

LAB - REMOVE DEVELOPERS' SHAMEFUL SECRETS

OR SIMPLY REMOVE SHAMEFUL DEVELOPERS...

BY Fabian Lim



Join the conversation

#DevSecCon

REMOVE DEVELOPERS' SHAMEFUL SECRETS

Lab Slides

Fair Warning

Although minimal, my scripts are written mainly in MacOS.

For security concerns and compatibility, you might consider executing them in a Linux VM.

The permissions set on the services trust that developers (you) are trustworthy and responsible, with the ability to change certain settings.

Please exercise caution while making changes and limit those changes to your own environment in order to have a conducive learning environment.

Lab Audience and Objective

- Developers - who want to implement and fix the problem
- Managers - who wants the problem to be fixed but don't know how
- Compliance / Auditors - who wants to see how a problem can be fixed

...

Remove and rotate secrets

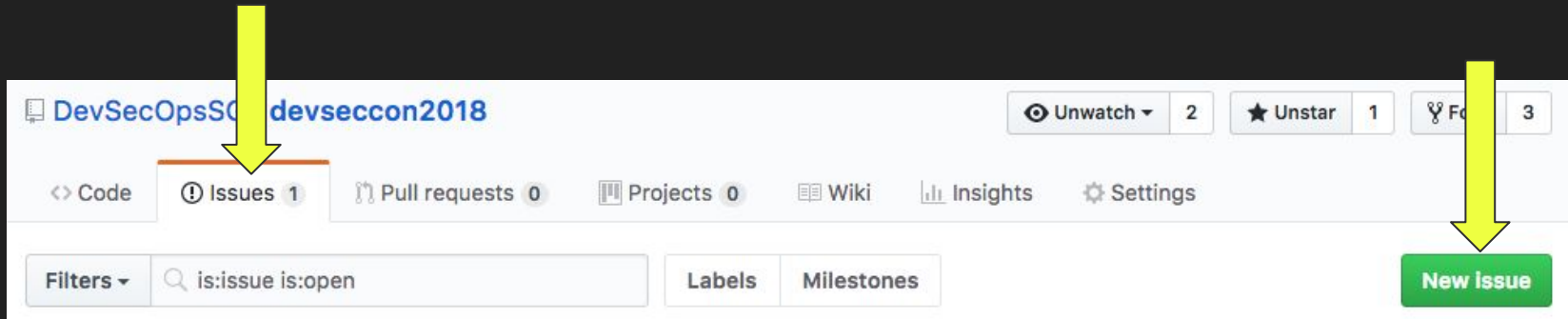
Use a secret management server

Integration with Jenkins

Lab 1 - Secrets Checked In

Setup / Login GitHub

- Login to your GitHub account
- Click on “Issues”
- Click “New issue” to create an issue in the devseccon2018 project so I know your username and I can invite you now!



Setup / Login GitHub

- Fill in some description about yourself and Click “Submit new issue”
- You will receive an email to join
- membership
- to <https://github.com/DevSecOpsSG>

DevSecOpsSG / devseccon2018

Unwatch 2

Code Issues 1 Pull requests 0 Projects 0 Wiki Insights Settings

invite me

Write Preview

Request

Add me to DevSecOpsSG Github Organizations

A bit about yourself

Name :

Email :

Organization :

Others (Projects/Portfolio) :

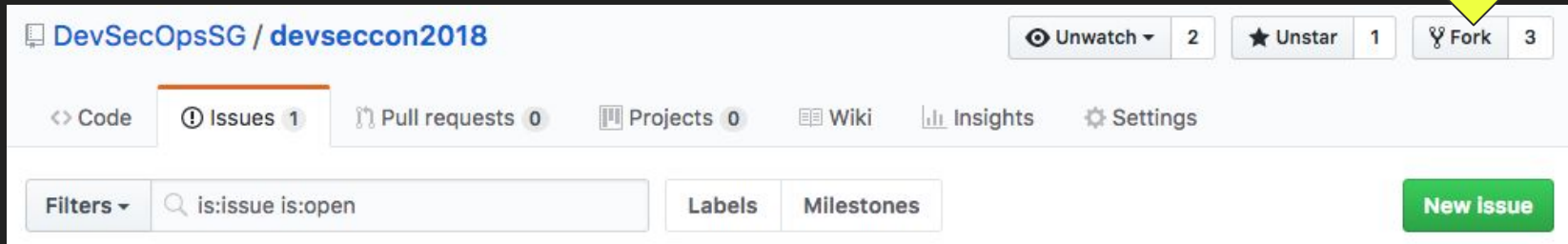
Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Submit new issue

