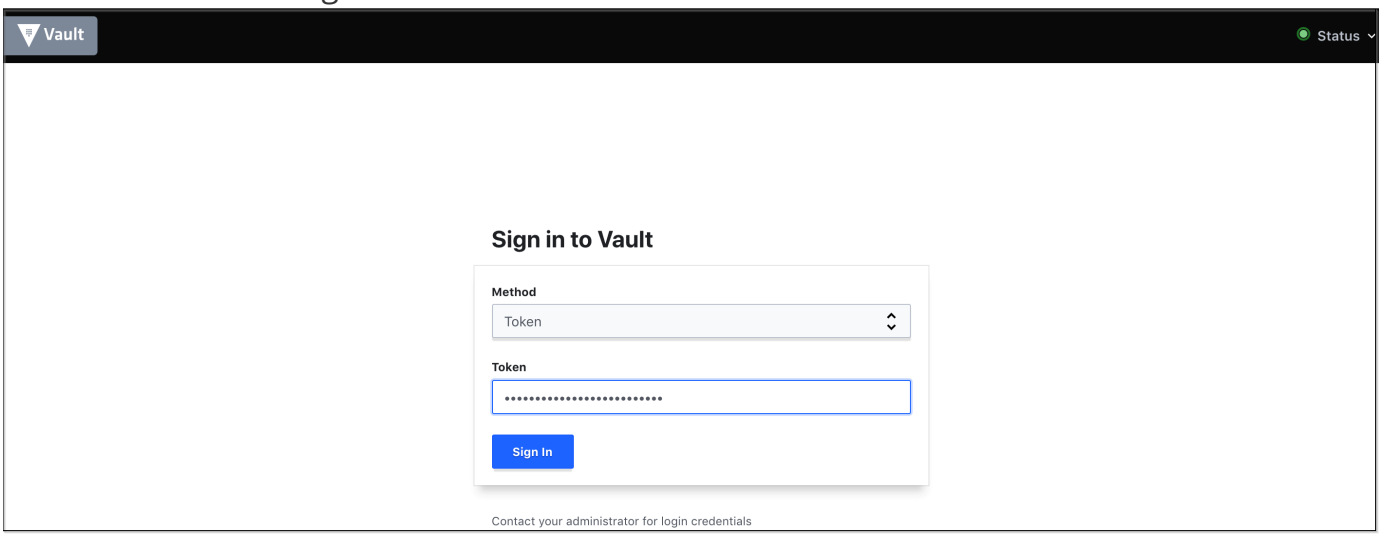
```
rm -rf hooks
```

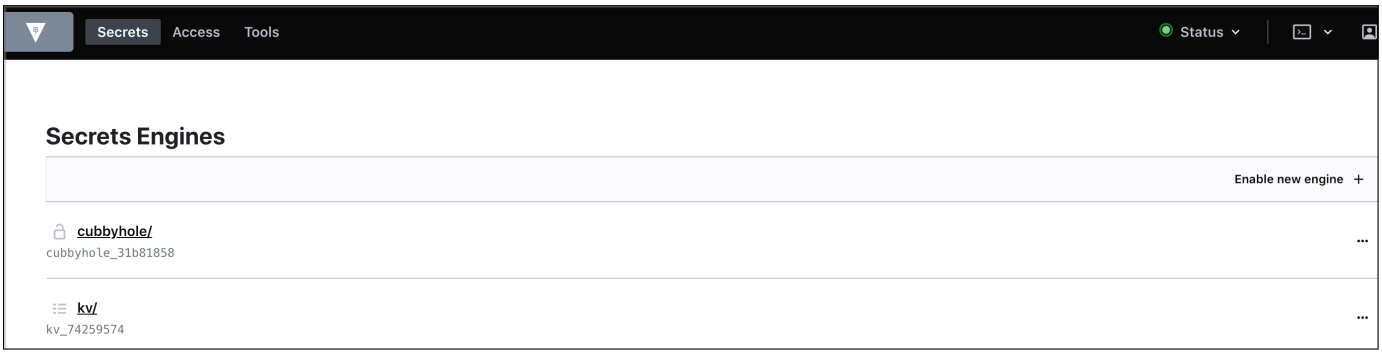
Secret Management

Vault GUI

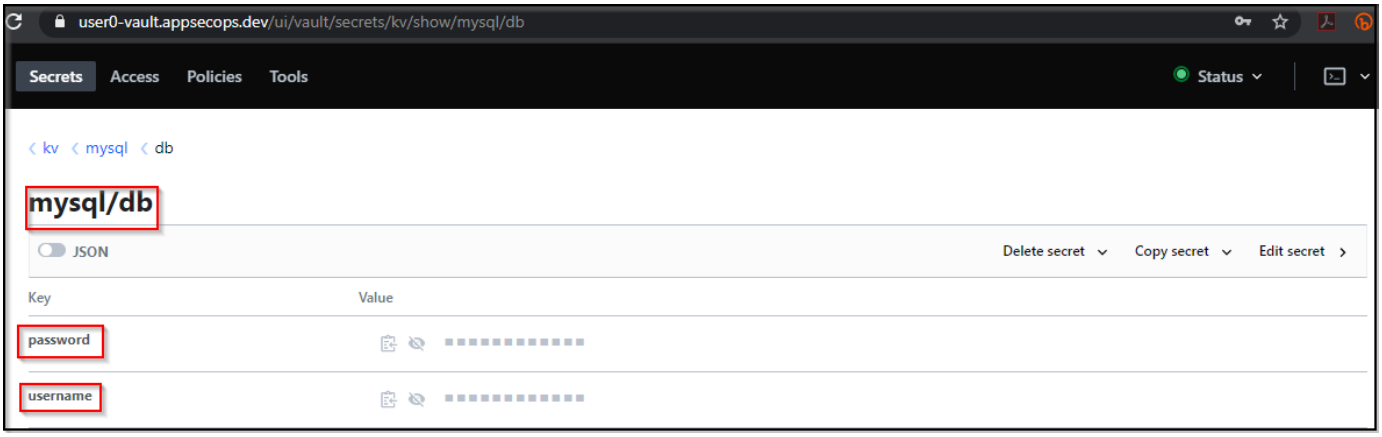
https://user0-vault.appsecops.dev/ui/vault/auth

Access Vault GUI using Root Token





mysql credentials stored in vault

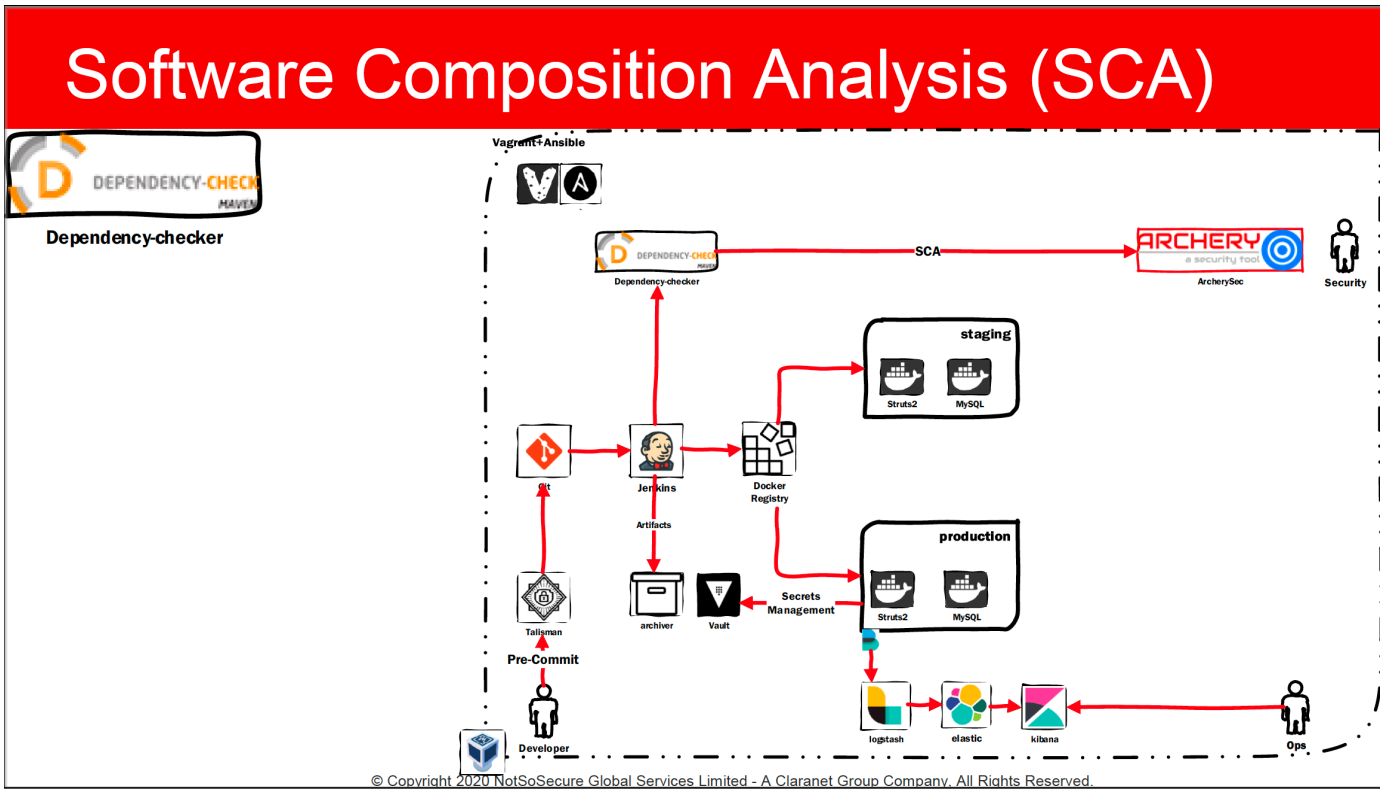


Vault Command Reference

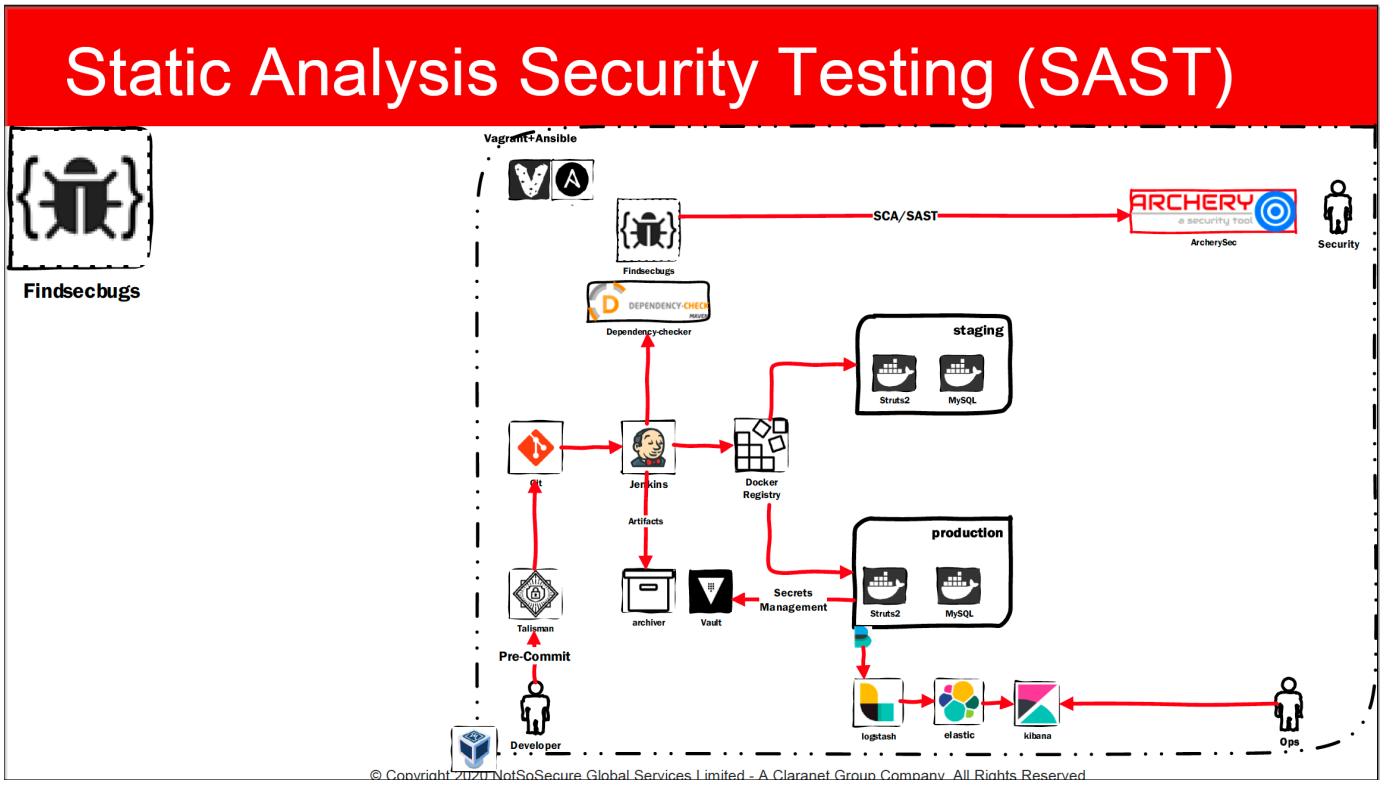
```
vault server -config=/etc/vault/config.json
export VAULT_ADDR=http://127.0.0.1:8200
vault status
vault operator init
vault operator unseal
vault login
vault kv put /secret/server/tomcat/credentials username=admin password=password
vault read /secret/server/tomcat/credentials
vault policy list
vault token create -policy=tomcat
curl -X GET -H "X-Vault-Token:$VAULT_TOKEN"
http://127.0.0.1:8200/v1/secret/server/tomcat/credentials
```

Continuous Delivery

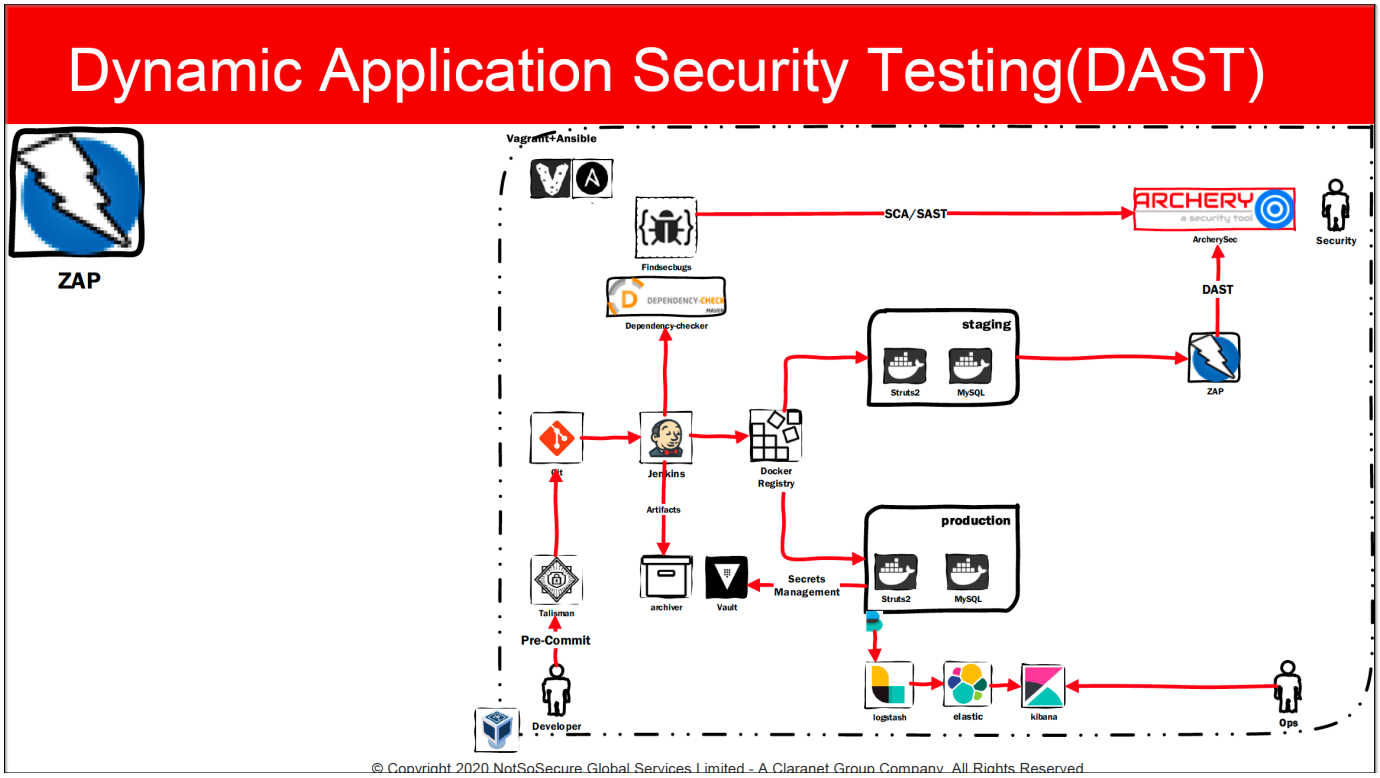
Software Composition Analysis (SCA)



Static Analysis Security Testing (SAST)



Dynamic Analysis Security Testing (DAST)



Software Composition Analysis

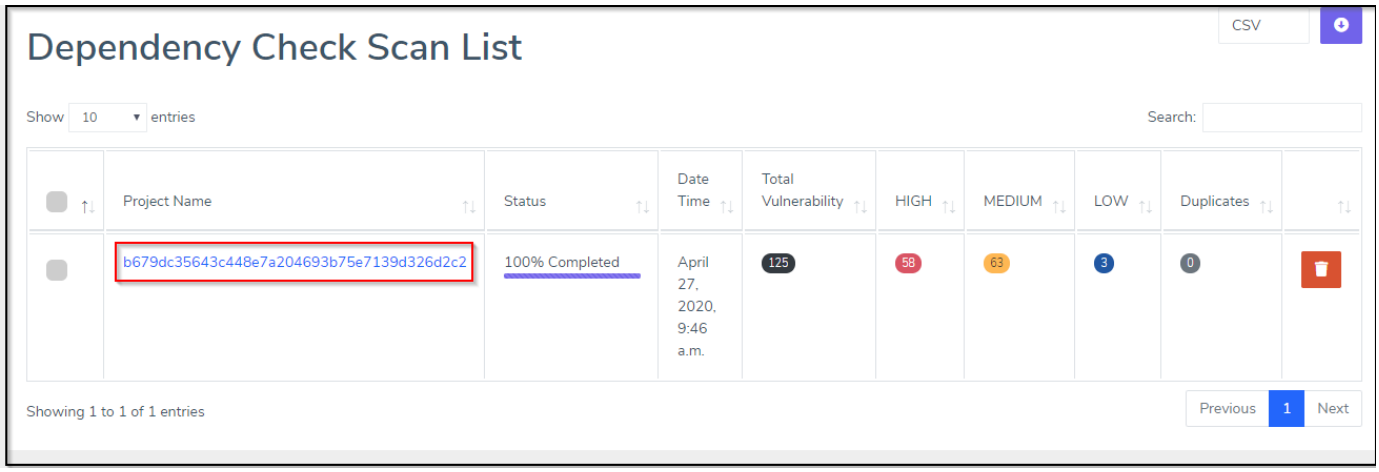
Integrating Dependency check in DevOps pipeline

1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in Jenkinsfile.SCA
2. Copy Contents of `Jenkinsfile.SCA`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

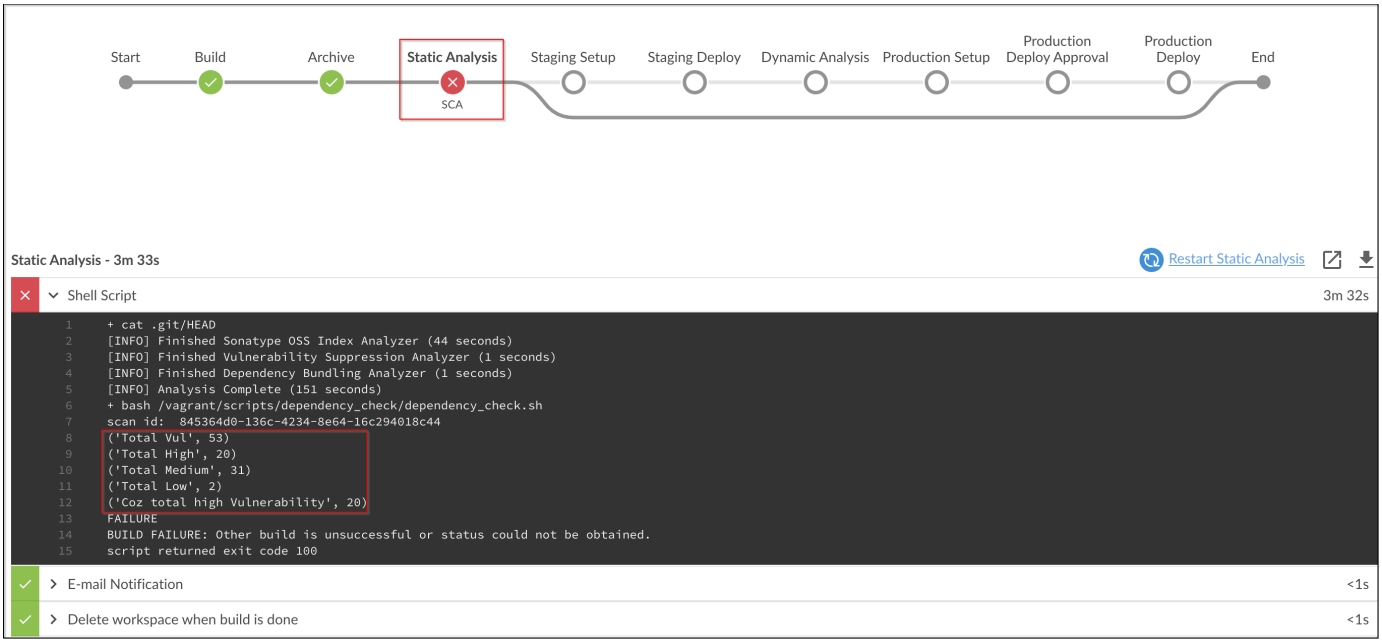
```
git add .
```

```
git commit -am "SCA"
```

Dependency-Check Results in Archerysec



Pipeline



Edit pom.xml file in root folder

Comment out the below Lines

```
<struts.version>2.3.8</struts.version>
<commons.version>4.0</commons.version>
<struts2.tiles>2.3.16</struts2.tiles>
<tiles.core>2.0.6</tiles.core>
```

Add the updated dependencies

```
<struts.version>2.3.37</struts.version>
<commons.version>4.2</commons.version>
<struts2.tiles>2.3.37</struts2.tiles>
<tiles.core>2.2.2</tiles.core>
```

Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "fixed SCA"
```

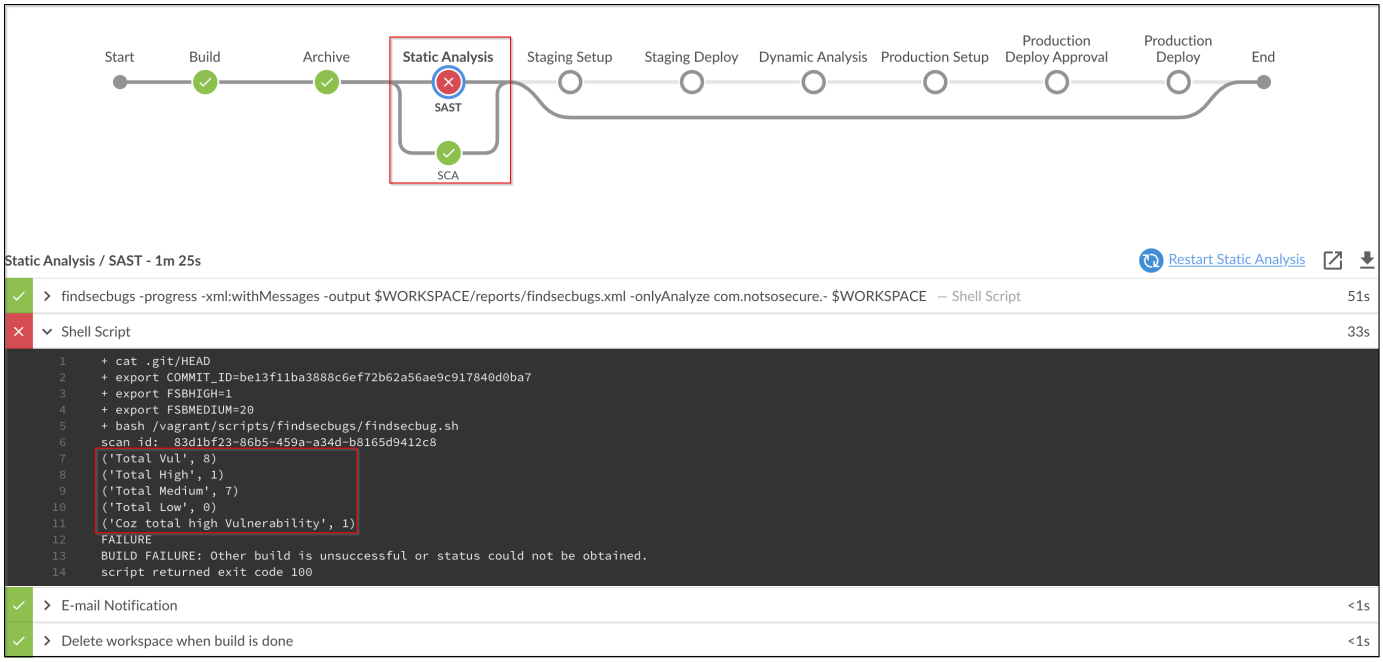
Static Application Security Testing

Integrating FindSecBug in DevOps pipeline

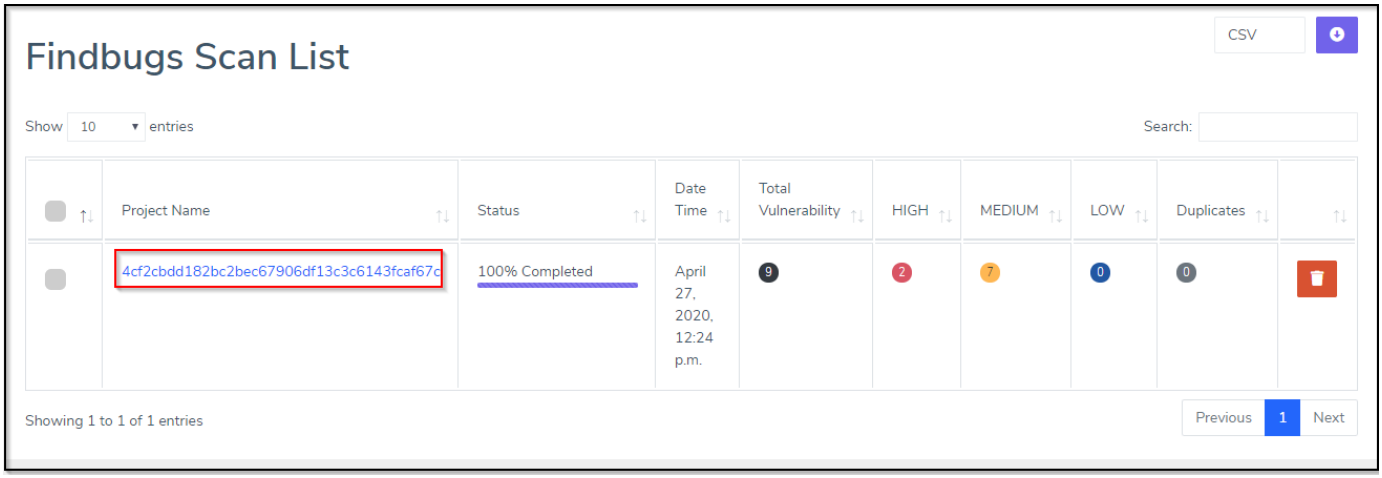
1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.SAST`
2. Copy Contents of `Jenkinsfile.SAST`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "SAST"
```



FindSecBugs Scanning results in Archerysec



Fixed Code

To fix XML_DECODER vulnerability go to the below path and comment the below piece of code displayed

`src/main/java/com/notsosecure/devsecops/actions/api/StudentAddApi.java`

```
//      VULNERABLE CODE
//      @Action("/api/add")
    public String execute() throws Exception {
        System.out.println("Inside Student Add");
        StudentService studentService = new StudentService();
        xmlString = servletRequest.getInputStream();
        XMLDecoder xmlDecoder = new XMLDecoder(xmlString);
        Student student = (Student) xmlDecoder.readObject();
//              Student student = new
Student("test","test","test","test","test","test@test.com");
//              System.out.println(student.getFirstName());
        String result;
        result=studentService.save(student.getUserName(), student.getPassword(),
student.getFirstName(), student.getLastName(),
                dateFormat.format(student.getDateOfBirth()),
student.getEmailAddress());
        PrintWriter printWriter = servletResponse.getWriter();
        switch (result) {
        case "SignupFailure-UserNameExists":
            printWriter.println("Username Exists");
            break;
        case "SignupSuccess":
            printWriter.println("Users Added Successfully");
            break;
        case "SignupFailure":
            printWriter.println("Something Went wrong");
            break;
        default:
            break;
        }

        return NONE;
    }
```

To fix SQL_INJECTION vulnerability go to the below path and replace the existing piece of code with below code

`src/main/java/com/notsosecure/devsecops/service/StudentService.java`

```
public String findByLogin(String userName, String password) {
        String result = "LoginFailure";
        if (studentRepository != null) {
//                      boolean status =
studentRepository.findByLoginSQL(userName, password);
            boolean status = studentRepository.findByLogin(userName, password);
            if (status) {
                result = "LoginSuccess";
            }
        }
        return result;
    }
```

To fix SQL_INJECTION vulnerability go to the below path and comment the below mentioned piece of code and replace with fixed code

`src/main/java/com/notsosecure/devsecops/respository/StudentRepository.java`

Comment this code

```
// VULNERABLE CODE
     public boolean findByLoginSQL(String userName, String password) {

        if (dbConnection != null) {
                try {

                        Statement stmt = dbConnection.createStatement();
                         ResultSet rs;
                         rs = stmt.executeQuery("select password from student
where userName="+userName);
                         if (rs != null) {
                                        while (rs.next()) {
                                                if
(rs.getString(1).equals(password)) {
                                                        return true;
                                                }
                                        }
                                }
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
        return false;
     }
```

Uncomment the below fixed code

```
public boolean findByLogin(String userName, String password) {
        if (dbConnection != null) {
            try {
                PreparedStatement prepStatement = dbConnection
                        .prepareStatement("select password from student where
userName = ?");
                prepStatement.setString(1, userName);

                ResultSet result = prepStatement.executeQuery();
                if (result != null) {
                    while (result.next()) {
                        if (result.getString(1).equals(password)) {
                            return true;
                        }
                    }
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return false;
    }
```

To fix OBJECT_DESERIALIZATION vulnerability go to the below path and comment the below mentioned piece of code and replace with fixed code

`src/main/java/com/notsosecure/devsecops/util/SerializeObject.java`

Comment this code

```
// VULNERABLE CODE
     public static Object stringToObject(String s) {

        byte[] data = Base64Utils.decodeByte(s);
        ObjectInputStream ois;
        Object o = new Object();
        try {
                ois = new ObjectInputStream(new ByteArrayInputStream(data));
                o = ois.readObject();
                ois.close();

        } catch (EOFException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        } catch (ClassNotFoundException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        } catch (IOException e) {
                e.printStackTrace();
        }
        System.out.println(o.toString());
        return o;
     }
```

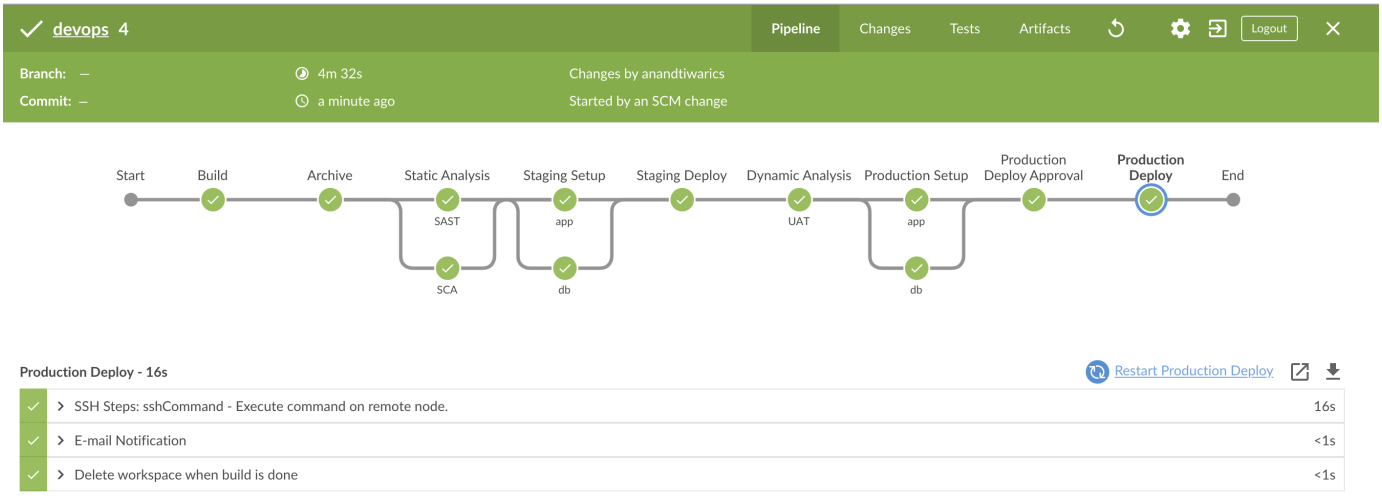
Uncomment the below fixed code

```
    public static Object stringToObject(String s) {

        return new StudentSer("goku");

    }
```

```
git add .
```

```
git commit -am "Fixed SAST"
```

Fire the below Git Commands to execute the Pipeline


devops 4
Pipeline
Changes
Tests
Artifacts
Logout
Branch: –
Commit: –
4m 32s
a minute ago
Changes by anandtiwarics
Started by an SCM change
Start
Build
Archive
Static Analysis
SAST
SCA
Staging Setup
app
db
Staging Deploy
Dynamic Analysis
UAT
Production Setup
app
db
Production Deploy Approval
Production Deploy
End

Production Deploy - 16s
Restart Production Deploy
SSH Steps: sshCommand - Execute command on remote node. 16s
E-mail Notification <1s
Delete workspace when build is done <1s

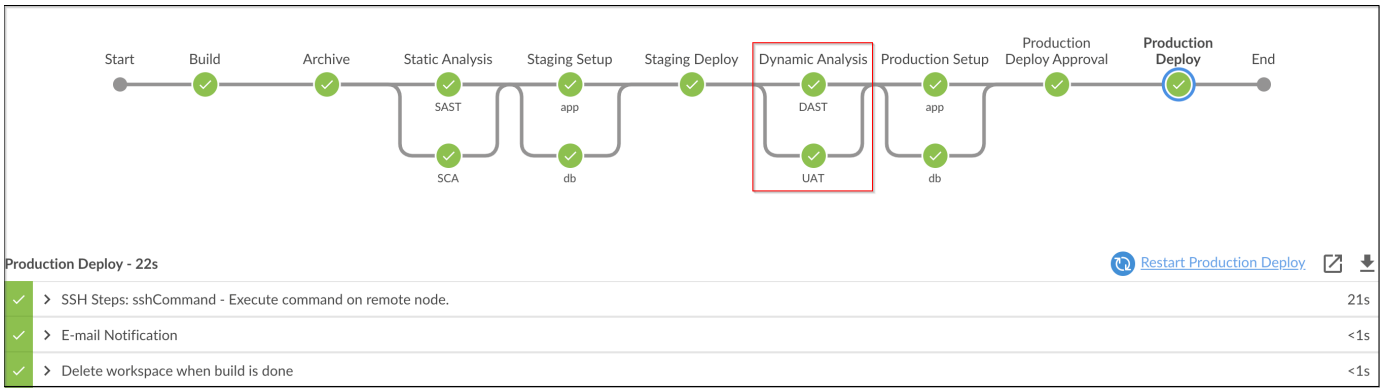
Dynamic Analysis

Integrating OWASP ZAP in DevOps pipeline

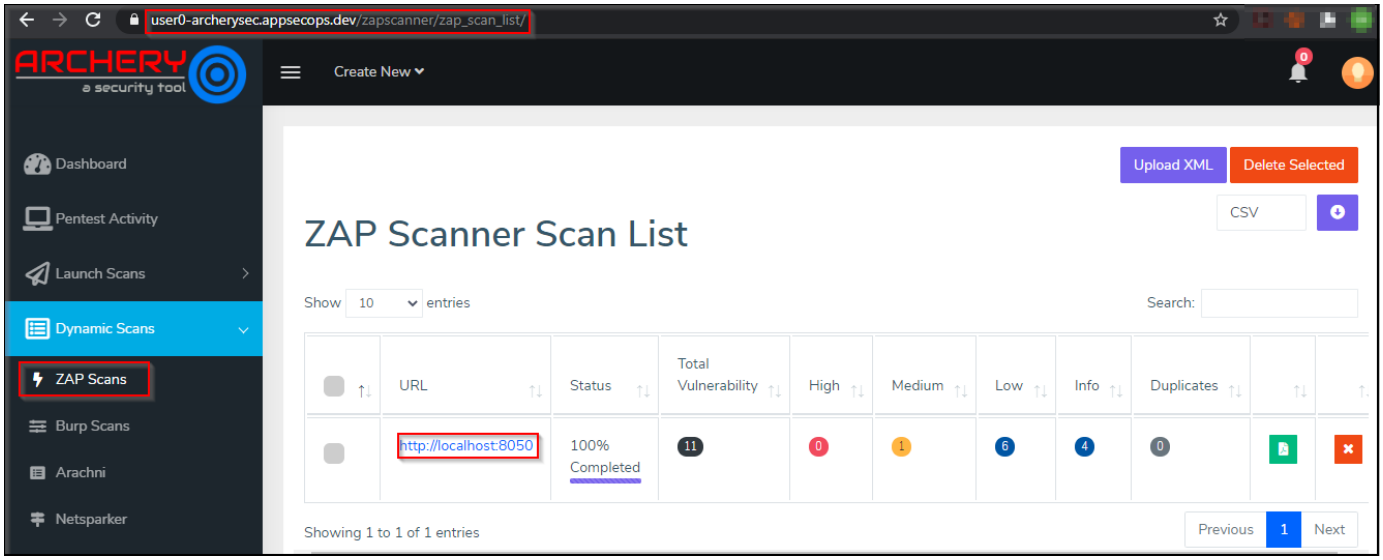
1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.DAST`
2. Copy Contents of `Jenkinsfile.DAST`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "DAST"
```

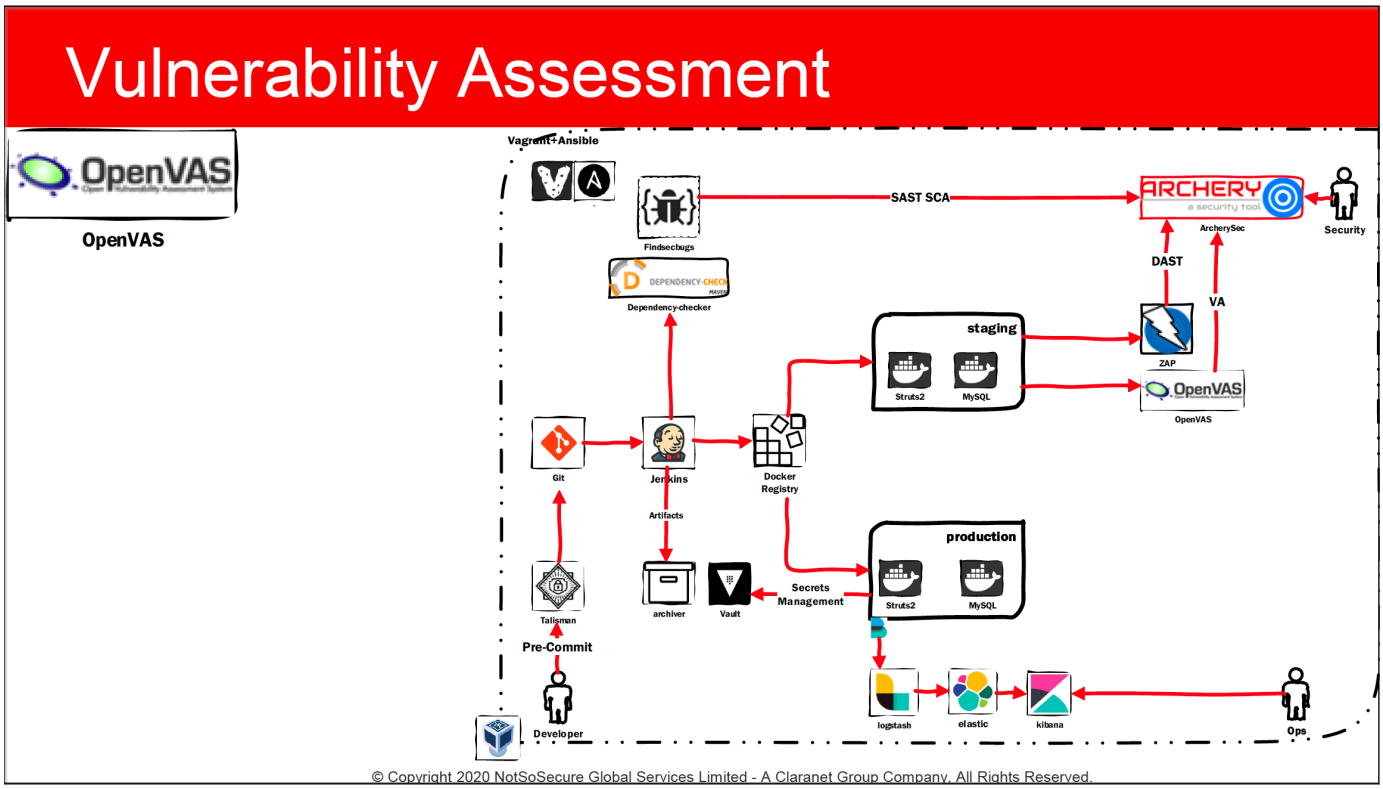


OWASP Zap Scanner results in Archerysec

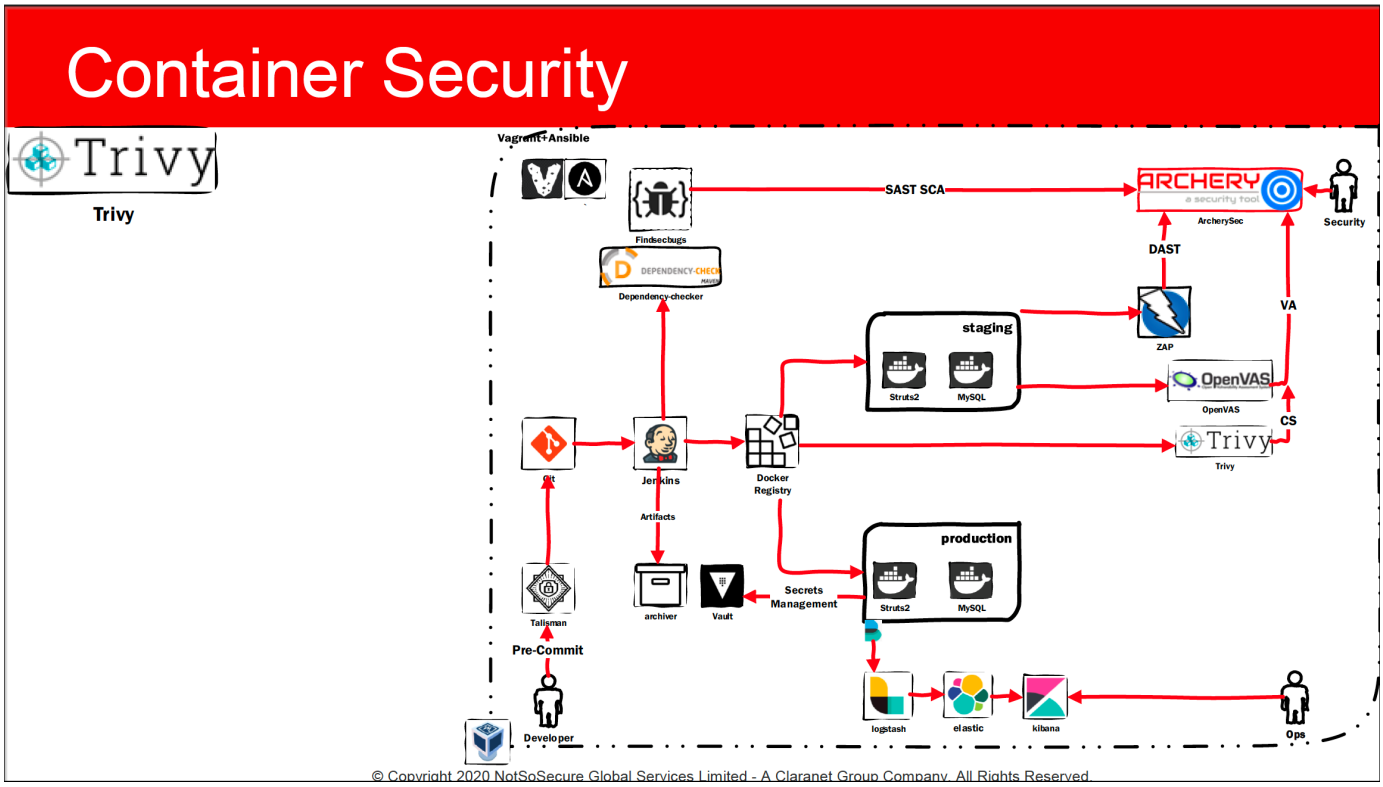


Infrastructure As Code

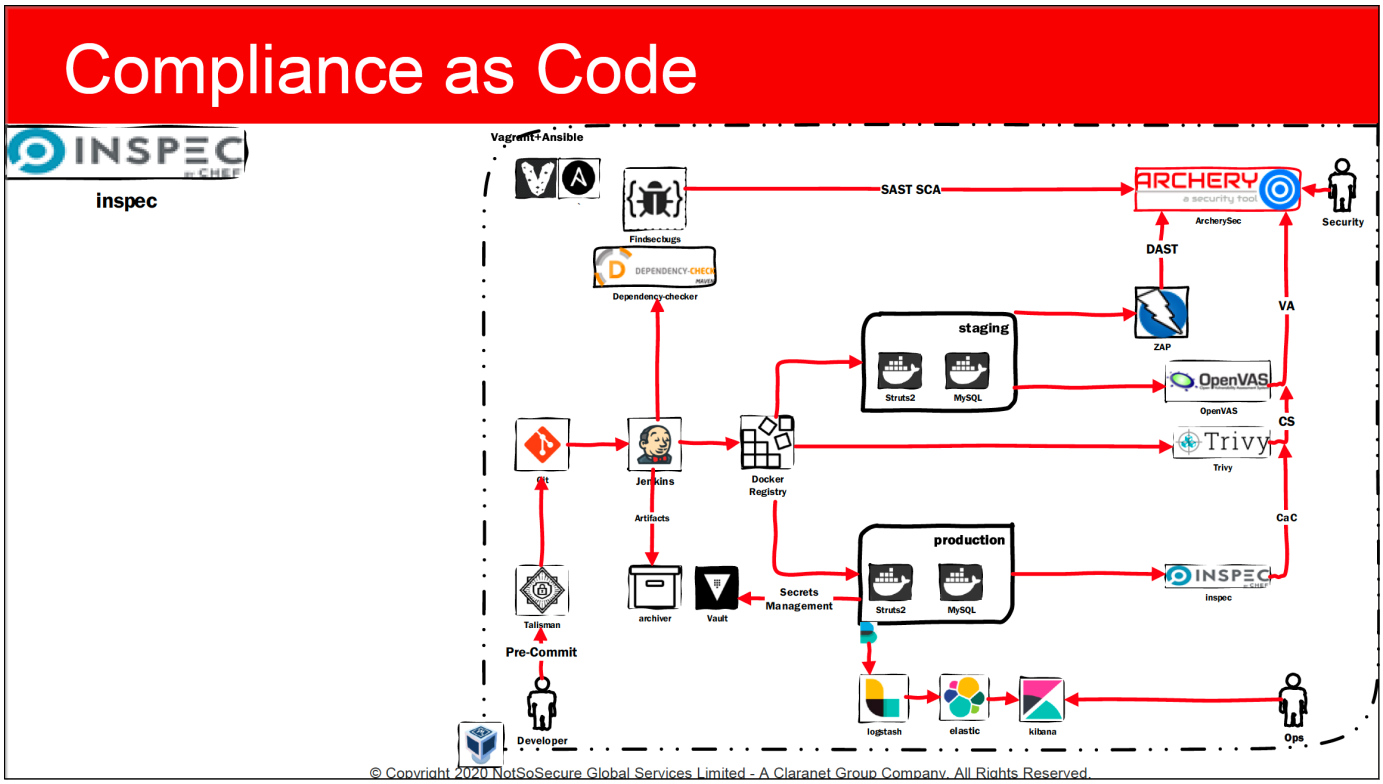
Vulnerability Assessment (VA)



Container Security (CS)



Compliance as Code (CaC)



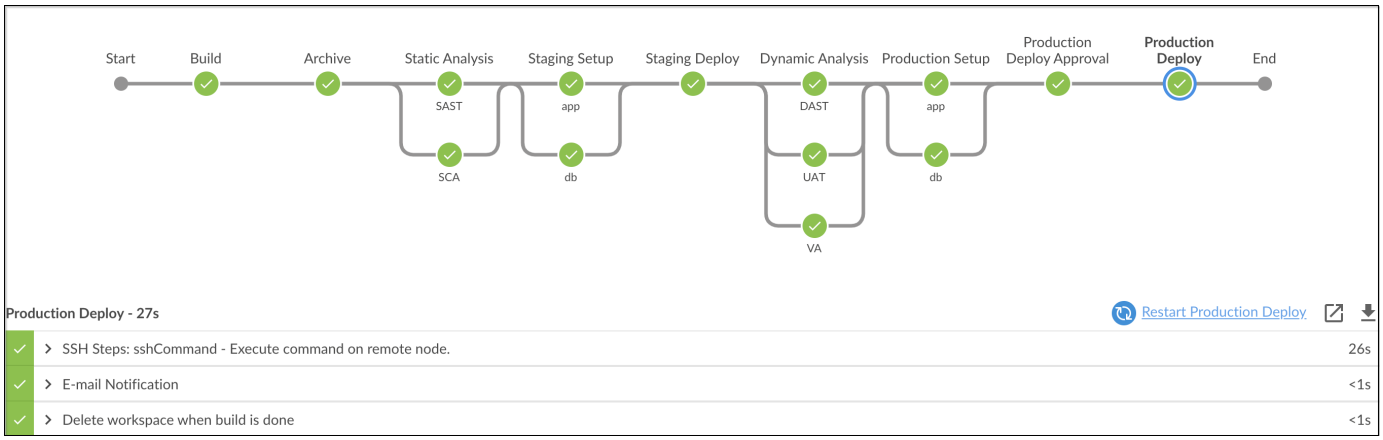
Vulnerability Assessment

Integrating OpenVAS Scanner in DevOps pipeline

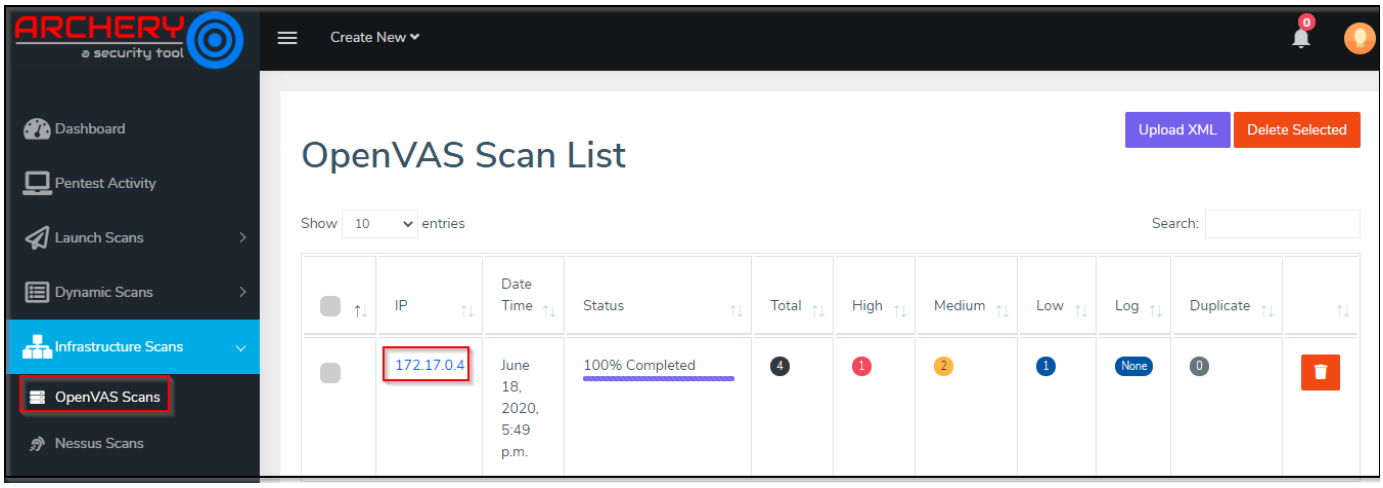
1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.VA`
2. Copy Contents of `Jenkinsfile.VA`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "VA"
```