Setup / Login GitHub

- Go to <https://github.com/DevSecOpsSG/devseccon2018.git>
- Click “Fork”
- You should be redirected to <https://github.com/<username>/devseccon2018.git>
- This is your fork (copy) of the repository



Setup / Login GitHub

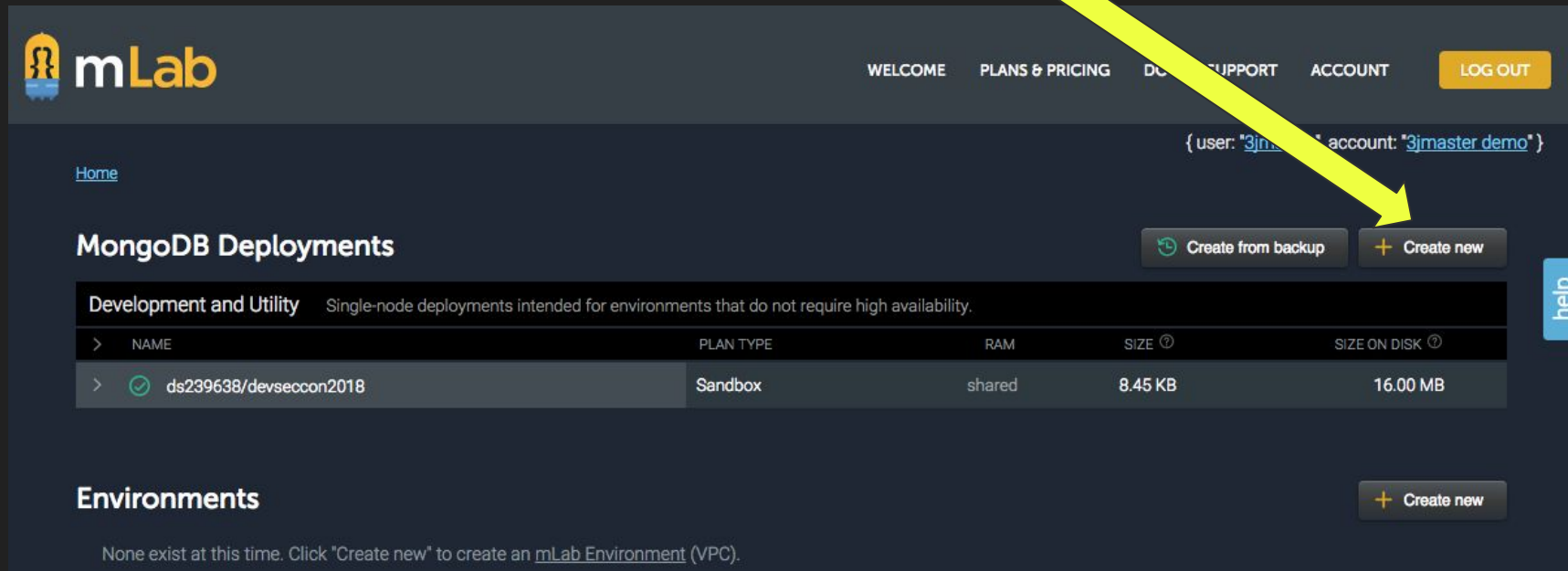
- Open a terminal and run to clone code to your local machine:

git clone <https://github.com/><username>/devseccon2018.git

- You now have a local version of the code

Setup mlab (MongoDB)

- Sign up an account at <https://mlab.com>
- Click “Create new” for MongoDB Deployments



The screenshot shows the mLab website interface. At the top, the mLab logo is on the left, and navigation links for WELCOME, PLANS & PRICING, DOCUMENTATION, SUPPORT, and ACCOUNT are on the right, along with a LOG OUT button. Below the navigation bar, the user's session is displayed as { user: "3jmaster demo", account: "3jmaster demo" }. The main section is titled "MongoDB Deployments" and includes a "Create from backup" button and a "+ Create new" button. A yellow arrow points from the second bullet point in the list above to the "+ Create new" button. Below this is a table of existing deployments under the "Development and Utility" plan type. The table has columns for NAME, PLAN TYPE, RAM, SIZE, and SIZE ON DISK. One deployment is listed: ds239638/devsecon2018, which is a Sandbox plan with shared RAM, 8.45 KB size, and 16.00 MB size on disk. At the bottom, there is an "Environments" section with a "+ Create new" button and a message stating that no environments exist at this time.

Home


WELCOME PLANS & PRICING DOCUMENTATION SUPPORT ACCOUNT LOG OUT

{ user: "3jmaster demo", account: "3jmaster demo" }

MongoDB Deployments

Create from backup + Create new

Development and Utility Single-node deployments intended for environments that do not require high availability.

>	NAME	PLAN TYPE	RAM	SIZE ⓘ	SIZE ON DISK ⓘ
>	 ds239638/devsecon2018	Sandbox	shared	8.45 KB	16.00 MB

Environments

+ Create new

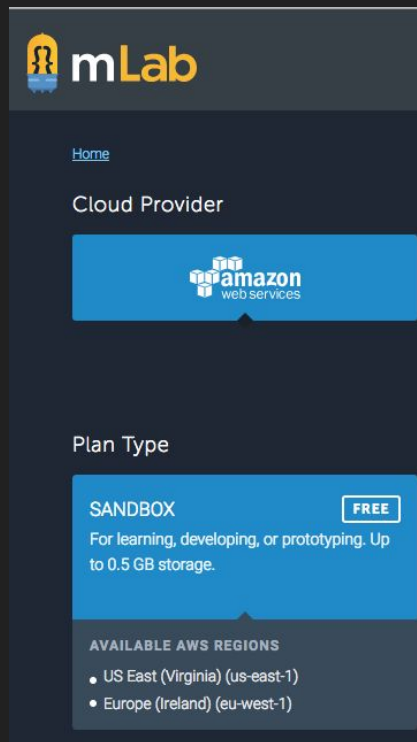
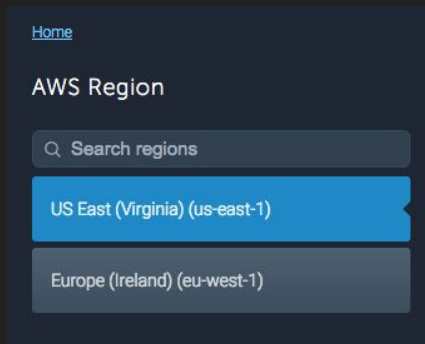
None exist at this time. Click "Create new" to create an [mLab Environment](#) (VPC).

help

Setup mlab (MongoDB)

- Choose any Cloud Provider (doesn't matter)
- Choose Plan Type SANDBOX (FREE)
- Click "Continue"

- Select a region (doesn't matter)
- Click "Continue"



Setup mlab (MongoDB)

- Give your database a cool name (doesn't matter)
- Click "Continue"

[Home](#)

Final Details

MONGODB VERSION 3.4.13 (MMAPv1)

DATABASE NAME








coolname

Setup mlab (MongoDB)

- Check your final order
- Click “SUBMIT ORDER”

[Home](#)

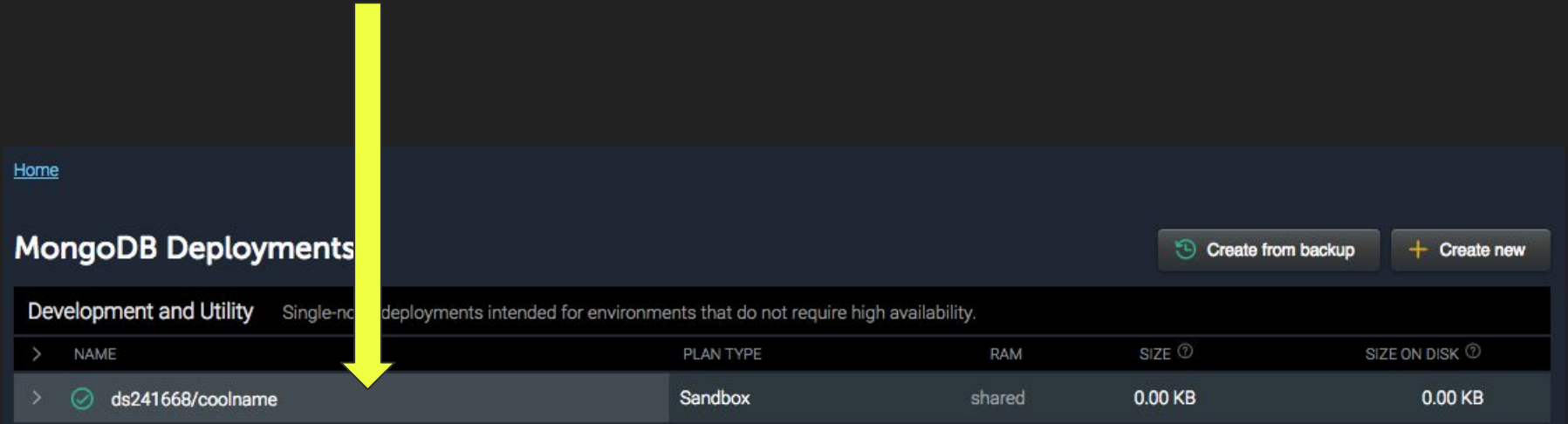
Order Confirmation

PLAN TYPE	Sandbox	
CLOUD PROVIDER	Amazon Web Services	
REGION	US East (Virginia) (us-east-1)	
PLAN LINE & SIZE	Sandbox	FREE 
	 STORAGE 0.5 GB	
MONGODB VERSION	3.4.13 (MMAPv1)	
DATABASE NAME	coolname	
Total Price		FREE