OpenVAS Scan Results in Archerysec



OpenVAS Scanner https://user0-openvas.appsecops.dev/login/login.html

username:admin

password:admin

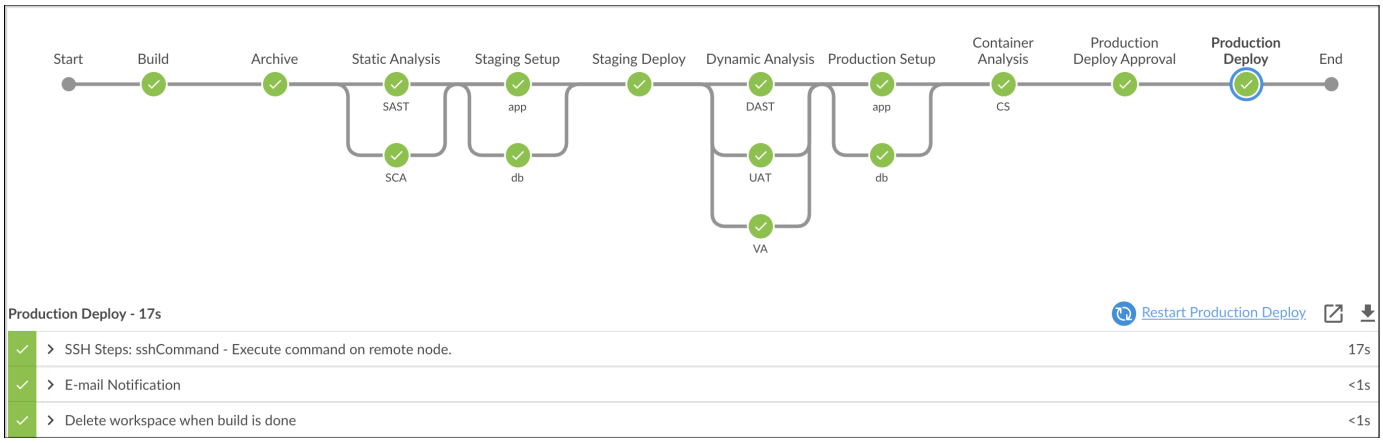
Container Scanning

Integrating Trivy in DevOps pipeline

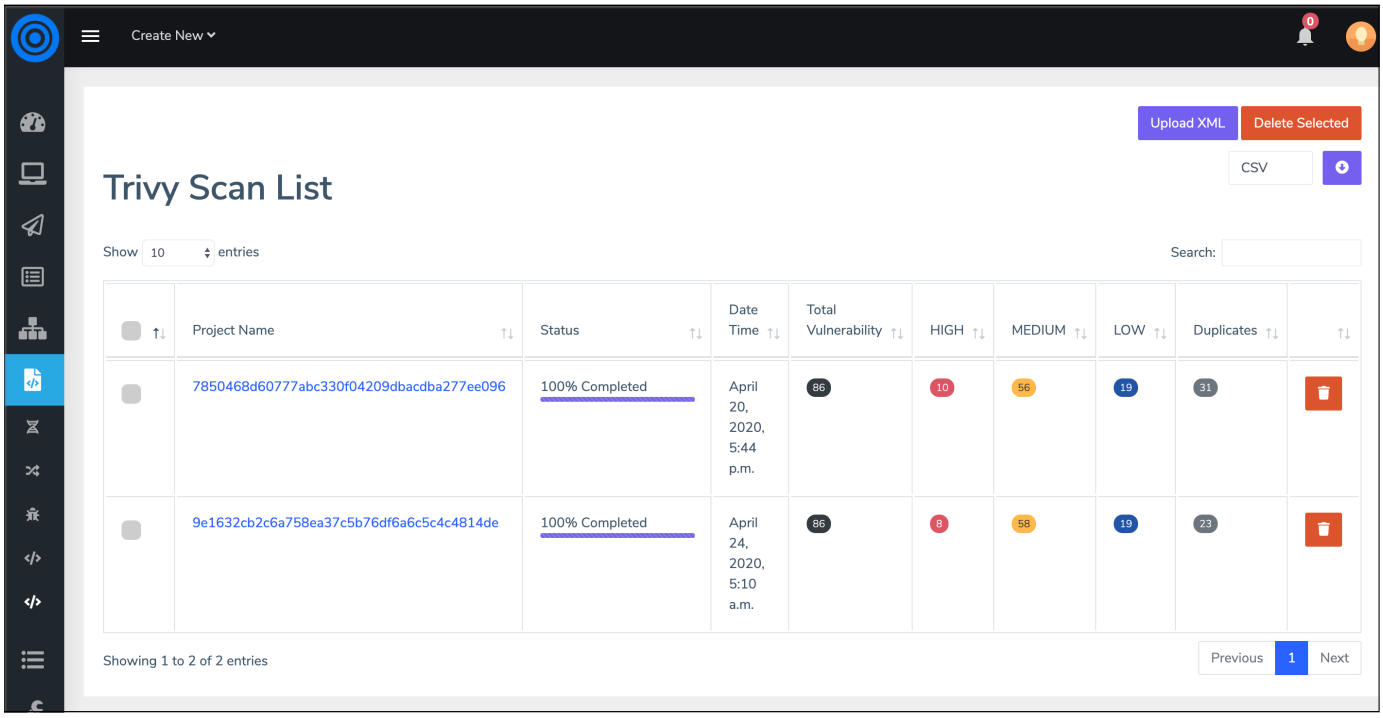
1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.CS`
2. Copy Contents of `Jenkinsfile.CS`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "CS"
```



Trivy Results In Archerysec


Create New
Upload XML
Delete Selected
CSV
Trivy Scan List
Show 10 entries
Search:
Project Name
Status
Date Time
Total Vulnerability
HIGH
MEDIUM
LOW
Duplicates
7850468d60777abc330f04209dbacdba277ee096
100% Completed
April 20, 2020, 5:44 p.m.
86
10
56
19
31
9e1632cb2c6a758ea37c5b76df6a6c5c4c4814de
100% Completed
April 24, 2020, 5:10 a.m.
86
8
58
19
23
Showing 1 to 2 of 2 entries
Previous
1
Next

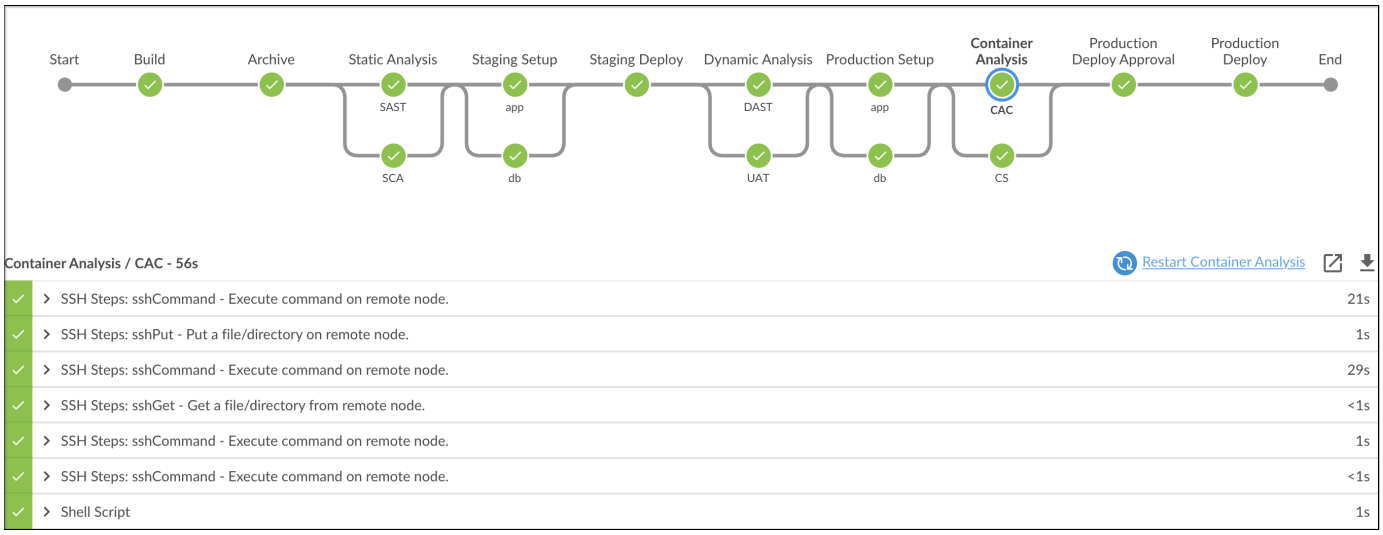
Compliance As a Code Scan

Integrating Inspec in DevOps pipeline

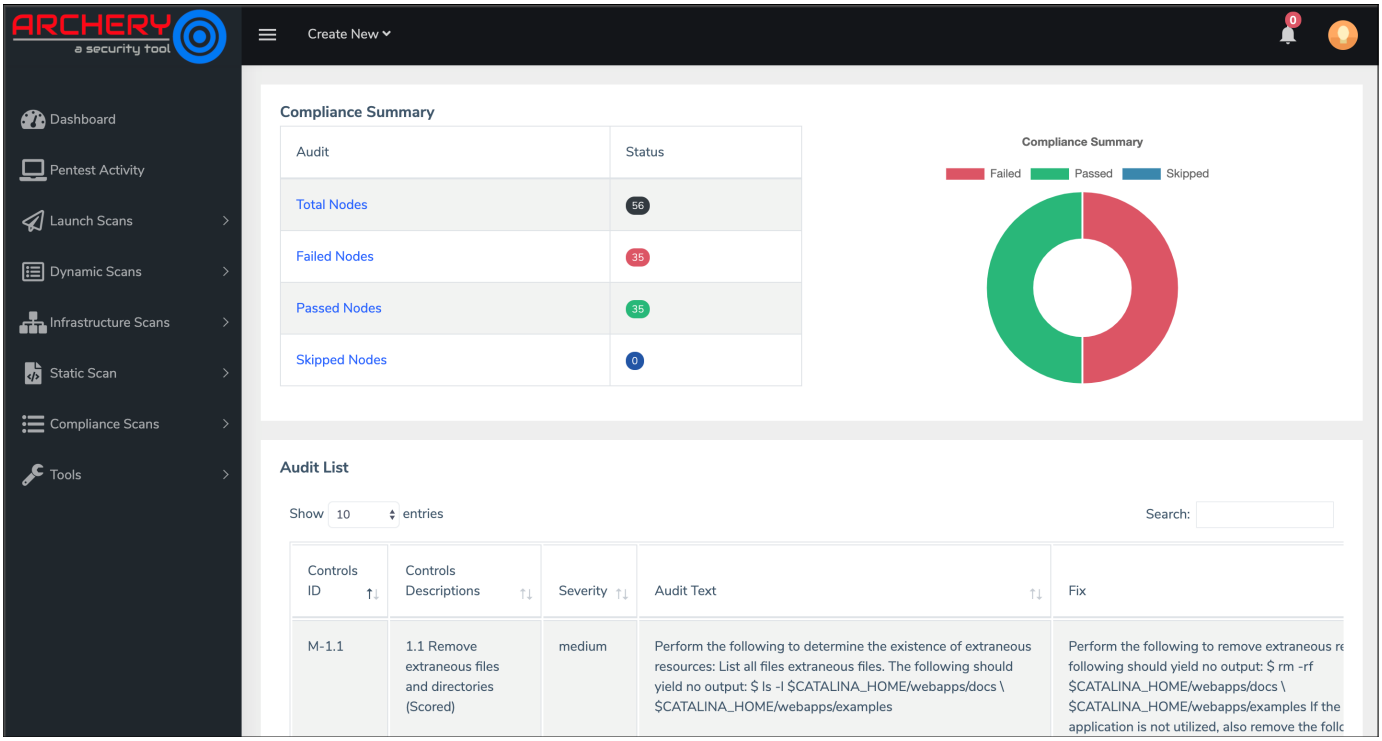
1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.CAC`
2. Copy Contents of `Jenkinsfile.CAC`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

```
git add .
```

```
git commit -am "CAC"
```

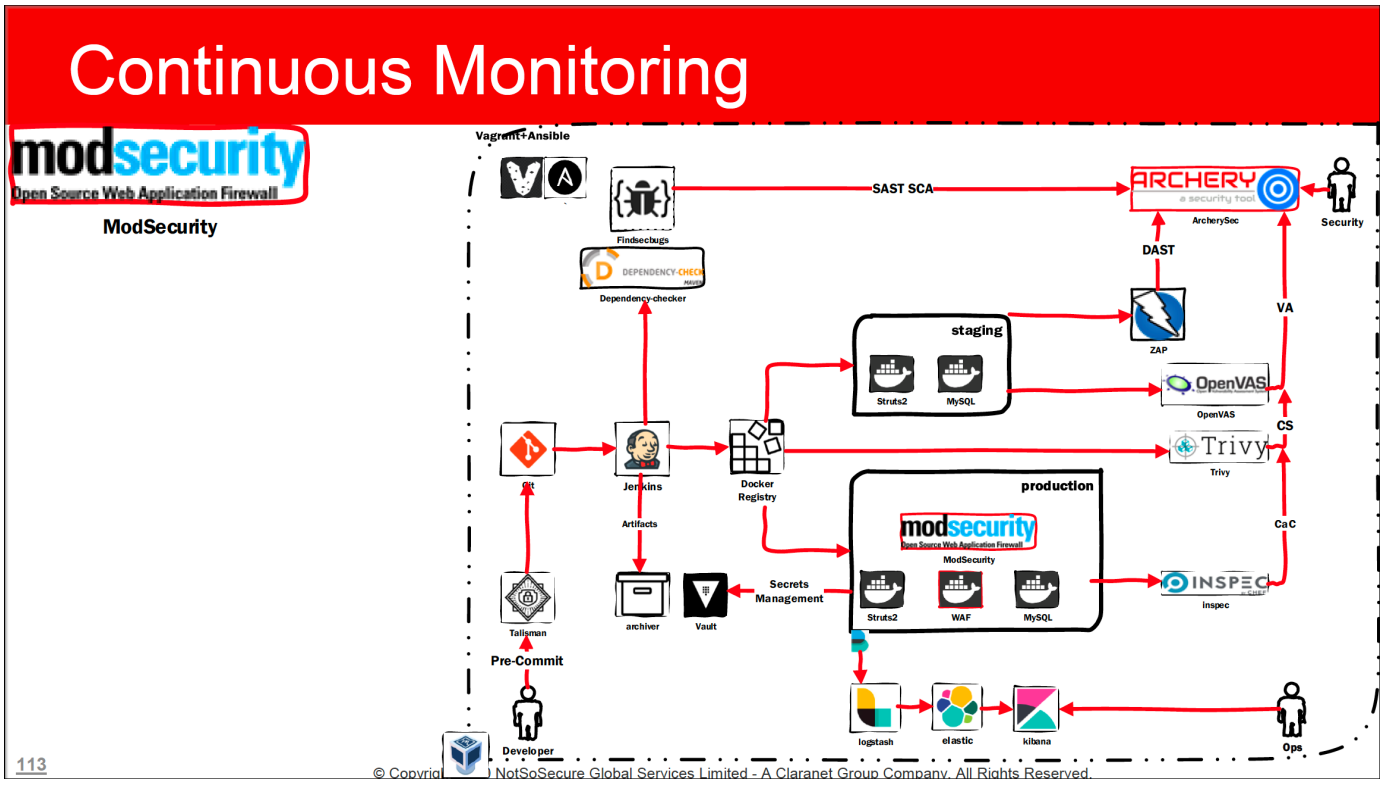


Inspec results in Archerysec


ARCHERY
a security tool
Create New
Dashboard
Pentest Activity
Launch Scans
Dynamic Scans
Infrastructure Scans
Static Scan
Compliance Scans
Tools
Compliance Summary
Audit
Status
Total Nodes
56
Failed Nodes
35
Passed Nodes
35
Skipped Nodes
0
Compliance Summary
Failed
Passed
Skipped
Audit List
Show 10 entries
Search:
Controls ID
Controls Descriptions
Severity
Audit Text
Fix
M-1.1
1.1 Remove extraneous files and directories (Scored)
medium
Perform the following to determine the existence of extraneous resources: List all files extraneous files. The following should yield no output: $ ls -l $CATALINA_HOME/webapps/docs \ $CATALINA_HOME/webapps/examples
Perform the following to remove extraneous re following should yield no output: $ rm -rf $CATALINA_HOME/webapps/docs \ $CATALINA_HOME/webapps/examples If the application is not utilized, also remove the follo

Continuous Monitoring

Alerting and Monitoring



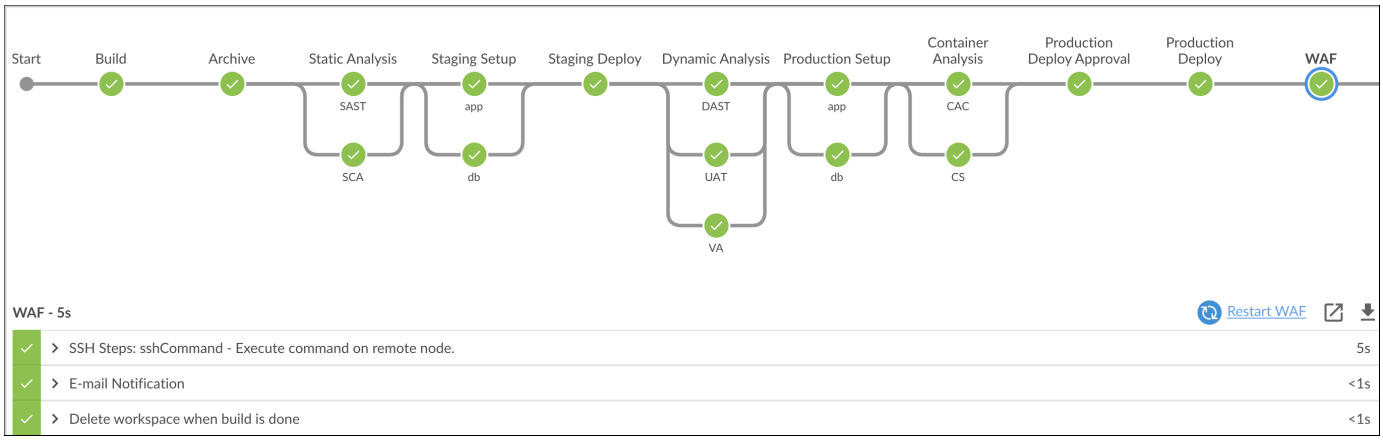
Web Application Firewall

Integrating WAF in DevOps pipeline

1. Don't forget to copy `VAULT_TOKEN_MYSQL` value with your generated code in `Jenkinsfile.WAF`
2. Copy Contents of `Jenkinsfile.WAF`
3. Paste the Contents into `Jenkinsfile` and Commit the code.
4. Fire the below Git Commands to execute the Pipeline

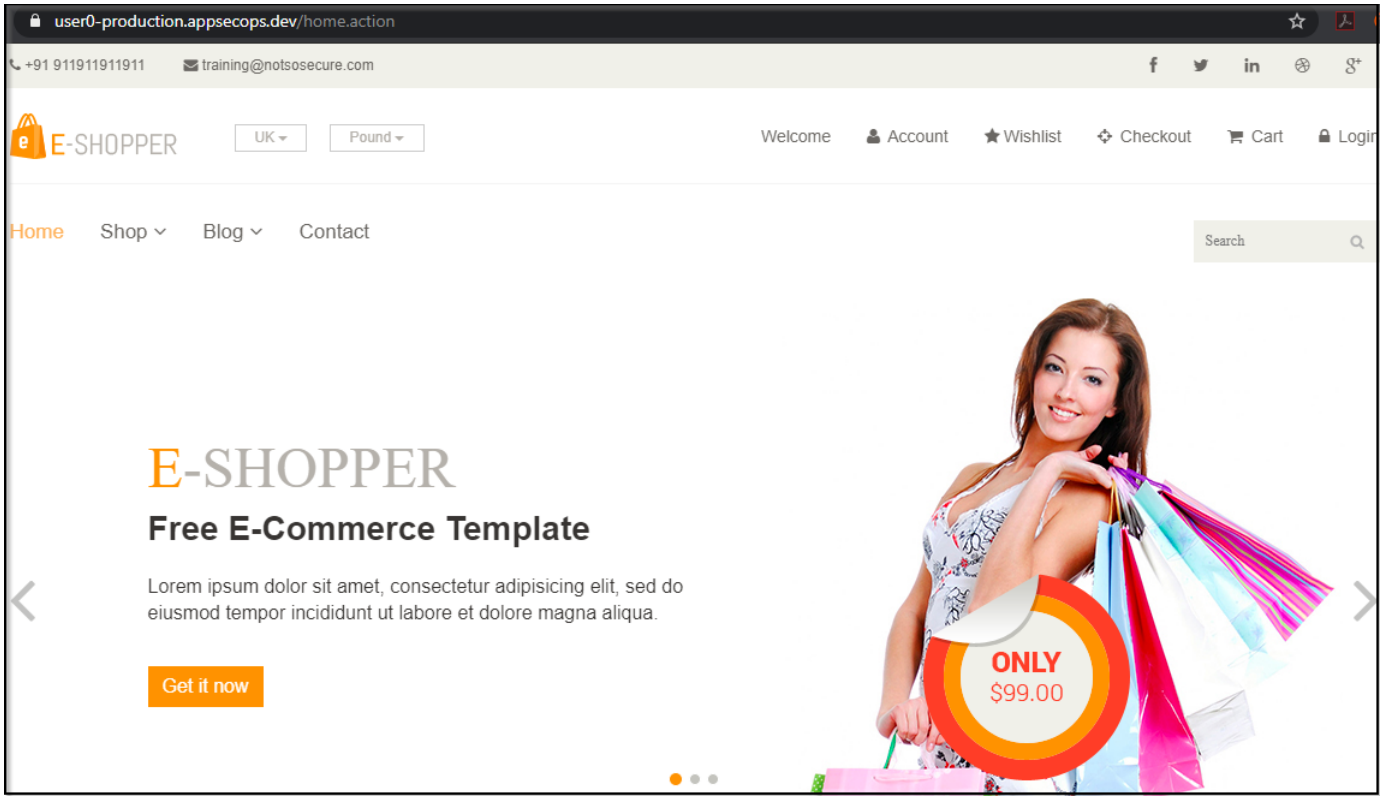
```
git add .
```

```
git commit -am "WAF"
```



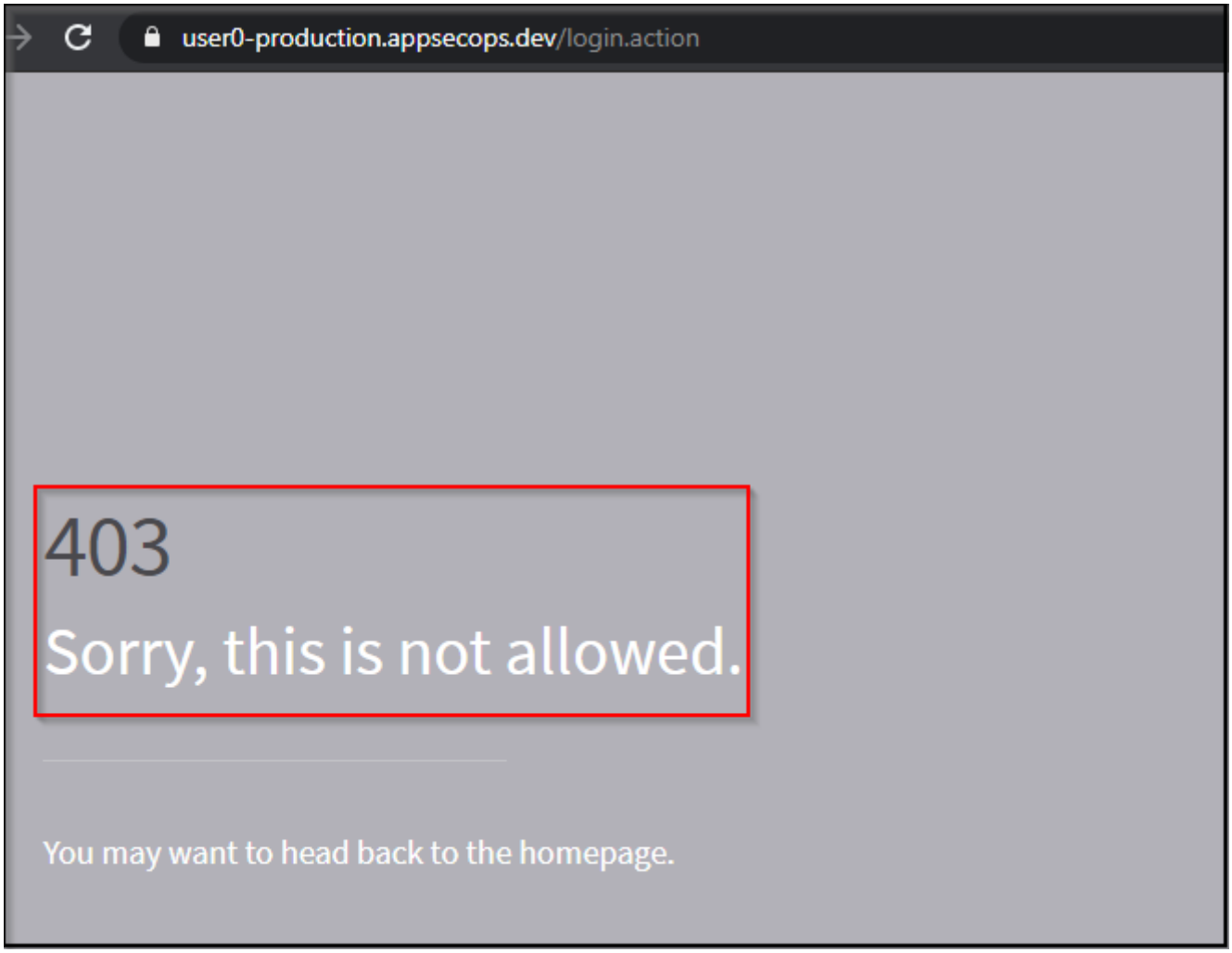
Application in Production Behind WAF

https://user0-production.appsecops.dev/view.action



Perform some attack

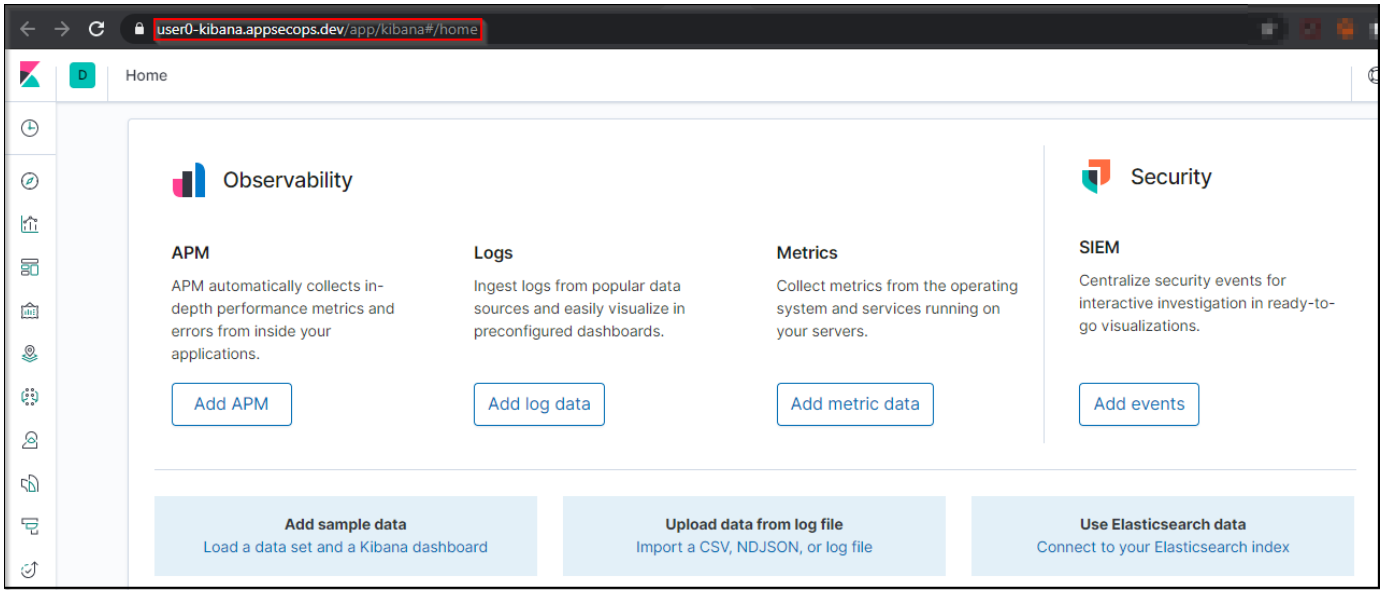
https://user0-production.appsecops.dev/view.action



Monitoring and Logging

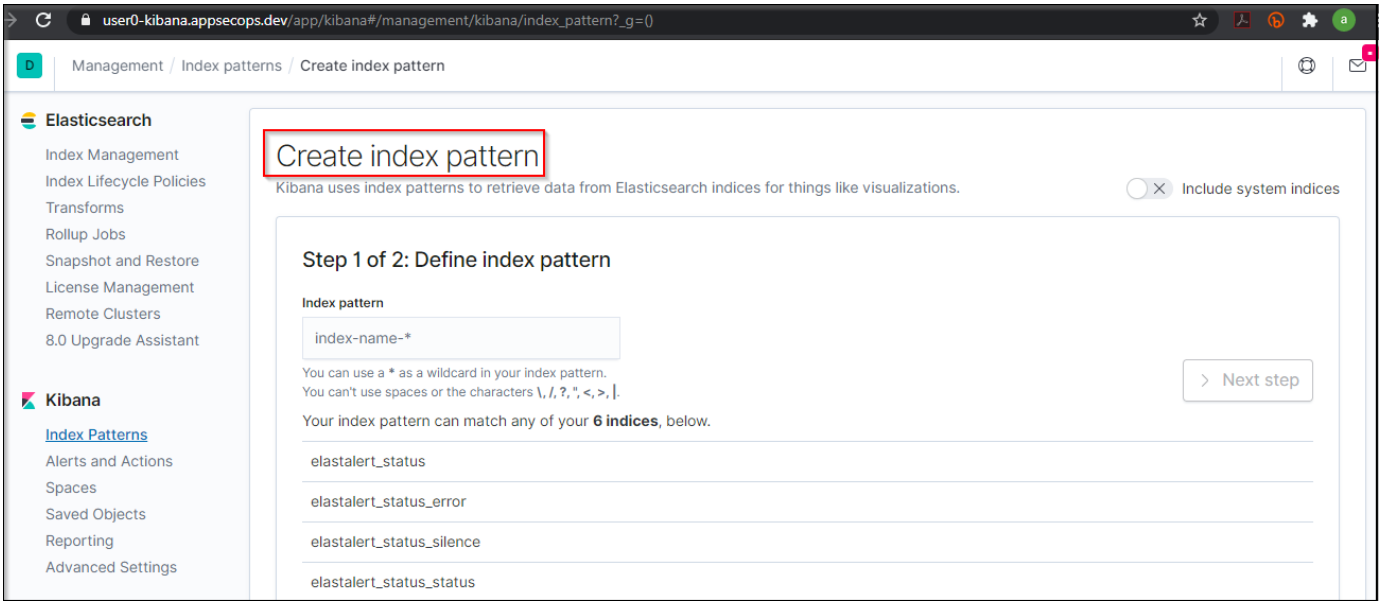
Kibana

https://user0-kibana.appsecops.dev/app/kibana#/home

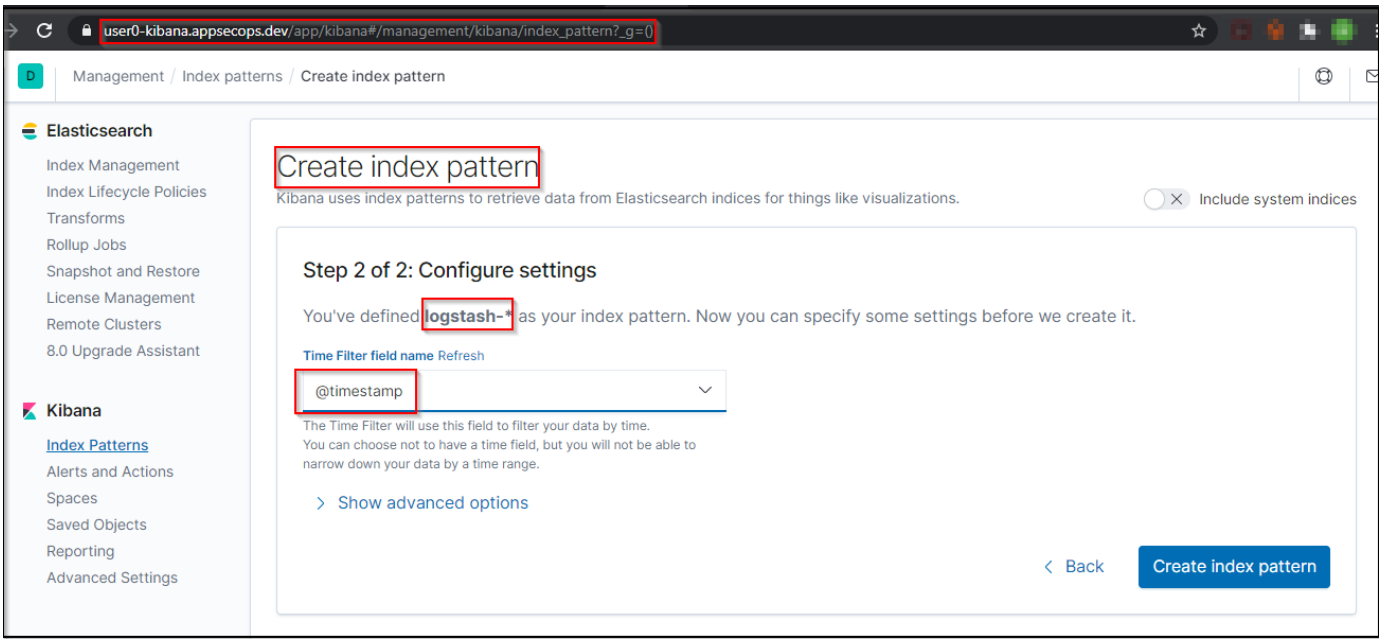


Setup up Kibana

1. Go to https://user0-kibana.appsecops.dev/app/kibana#/management?_g=()
2. Click on Index pattern
3. Create Index pattern
4. Type `logstash-*` in Index pattern
5. Next setup

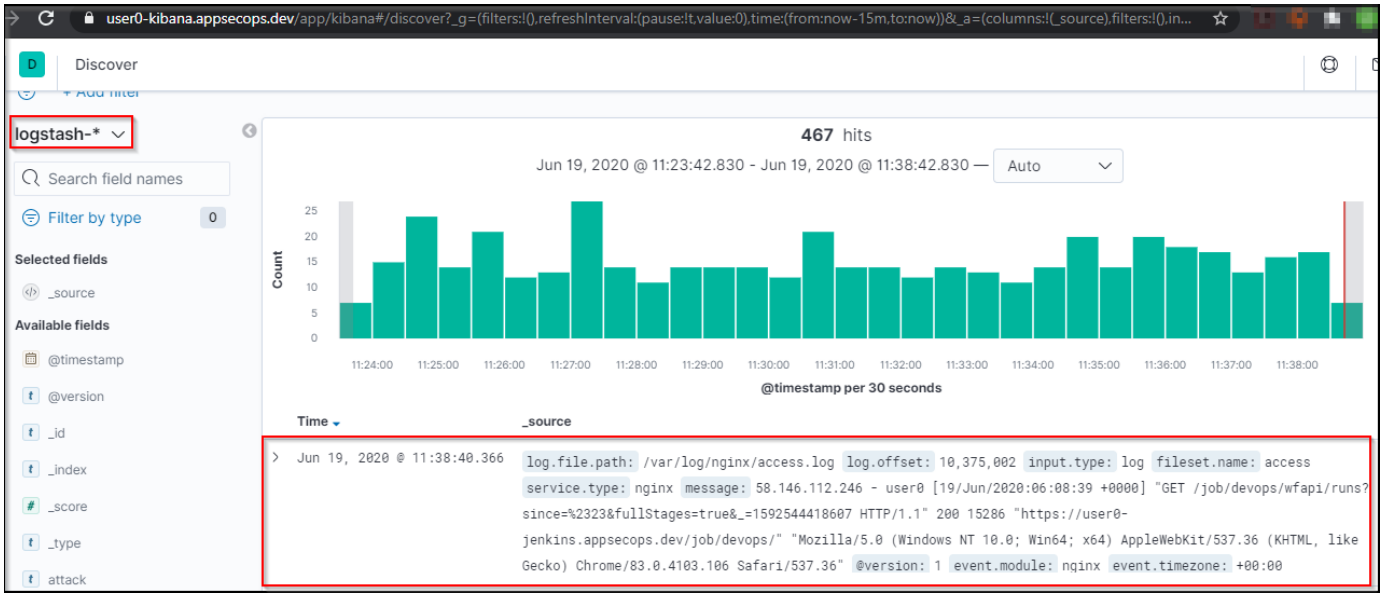


6. Time Filter field name `@timestamp`

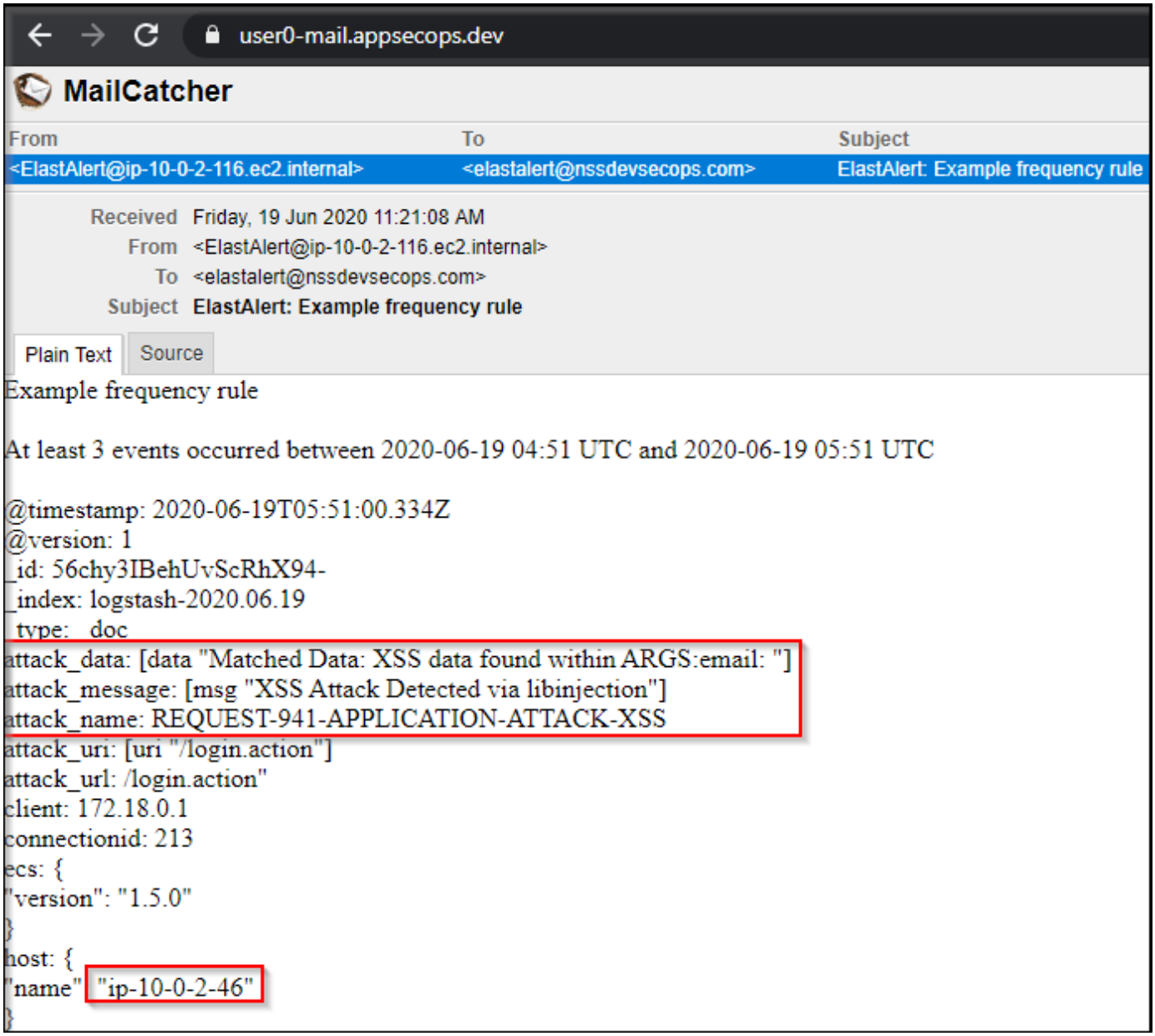


7. Create Index pattern

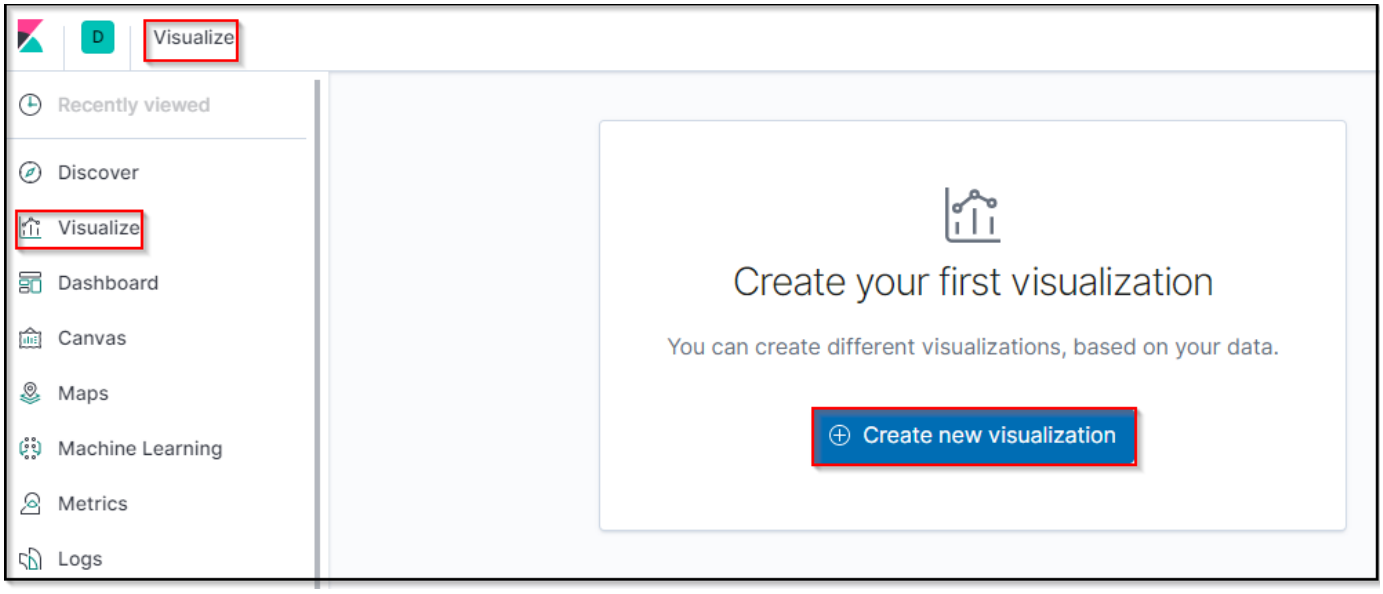
8. Go to Discover

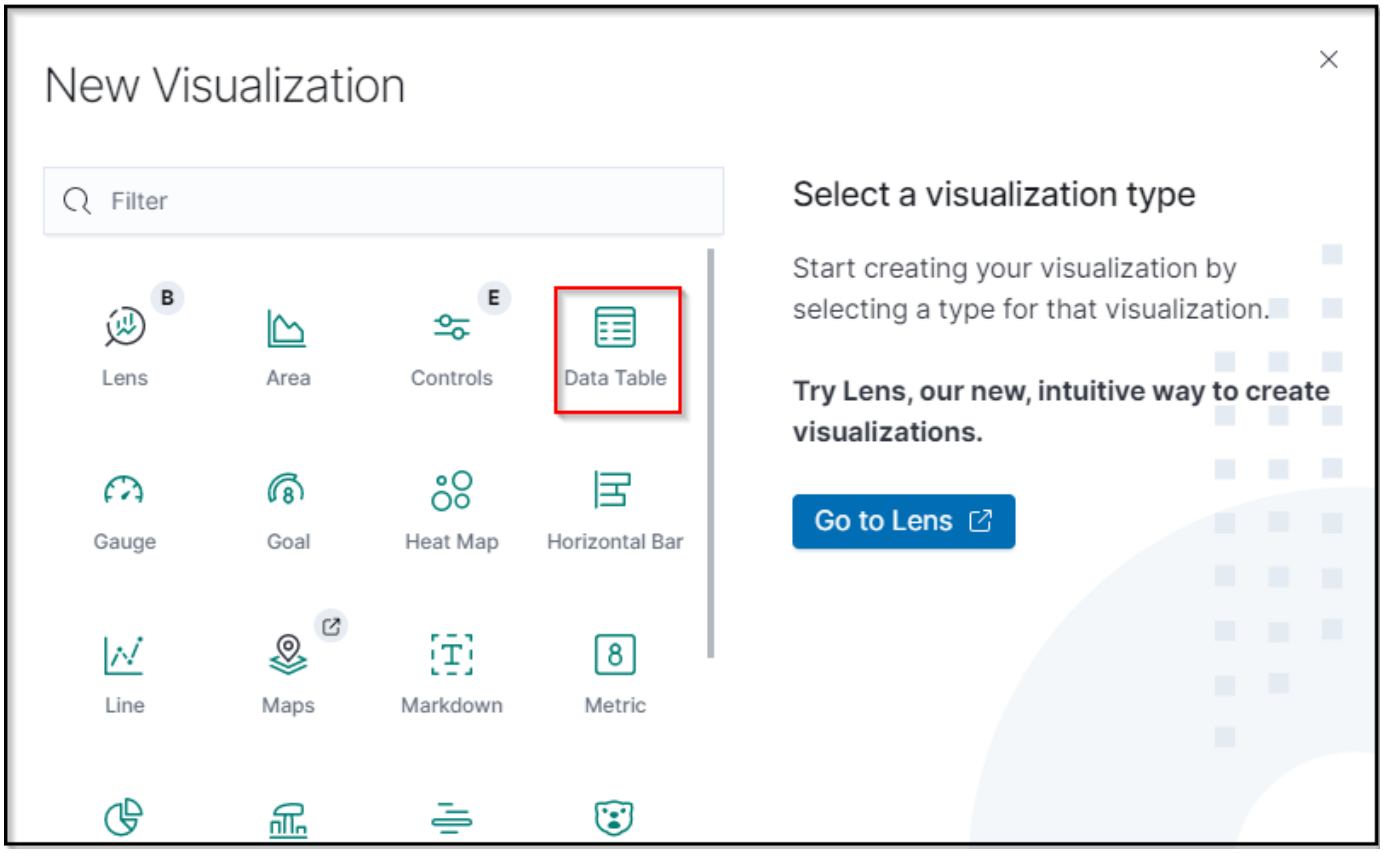


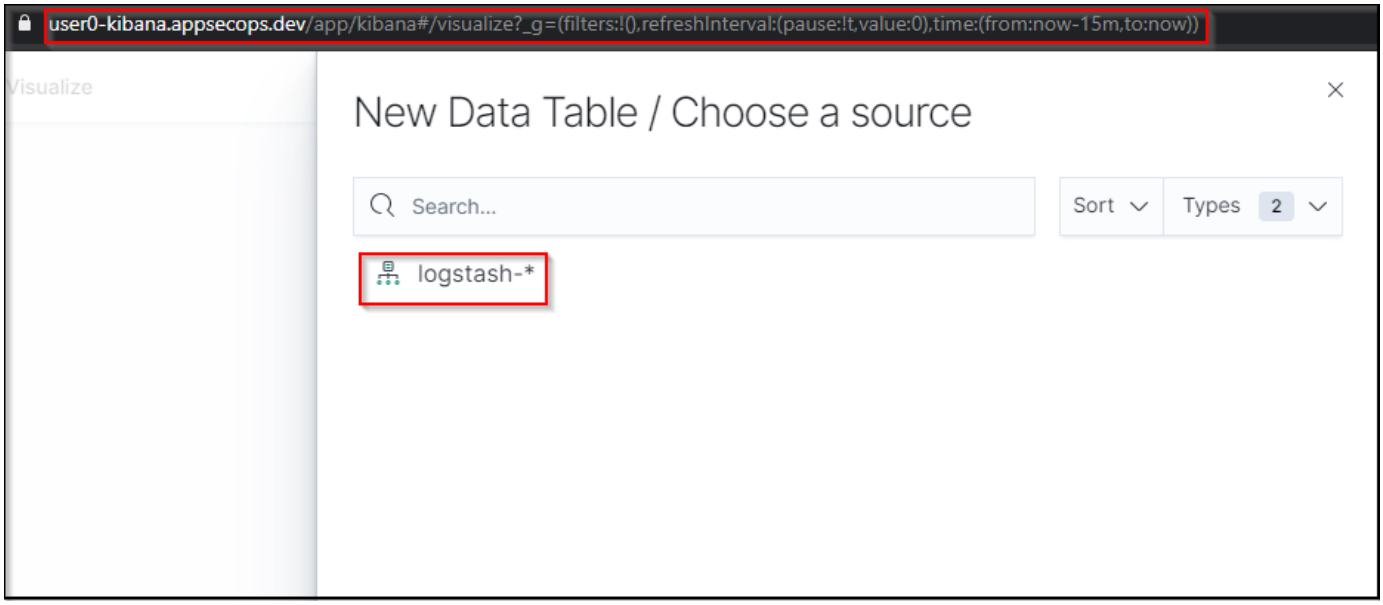
9. Open Mail Server

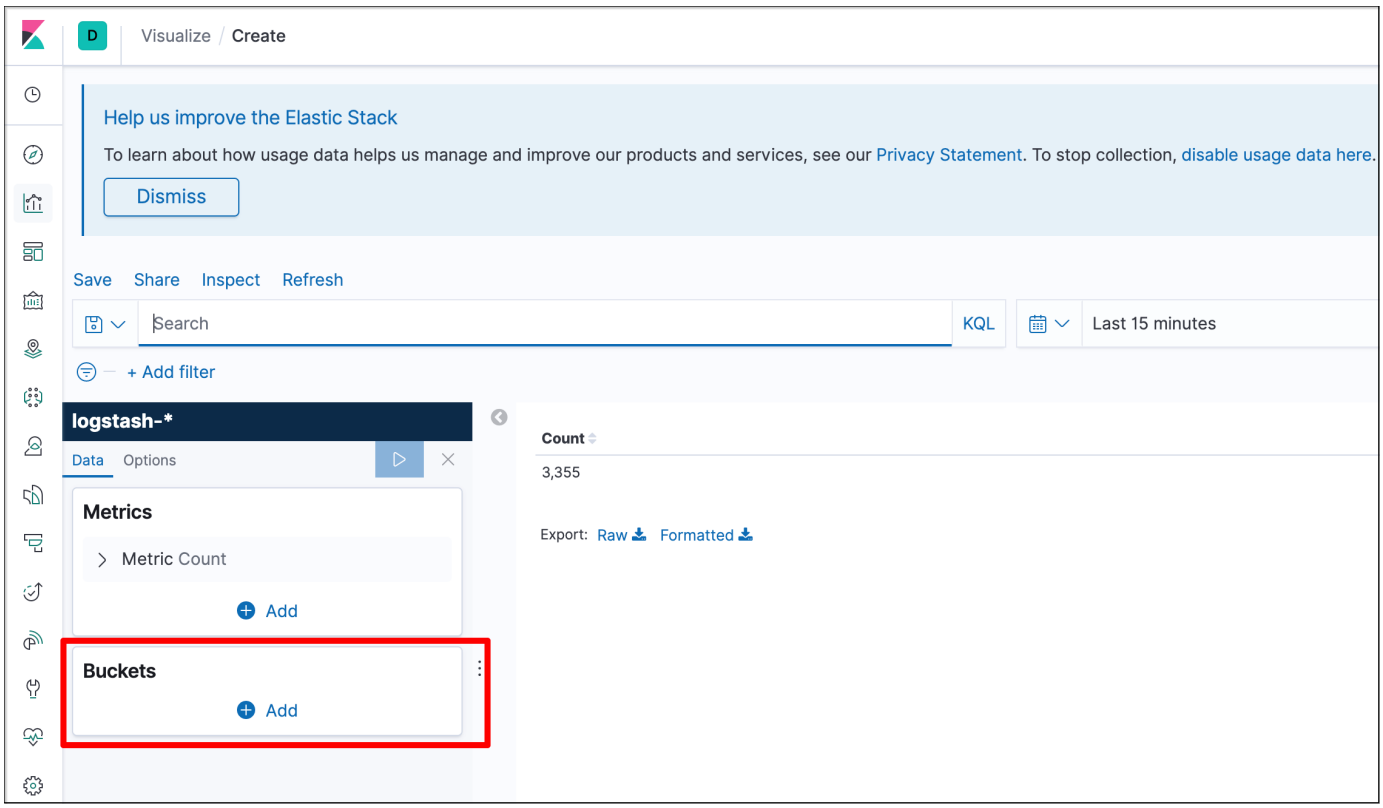


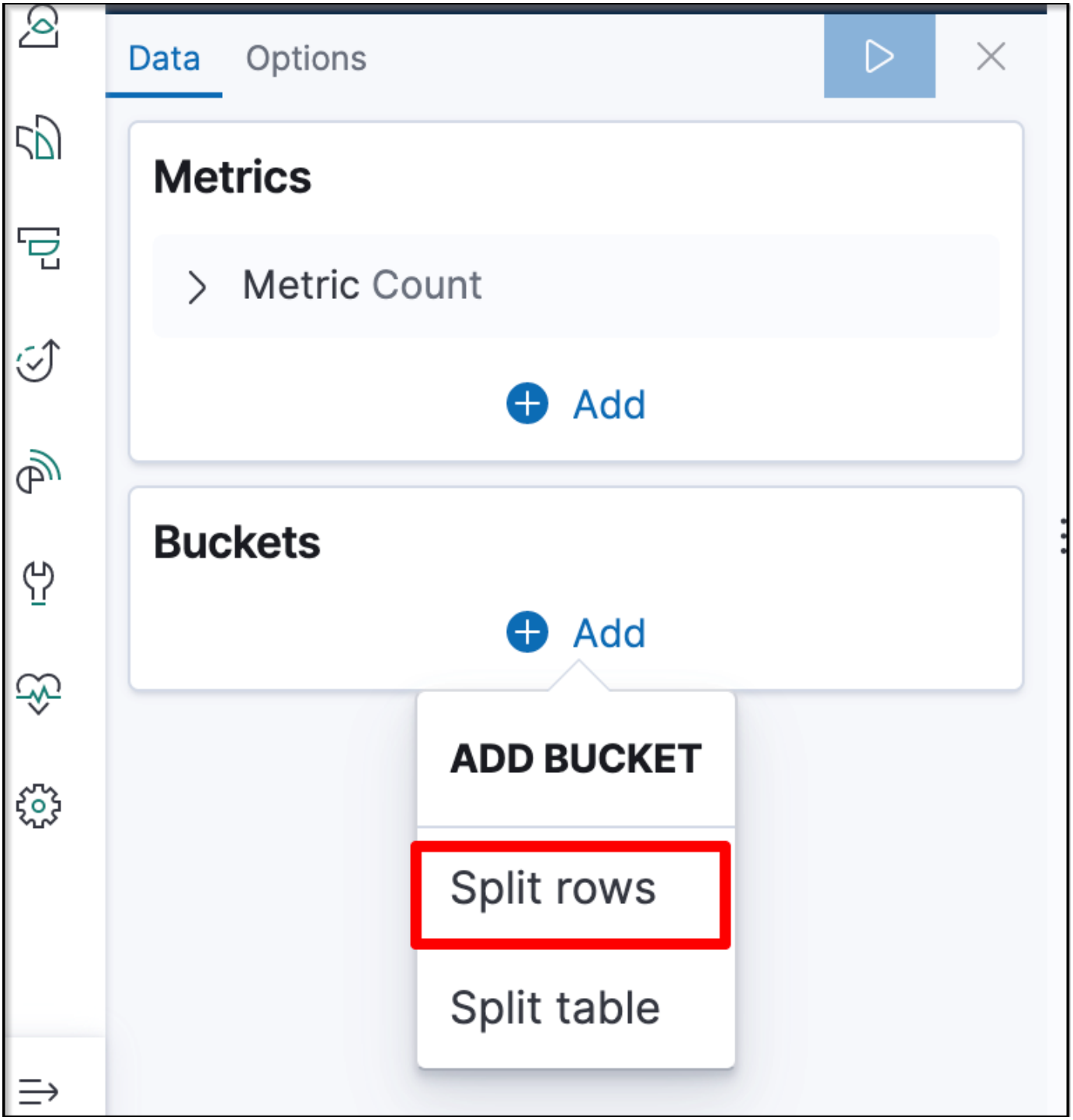
Setup Attack Visualize

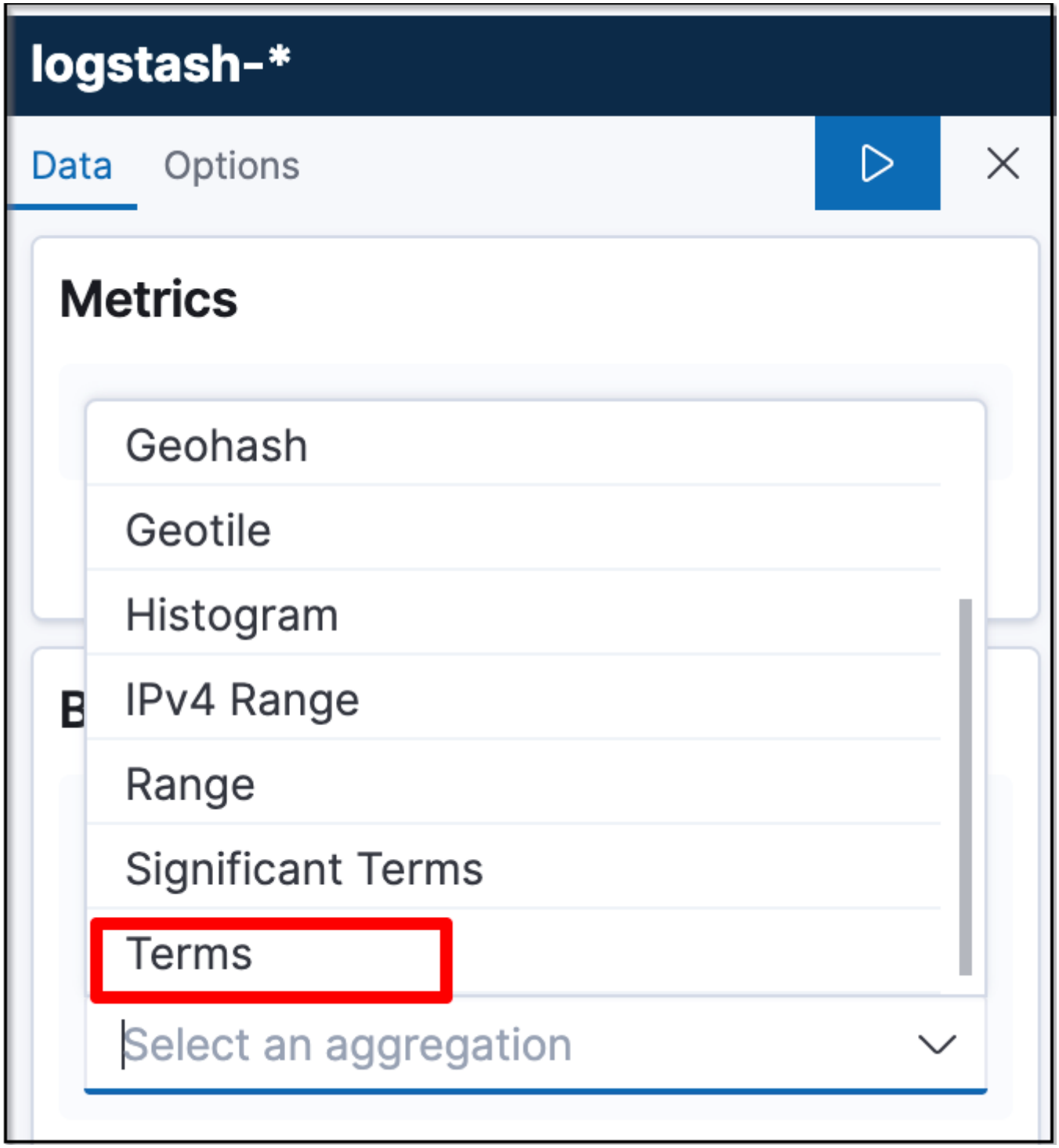


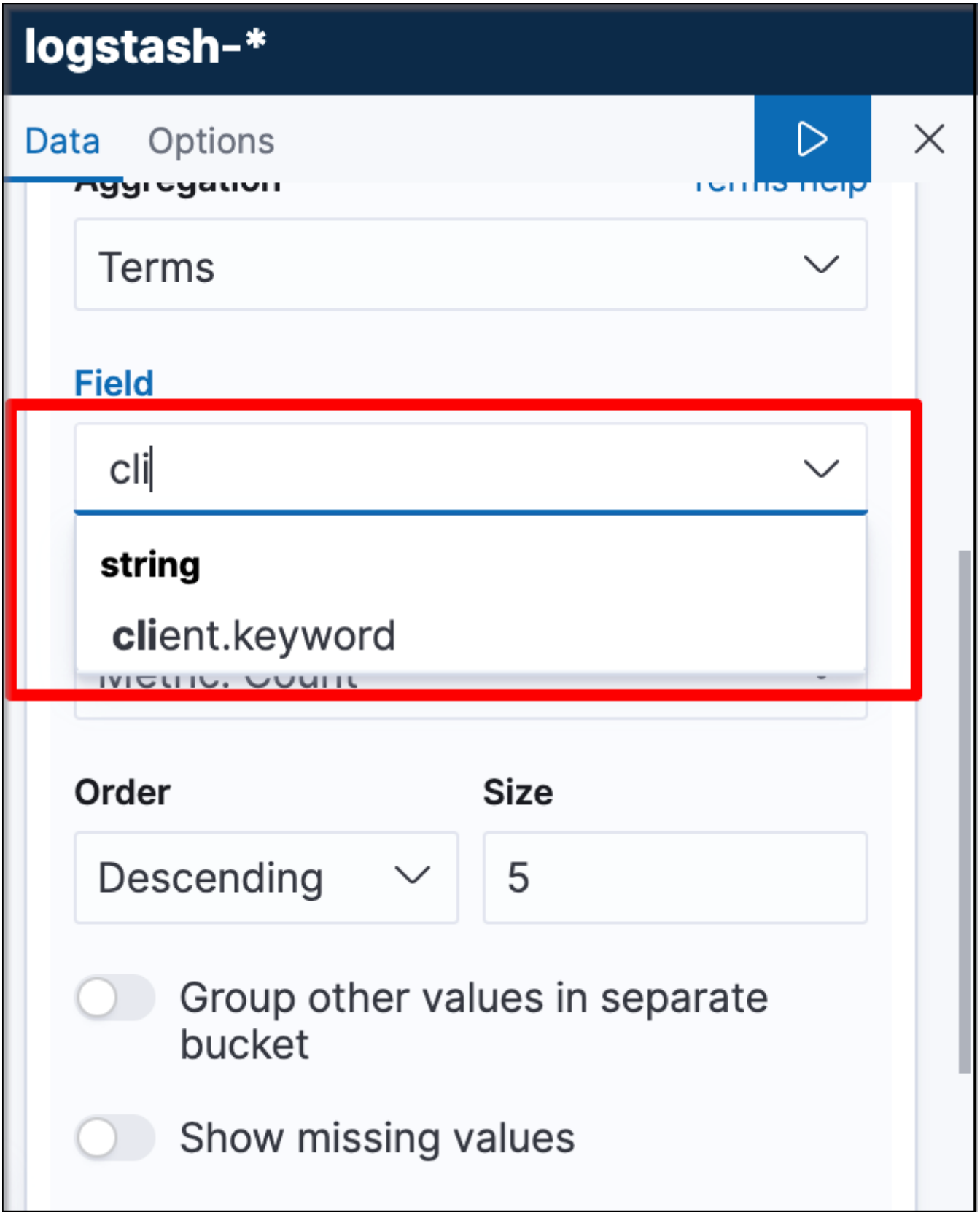


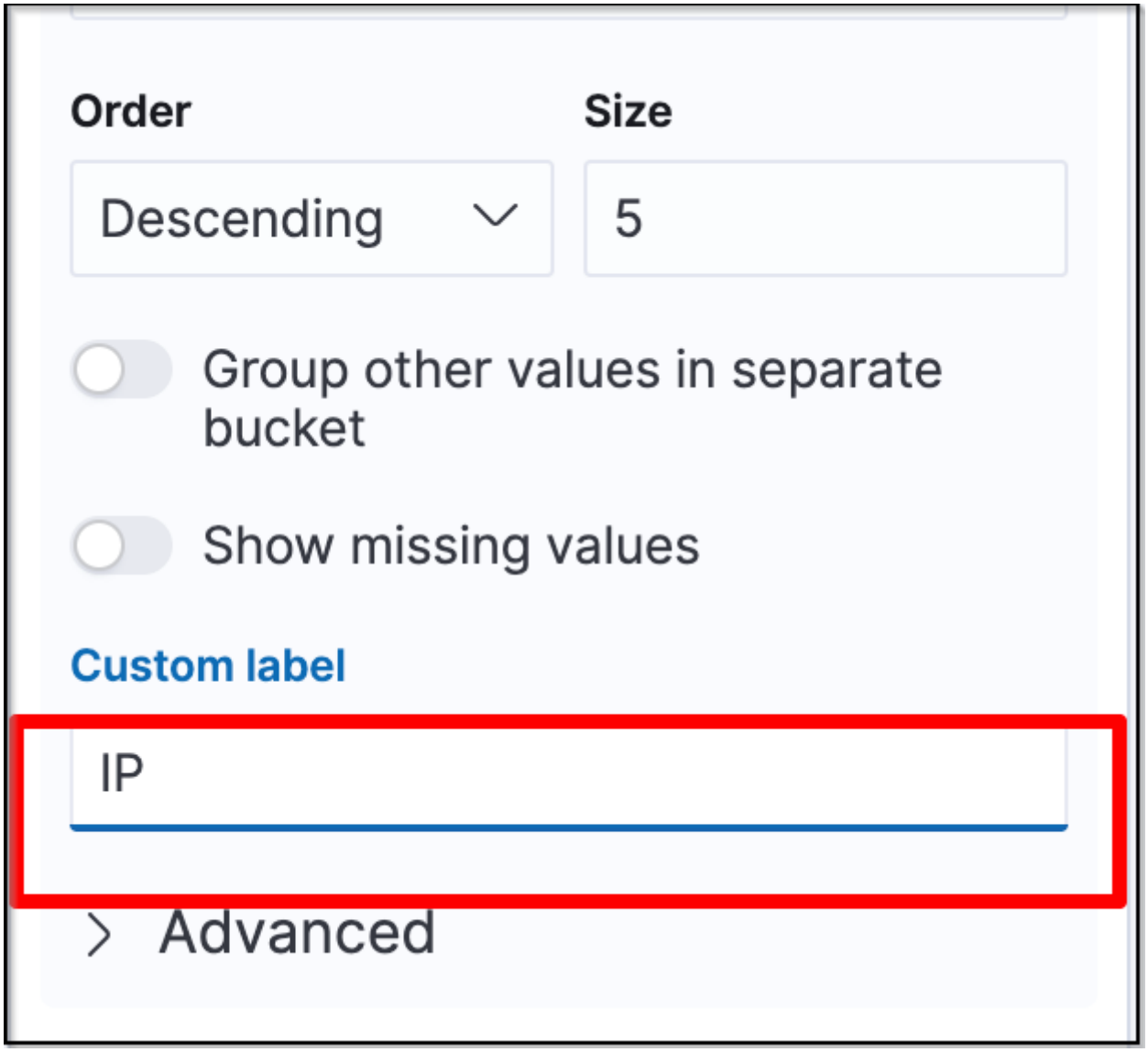

user0-kibana.appsecops.dev/app/kibana#/visualize?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,to:now))
Visualize
New Data Table / Choose a source
Search...
Sort
Types 2
logstash-*

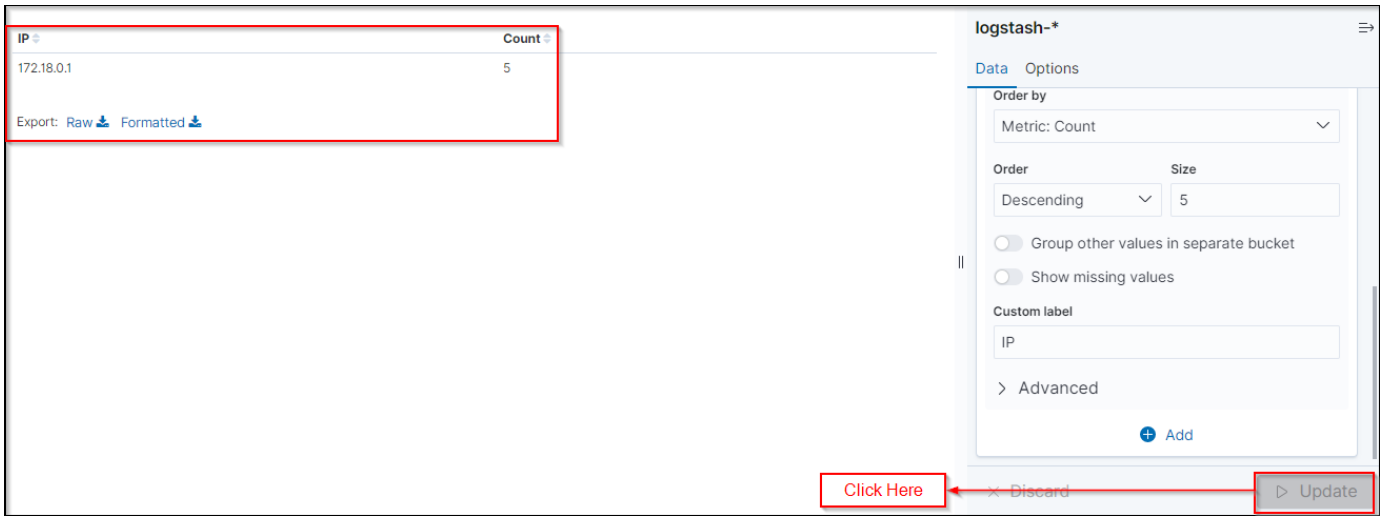

D
Visualize / Create
Help us improve the Elastic Stack
To learn about how usage data helps us manage and improve our products and services, see our Privacy Statement. To stop collection, disable usage data here.
Dismiss
Save Share Inspect Refresh
Search
KQL
Last 15 minutes
+ Add filter
logstash-*
Data Options
Metrics
Metric Count
Add
Buckets
Add
Count
3,355
Export: Raw Formatted

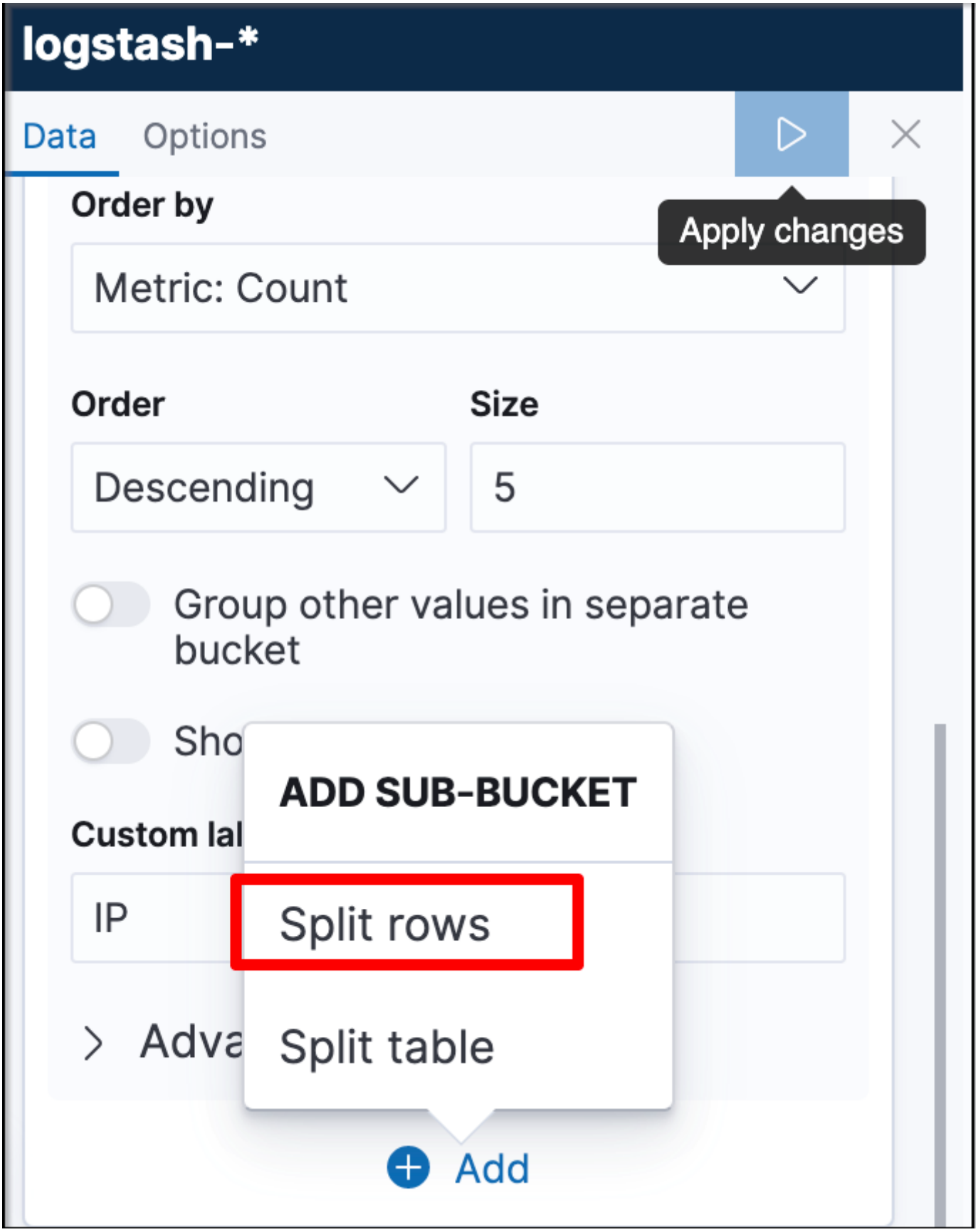

Data
Options
Metrics
Metric Count
Add
Buckets
Add
ADD BUCKET
Split rows
Split table

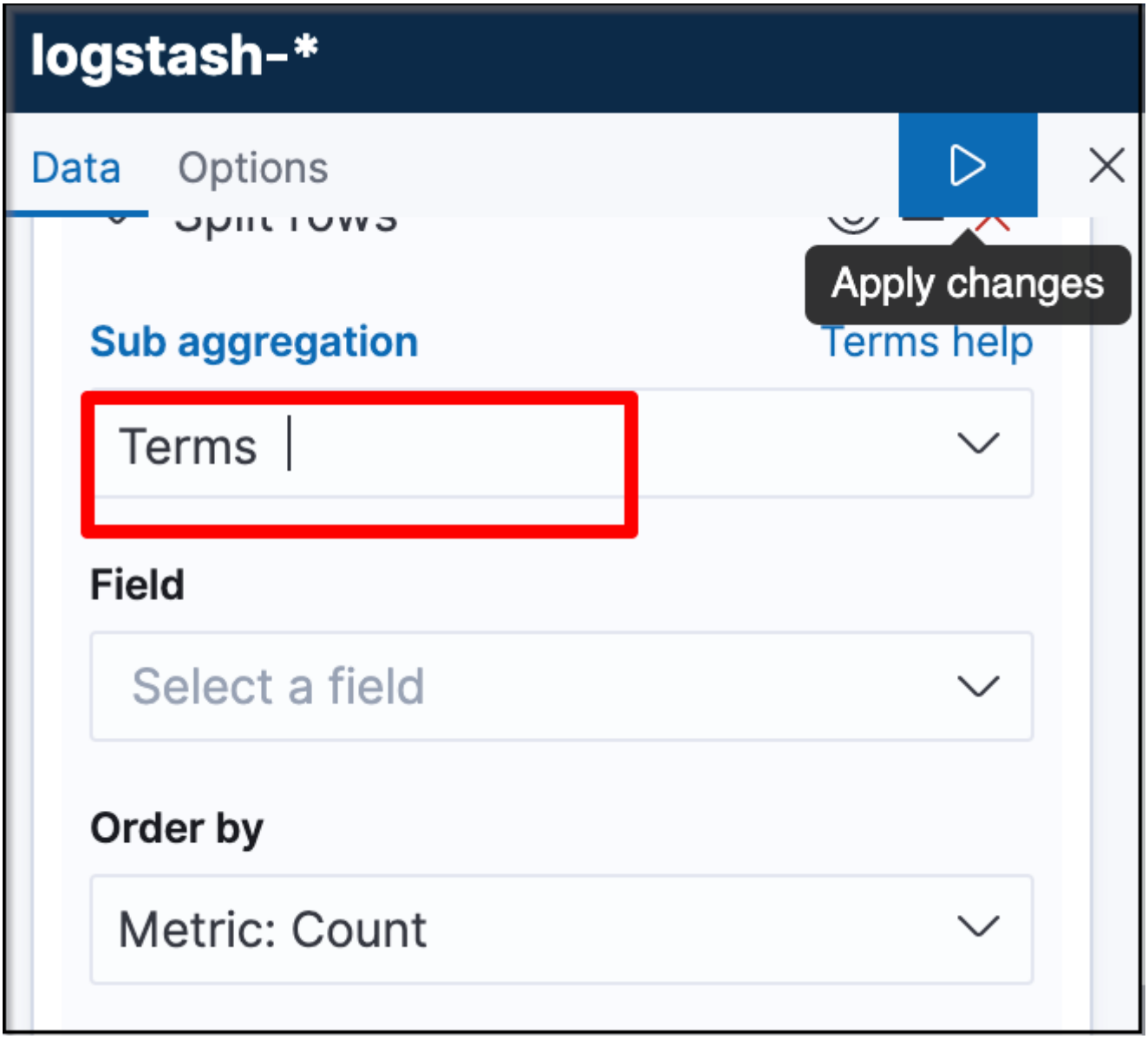

logstash-*
Data
Options
Metrics
Geohash
Geotile
Histogram
IPv4 Range
Range
Significant Terms
Terms
Select an aggregation

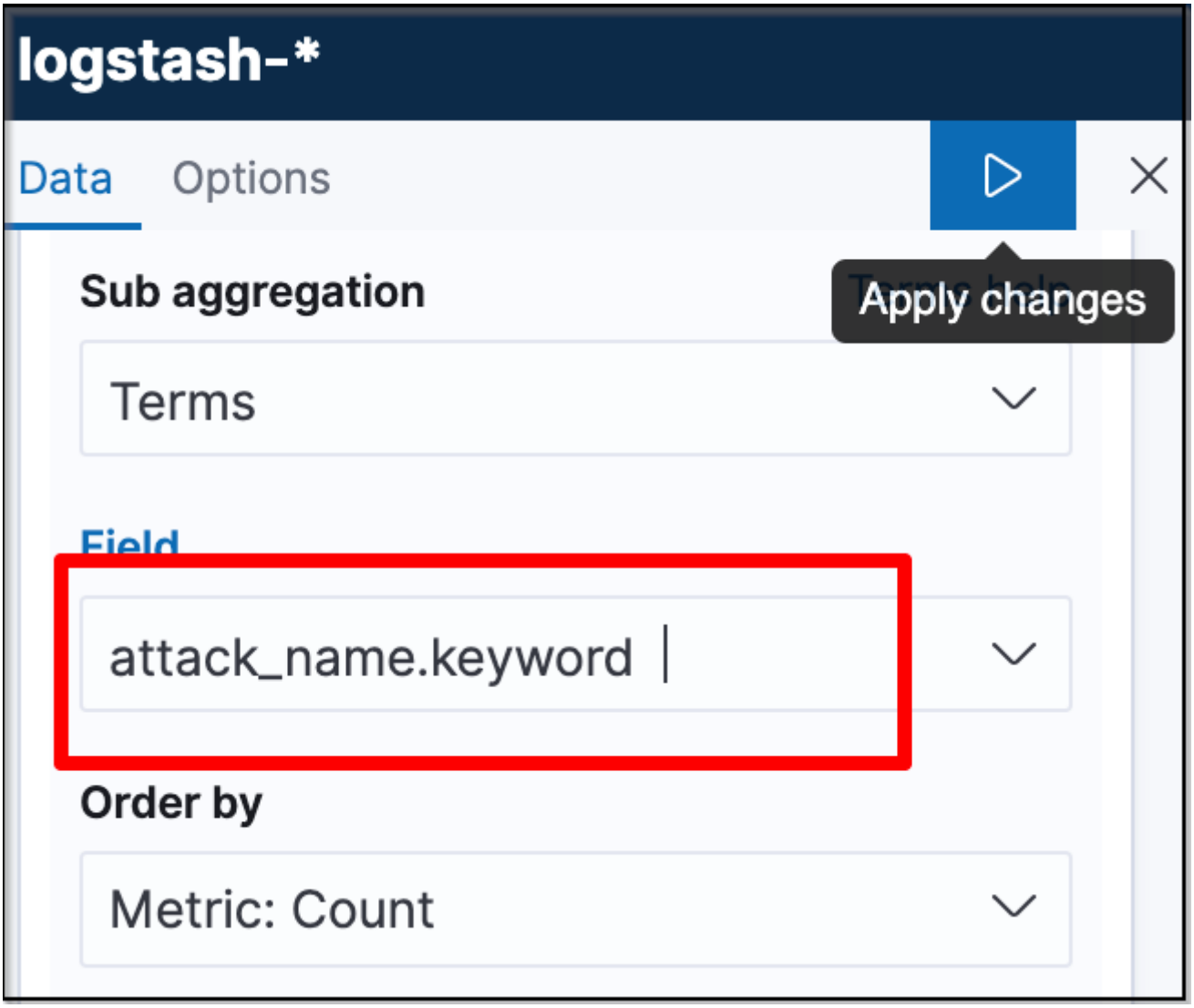

logstash-*
Data
Options
Terms
Field
cli
string
client.keyword
Order
Size
Descending
5
Group other values in separate bucket
Show missing values

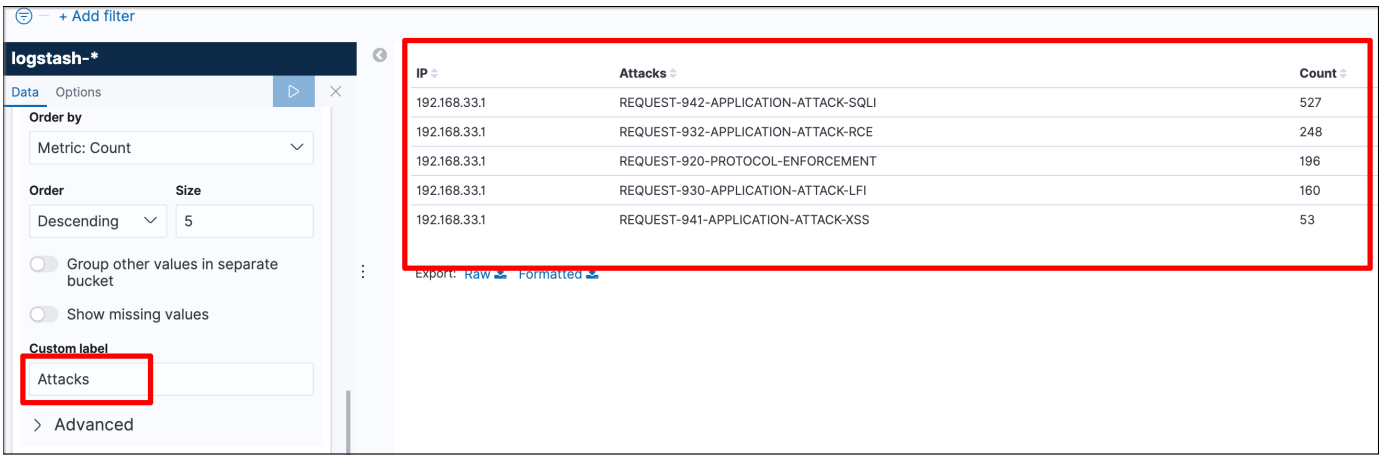

Order
Size
Descending
5
Group other values in separate bucket
Show missing values
Custom label
IP
Advanced

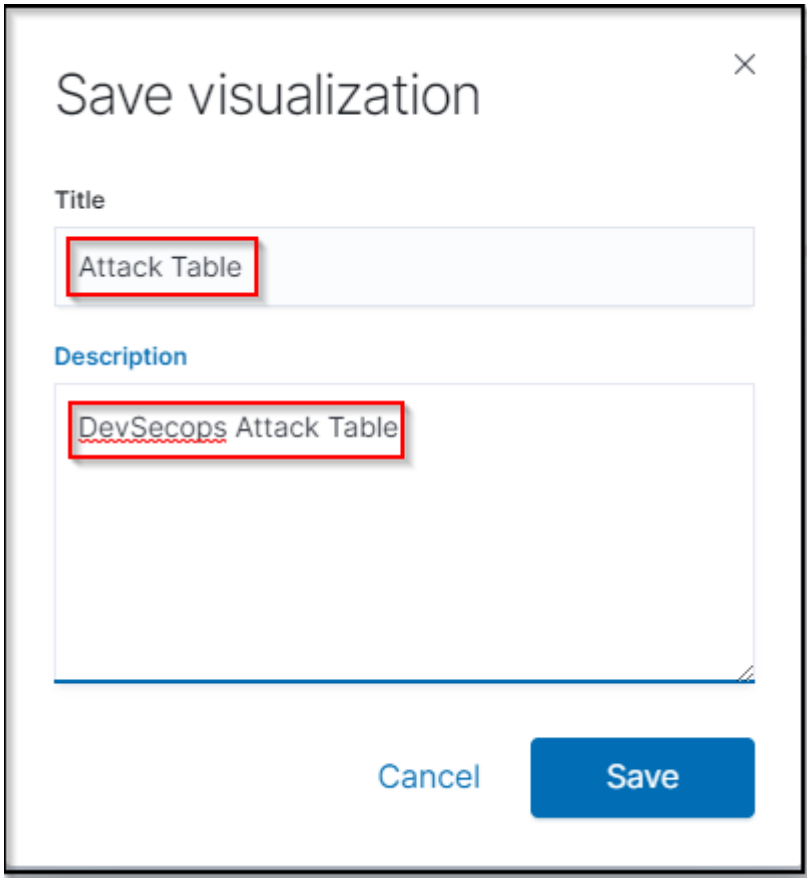

IP
Count
172.18.0.1
5
Export: Raw Formatted
logstash-*
Data Options
Order by
Metric: Count
Order
Size
Descending
5
Group other values in separate bucket
Show missing values
Custom label
IP
Advanced
Add
Click Here
Discard
Update

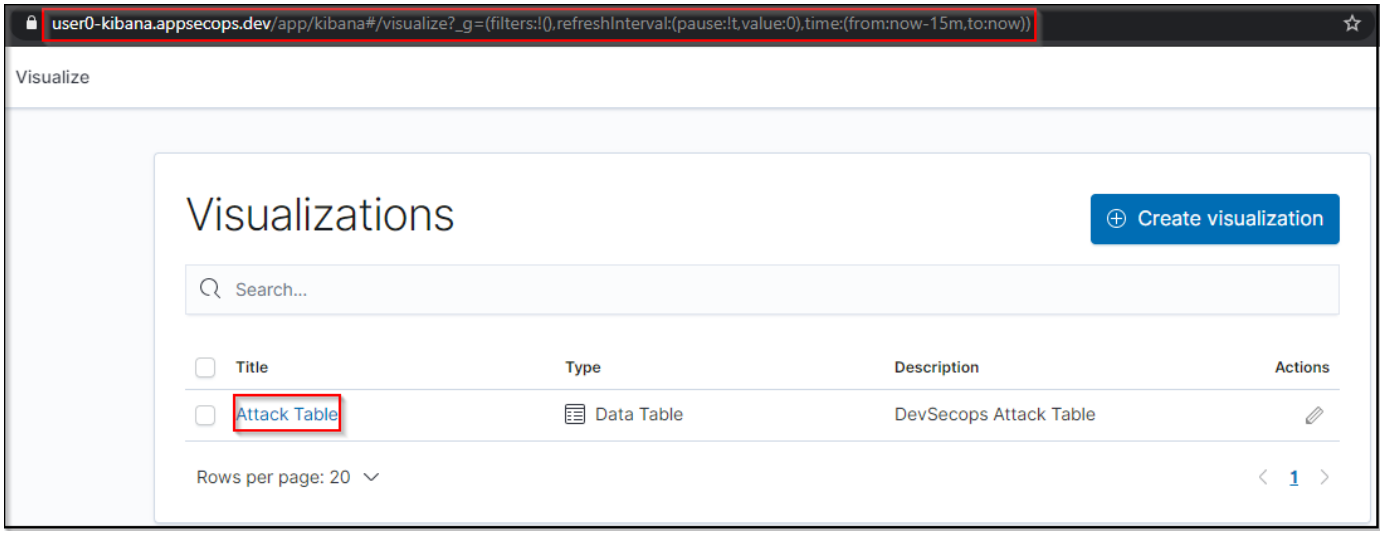

logstash-*
Data
Options
Apply changes
Order by
Metric: Count
Order
Size
Descending
5
Group other values in separate bucket
Sho
ADD SUB-BUCKET
Custom lal
IP
Split rows
Adva
Split table
Add

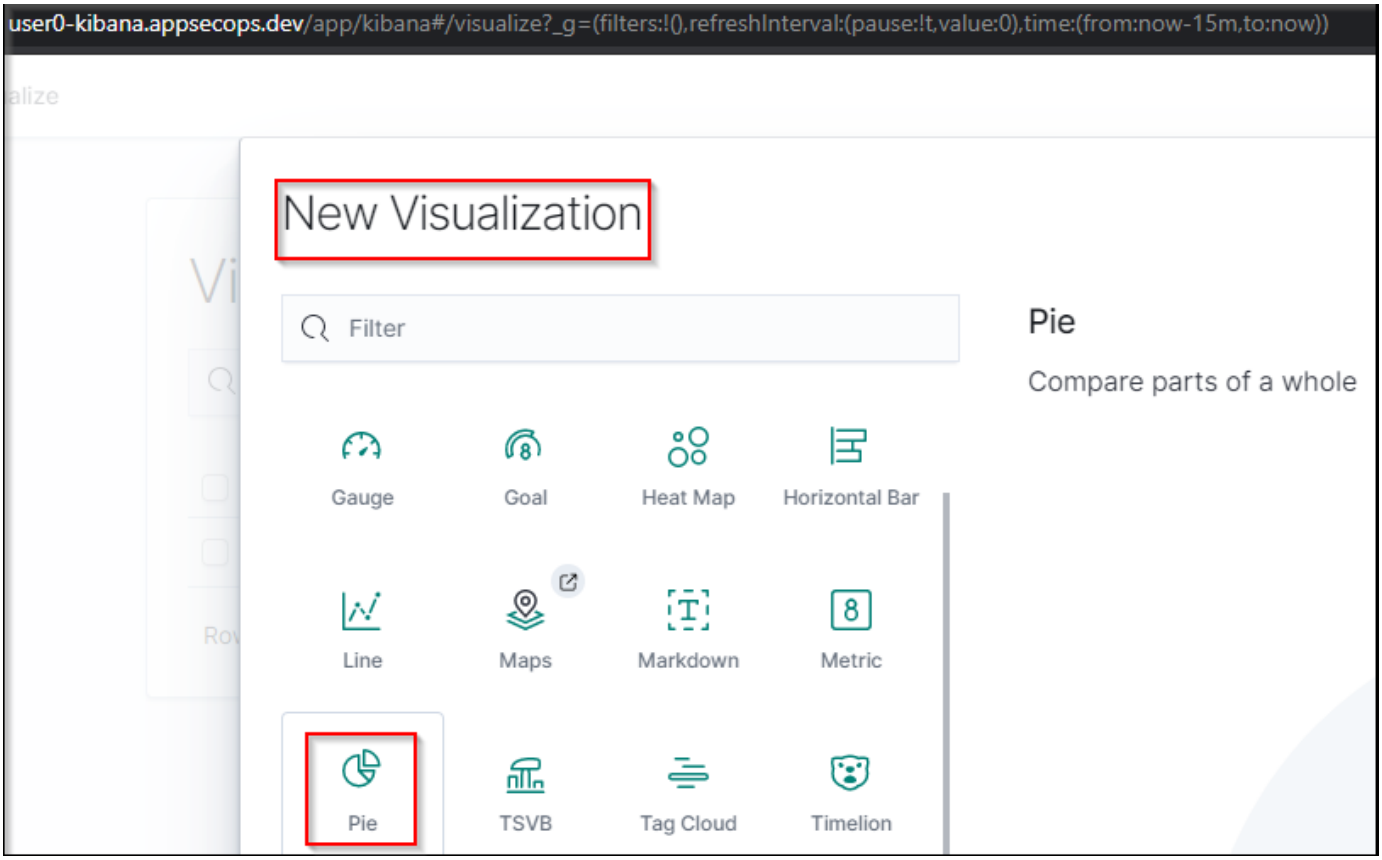

logstash-*
Data
Options
Apply changes
Sub aggregation
Terms help
Terms
Field
Select a field
Order by
Metric: Count

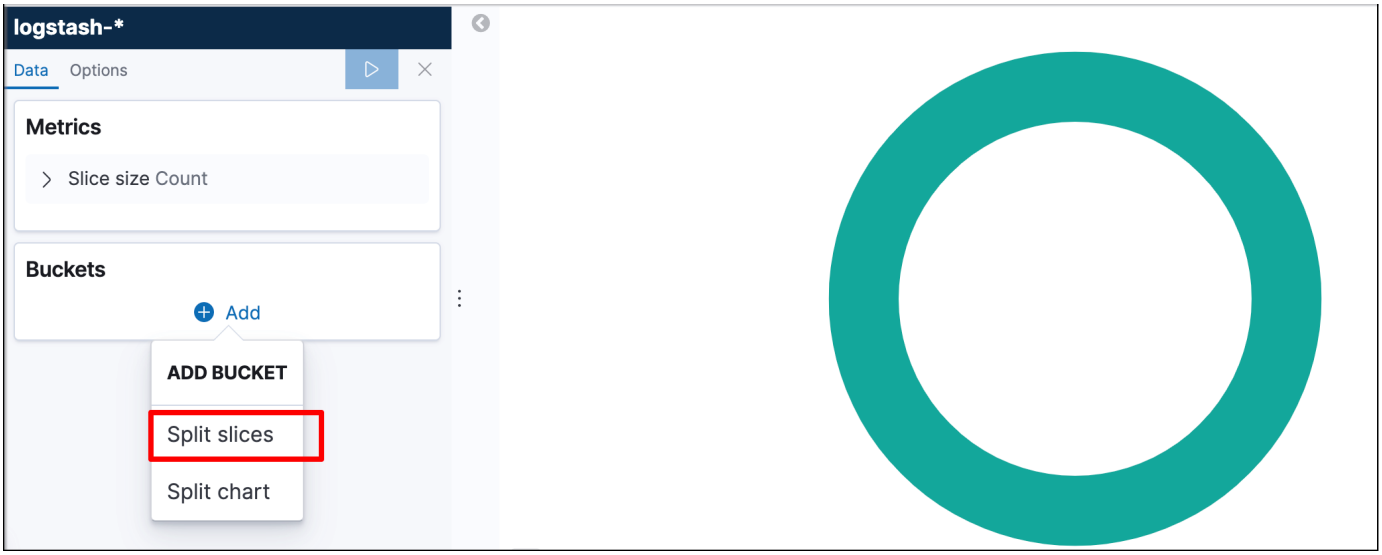

logstash-*
Data
Options
Apply changes
Sub aggregation
Terms
Field
attack_name.keyword
Order by
Metric: Count

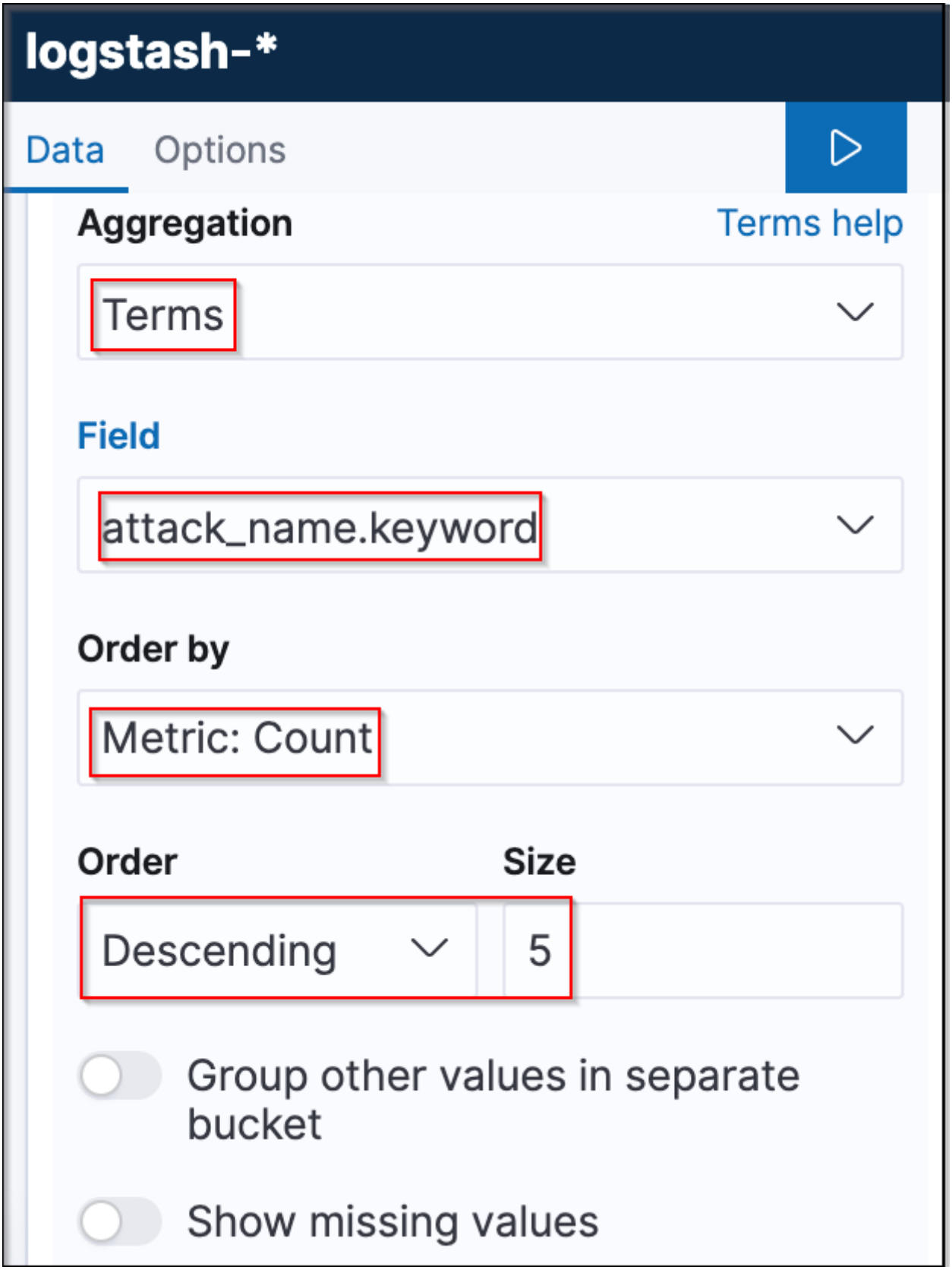

+ Add filter
logstash-*
Data
Options
Order by
Metric: Count
Order
Size
Descending
5
Group other values in separate bucket
Show missing values
Custom label
Attacks
Advanced
IP
Attacks
Count
192.168.33.1
REQUEST-942-APPLICATION-ATTACK-SQLI
527
192.168.33.1
REQUEST-932-APPLICATION-ATTACK-RCE
248
192.168.33.1
REQUEST-920-PROTOCOL-ENFORCEMENT
196
192.168.33.1
REQUEST-930-APPLICATION-ATTACK-LFI
160
192.168.33.1
REQUEST-941-APPLICATION-ATTACK-XSS
53
Export: Raw Formatted

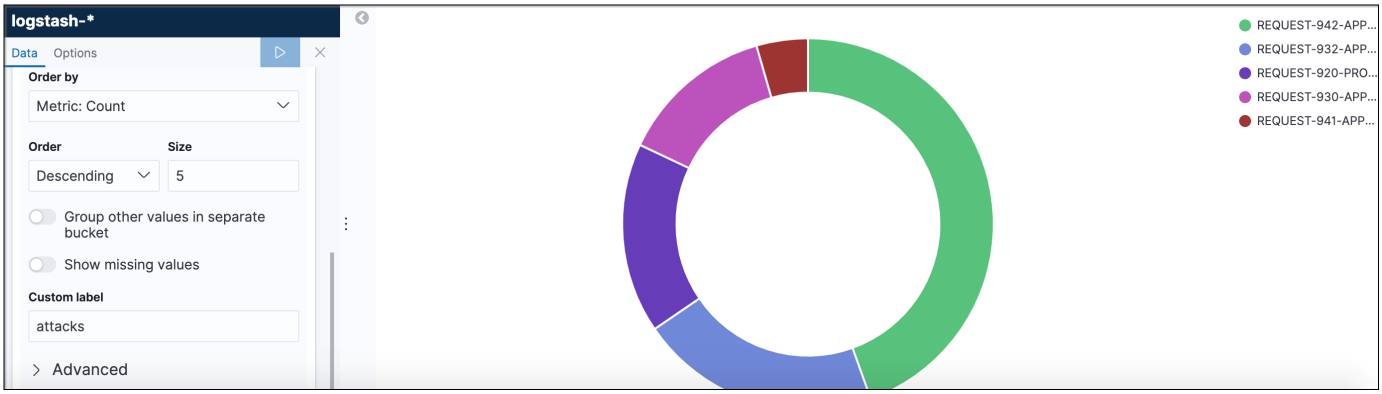

Save visualization
Title
Attack Table
Description
DevSecops Attack Table
Cancel
Save

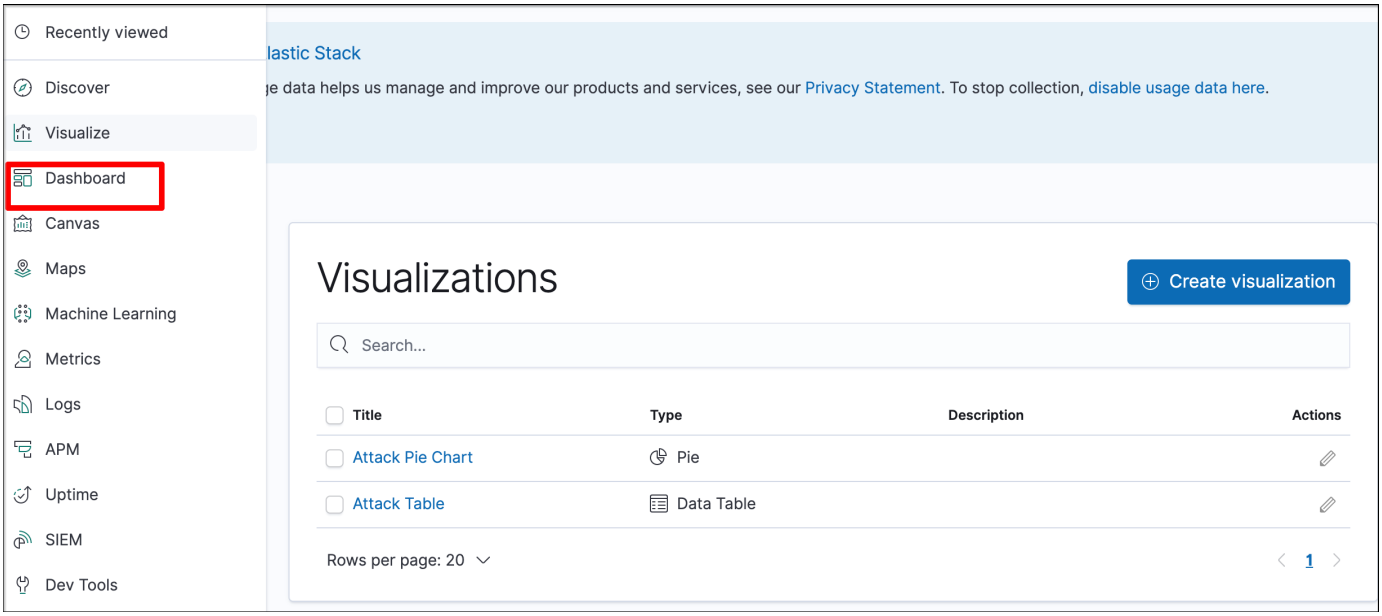

user0-kibana.appsecops.dev/app/kibana#/visualize?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,to:now))
Visualize
Visualizations
Create visualization
Search...
Title
Type
Description
Actions
Attack Table
Data Table
DevSecops Attack Table
Rows per page: 20
1

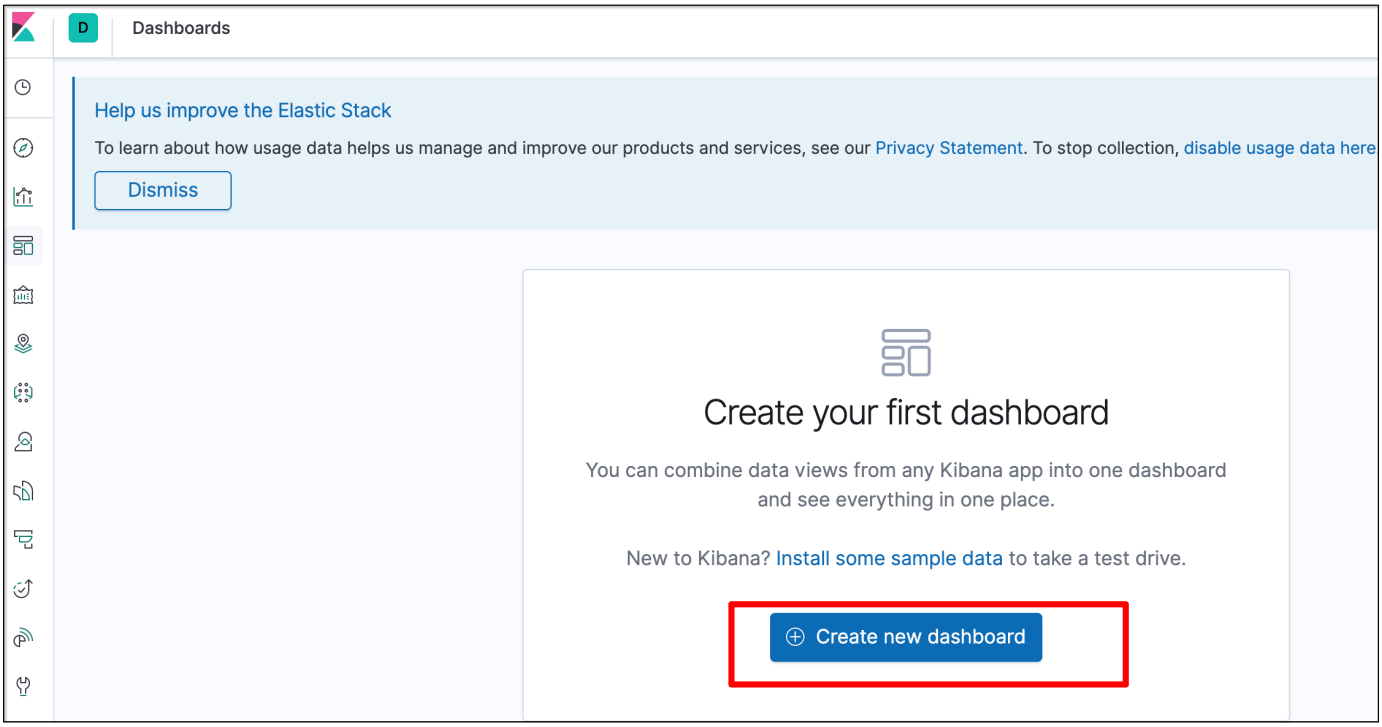

user0-kibana.appsecops.dev/app/kibana#/visualize?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,to:now))
New Visualization
Filter
Gauge
Goal
Heat Map
Horizontal Bar
Line
Maps
Markdown
Metric
Pie
TSVB
Tag Cloud
Timelion
Pie
Compare parts of a whole

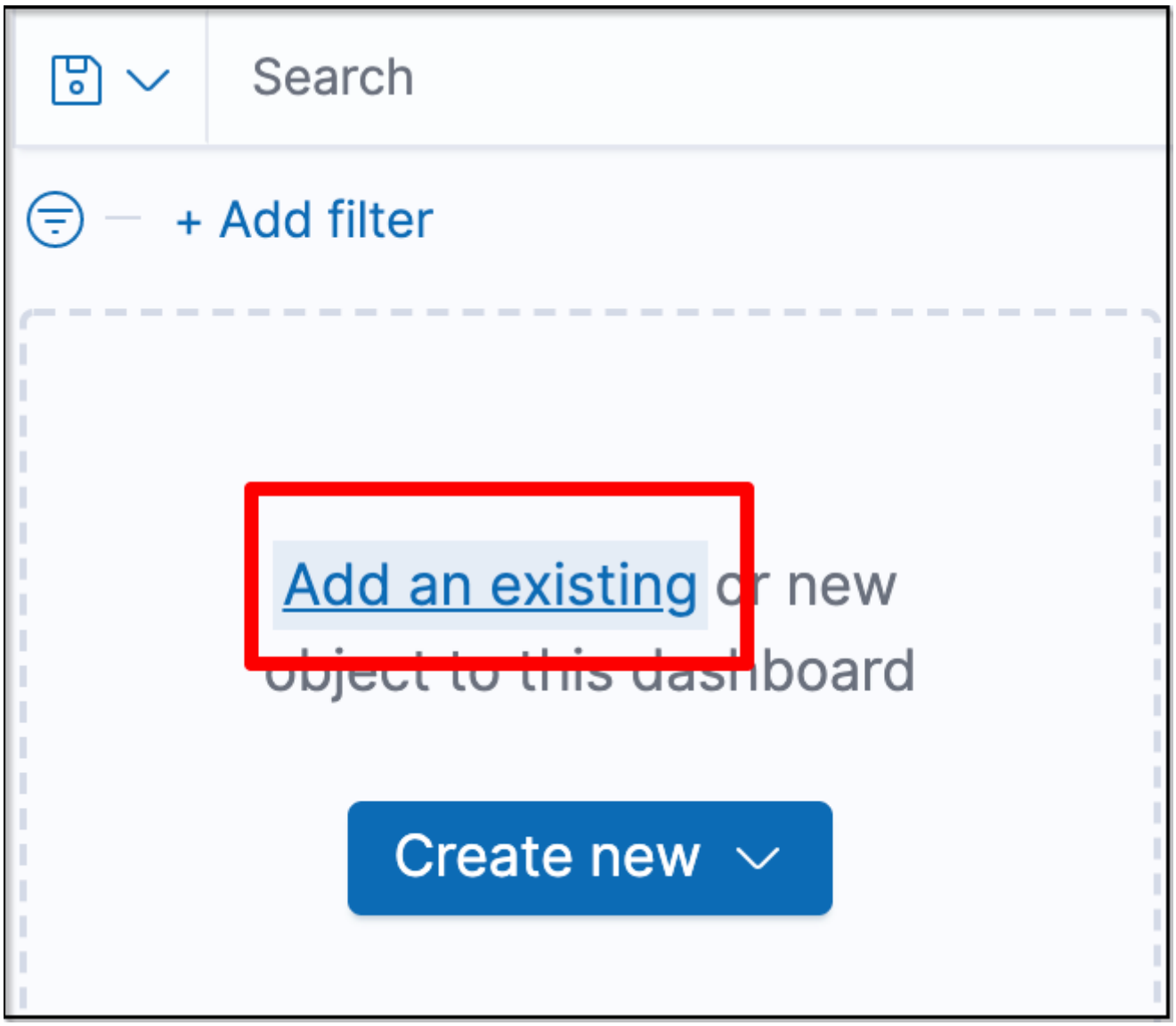

logstash-*
Data
Options
Metrics
Slice size Count
Buckets
Add
ADD BUCKET
Split slices
Split chart

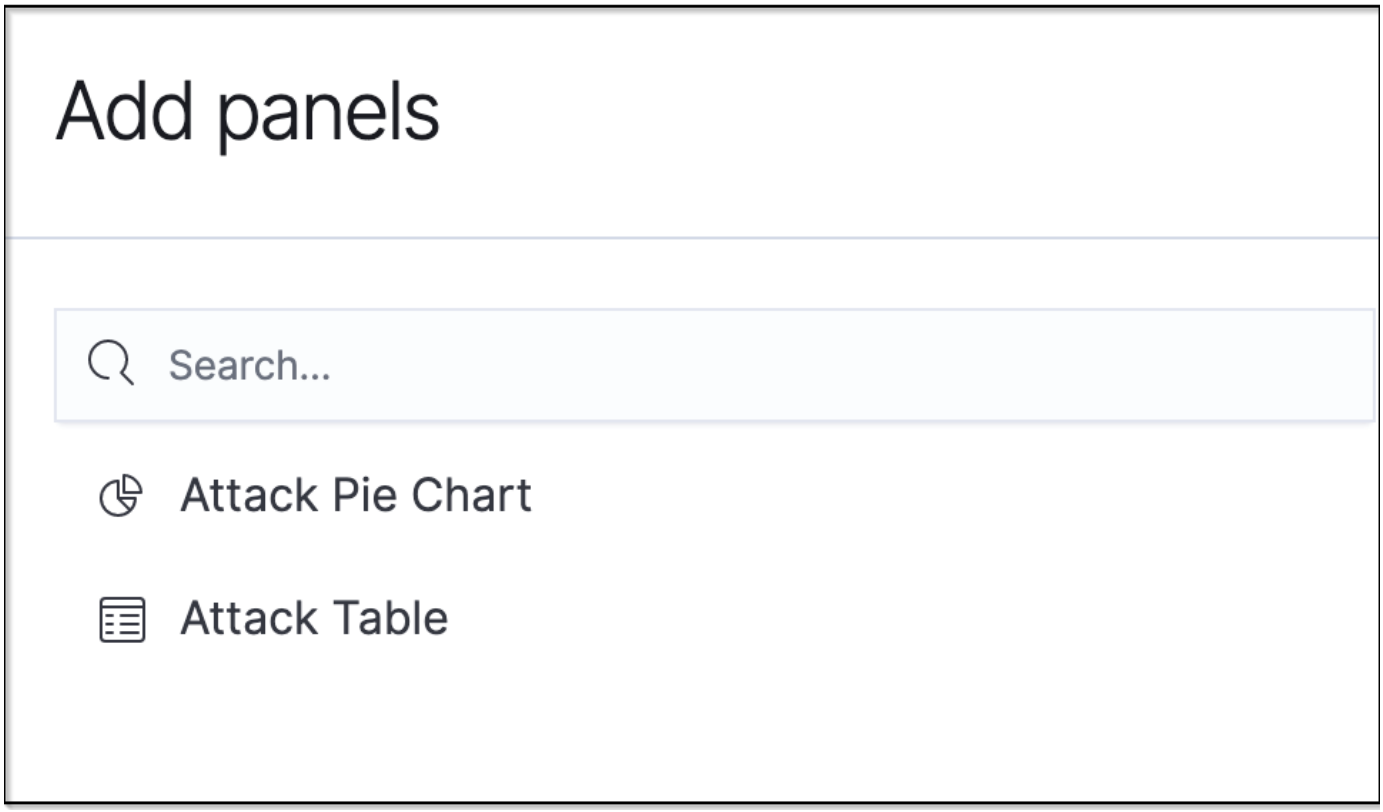

logstash-*
Data
Options
Aggregation
Terms help
Terms
Field
attack_name.keyword
Order by
Metric: Count
Order
Size
Descending
5
Group other values in separate bucket
Show missing values

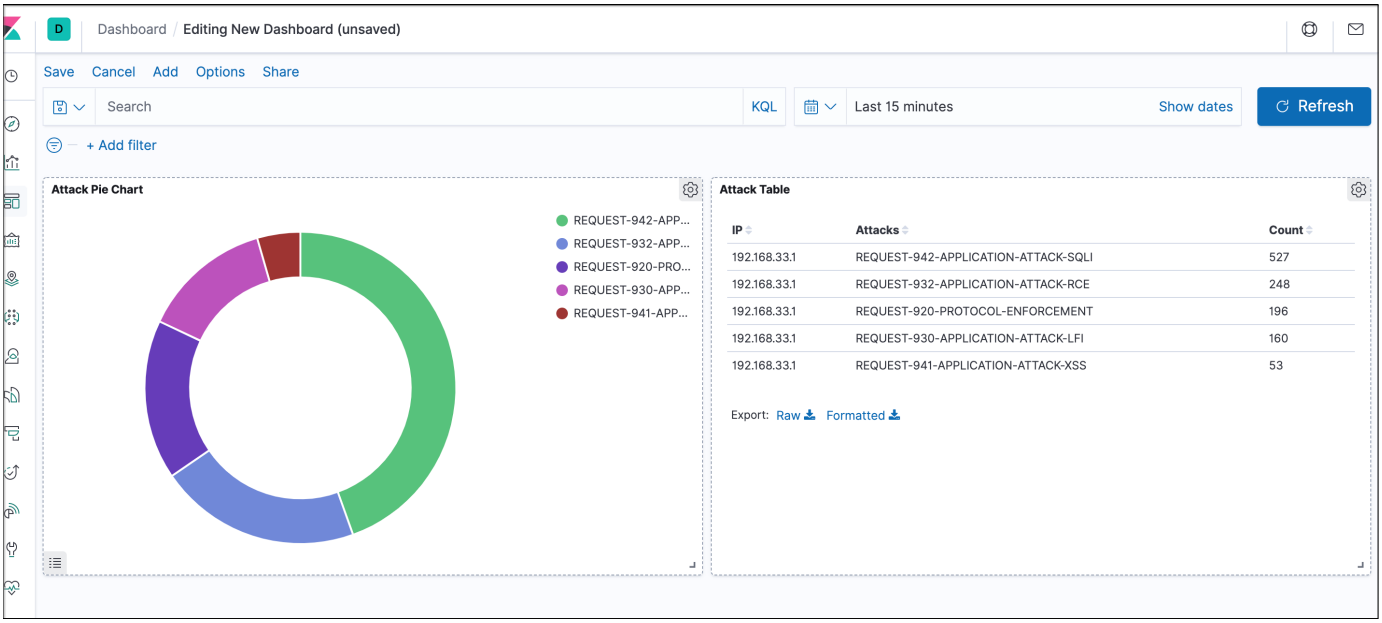

logstash-*
Data
Options
Order by
Metric: Count
Order
Size
Descending
5
Group other values in separate bucket
Show missing values
Custom label
attacks
Advanced
REQUEST-942-APP...
REQUEST-932-APP...
REQUEST-920-PRO...
REQUEST-930-APP...
REQUEST-941-APP...


Recently viewed
Discover
Visualize
Dashboard
Canvas
Maps
Machine Learning
Metrics
Logs
APM
Uptime
SIEM
Dev Tools
lastic Stack
e data helps us manage and improve our products and services, see our Privacy Statement. To stop collection, disable usage data here.
Visualizations
Create visualization
Search...
Title
Type
Description
Actions
Attack Pie Chart
Pie
Attack Table
Data Table
Rows per page: 20
1


D
Dashboards
Help us improve the Elastic Stack
To learn about how usage data helps us manage and improve our products and services, see our Privacy Statement. To stop collection, disable usage data here
Dismiss
Create your first dashboard
You can combine data views from any Kibana app into one dashboard and see everything in one place.
New to Kibana? Install some sample data to take a test drive.
Create new dashboard


Search
+ Add filter
Add an existing or new
object to this dashboard
Create new


Add panels
Search...
Attack Pie Chart
Attack Table


Dashboard / Editing New Dashboard (unsaved)
Save Cancel Add Options Share
Search
KQL
Last 15 minutes
Show dates
Refresh
+ Add filter
Attack Pie Chart
REQUEST-942-APP...
REQUEST-932-APP...
REQUEST-920-PRO...
REQUEST-930-APP...
REQUEST-941-APP...
Attack Table
IP
Attacks
Count
192.168.33.1 REQUEST-942-APPLICATION-ATTACK-SQLI 527
192.168.33.1 REQUEST-932-APPLICATION-ATTACK-RCE 248
192.168.33.1 REQUEST-920-PROTOCOL-ENFORCEMENT 196
192.168.33.1 REQUEST-930-APPLICATION-ATTACK-LFI 160
192.168.33.1 REQUEST-941-APPLICATION-ATTACK-XSS 53
Export: Raw Formatted

Save dashboard

Title

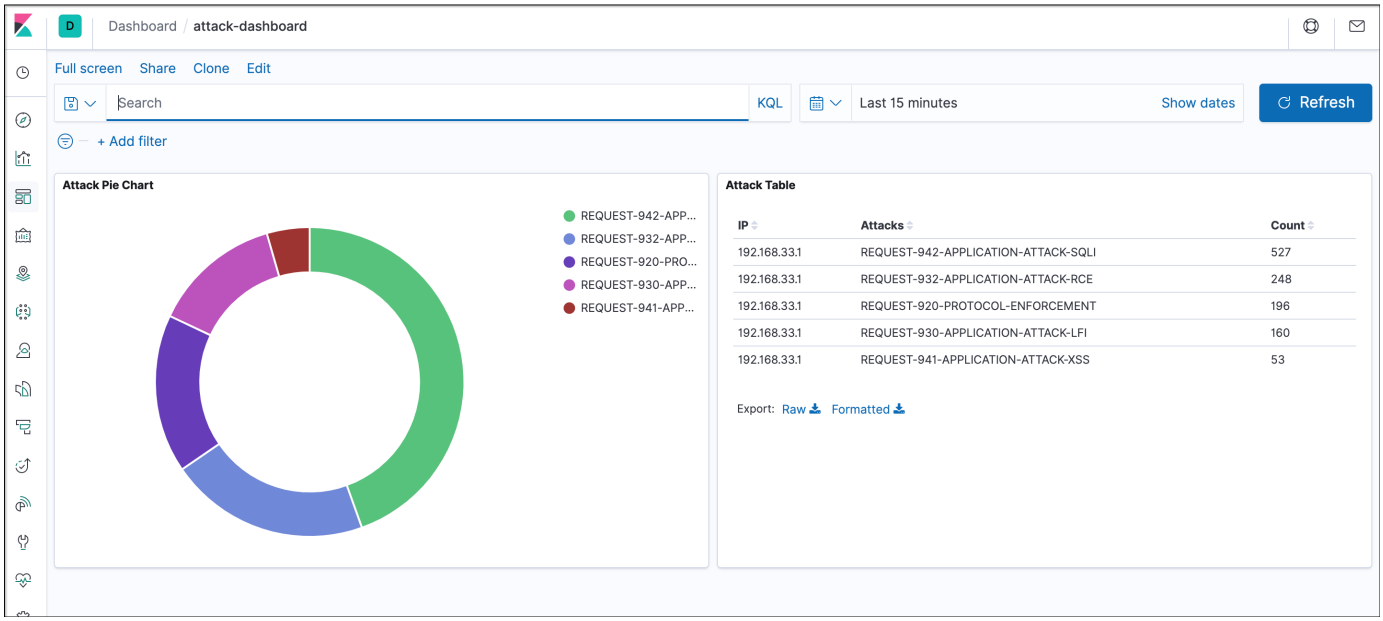
attack-dashboard

Description

Store time with dashboard

This changes the time filter to the currently selected time each time this dashboard is loaded.

Cancel Save


Dashboard / attack-dashboard
Full screen
Share
Clone
Edit
Search
KQL
Last 15 minutes
Show dates
Refresh
+ Add filter
Attack Pie Chart
REQUEST-942-APP...
REQUEST-932-APP...
REQUEST-920-PRO...
REQUEST-930-APP...
REQUEST-941-APP...
Attack Table
IP
Attacks
Count
192.168.33.1 REQUEST-942-APPLICATION-ATTACK-SQLI 527
192.168.33.1 REQUEST-932-APPLICATION-ATTACK-RCE 248
192.168.33.1 REQUEST-920-PROTOCOL-ENFORCEMENT 196
192.168.33.1 REQUEST-930-APPLICATION-ATTACK-LFI 160
192.168.33.1 REQUEST-941-APPLICATION-ATTACK-XSS 53
Export: Raw Formatted

Troubleshooting

```
scan id:
sleep: invalid time interval '20\r'
Try 'sleep --help' for more information.
Traceback (most recent call last):
  File "/var/lib/jenkins/jobs/devops/workspace/scripts/vuln_check_rule.py", line 197, in <module>
    for j in j_result:
TypeError: 'NoneType' object is not iterable
Traceback (most recent call last):
  File "/var/lib/jenkins/jobs/devops/workspace/scripts/vuln_check_rule.py", line 197, in <module>
    for j in j_result:
TypeError: 'NoneType' object is not iterable
/var/lib/jenkins/jobs/devops/workspace/scripts/dependency_check/dependency_check.sh: line 33: $'\r': command not found
/var/lib/jenkins/jobs/devops/workspace/scripts/dependency_check/dependency_check.sh: line 41: syntax error near unexpected token `fi
/var/lib/jenkins/jobs/devops/workspace/scripts/dependency_check/dependency_check.sh: line 41: `fi'
script returned exit code 2
```

- During pipeline build stages if you encounter any issues related to CRLF or syntax errors, Kindly use dos2unix tool to rectify the issues as shown below

dos2unix

Download dos2unix

```
apt-get update
```

```
apt-get install dos2unix
```

Run dos2unix

Navigate to scripts folder and run the below command for each tool as shown below :

```
dos2unix dependency_check.sh
```

Takeaway

About NotSoSecure

IT security specialist company providing cutting-edge IT security consultancy and training.

Penetration Testing

- Web Application Security Assessment
- Infrastructure Security Assessment
- Mobile Application Security Assessment
- Source Code Review
- IoT Security Assessment
- Red Team Exercises

Training

- Beginner Friendly
  - Hacking 101
  - Basic Infrastructure Hacking
  - Basic Web Hacking
- Advanced/Specialist Offensive Courses
  - Advanced Infrastructure Hacking
  - Advanced Web Hacking
  - Hacking and Securing Cloud
- Specialist Defensive Courses
  - Application Security for Developers
  - DevSecOps

© Copyright 2020 NotSoSecure Global Services Limited - A Claranet Group Company, All Rights Reserved.

© NotSoSecure Global Services Limited 2020, All Rights Reserved.