© 2018 ObjectLabs Corporation [Terms of Service](#) [Privacy Policy](#) [cancel](#) [BACK](#) [SUBMIT ORDER](#)

Setup mlab (MongoDB)

- Click on the new database



The screenshot shows the MongoDB Deployments interface. At the top, there's a 'Home' link and a 'MongoDB Deployments' header. To the right of the header are two buttons: 'Create from backup' and 'Create new'. Below the header, there's a section titled 'Development and Utility' with a description: 'Single-node deployments intended for environments that do not require high availability.' Below this is a table with columns: NAME, PLAN TYPE, RAM, SIZE, and SIZE ON DISK. A yellow arrow points to the first row of the table, which has the name 'ds241668/coolname'.

>	NAME	PLAN TYPE	RAM	SIZE [?]	SIZE ON DISK [?]
>	✓ ds241668/coolname	Sandbox	shared	0.00 KB	0.00 KB

Setup mlab (MongoDB)

- Click on “Users”
- Click on “Add database user”

The screenshot shows the MongoDB mlab web interface. At the top, there's a 'Home' link and the database name 'coolname'. A 'Delete database' button is in the top right. Below this, there's a section for connecting to the database, showing the mongo shell command and the standard MongoDB URI. A yellow arrow points from the 'Users' tab in the bottom navigation bar to the 'Add database user' button in the 'Database Users' section. The 'Database Users' section shows '[None at this time]'. There are also two warning messages: one about sandbox databases not being suitable for production, and another about requiring a database user to connect.

Home
Database: coolname Delete database

To connect using the mongo shell:
`% mongo ds241668.mlab.com:41668/coolname -u <dbuser> -p <dbpassword>`

To connect using a driver via the standard MongoDB URI ([what's this?](#)):
`mongodb://<dbuser>:<dbpassword>@ds241668.mlab.com:41668/coolname`

mongod version: 3.4.13 (MMAPv1)

⚠ Sandbox databases do not have redundancy and therefore are not suitable for production. Read our documentation on [how to upgrade](#).

⚠ A database user is required to connect to this database. To create one now, visit the 'Users' tab and click the 'Add database user' button.

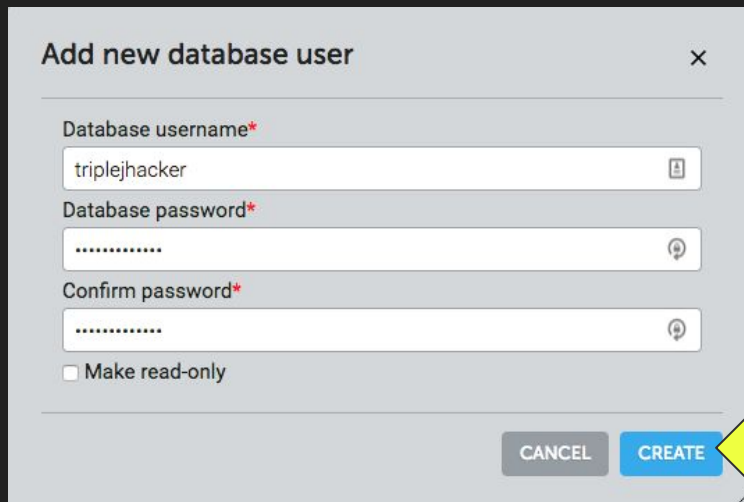
Collections **Users** Stats Backups Tools

Database Users Add database user

[None at this time]

Setup mlab (MongoDB)

- Enter a username and password (don't use anything personal or sensitive)
 - To avoid syntax error in the later steps, do not use the “@” symbol at all; you can use it but you must [encode the character](#) in the script later
- Leave “Make read-only” box unchecked
- Click “Create”



Add new database user [X]

Database username*
triplejhacker

Database password*
.....

Confirm password*
.....

☐ Make read-only

CANCEL CREATE

Setup mlab (MongoDB)

From the database page, construct your mlab mongodb instance URL (this contains secret) replacing <dbuser> and <dbpassword> that you previously entered.

[Home](#)

Database: coolname

To connect using the mongo shell:

```
% mongo ds241668.mlab.com:41668/coolname -u <dbuser> -p <dbpassword>
```

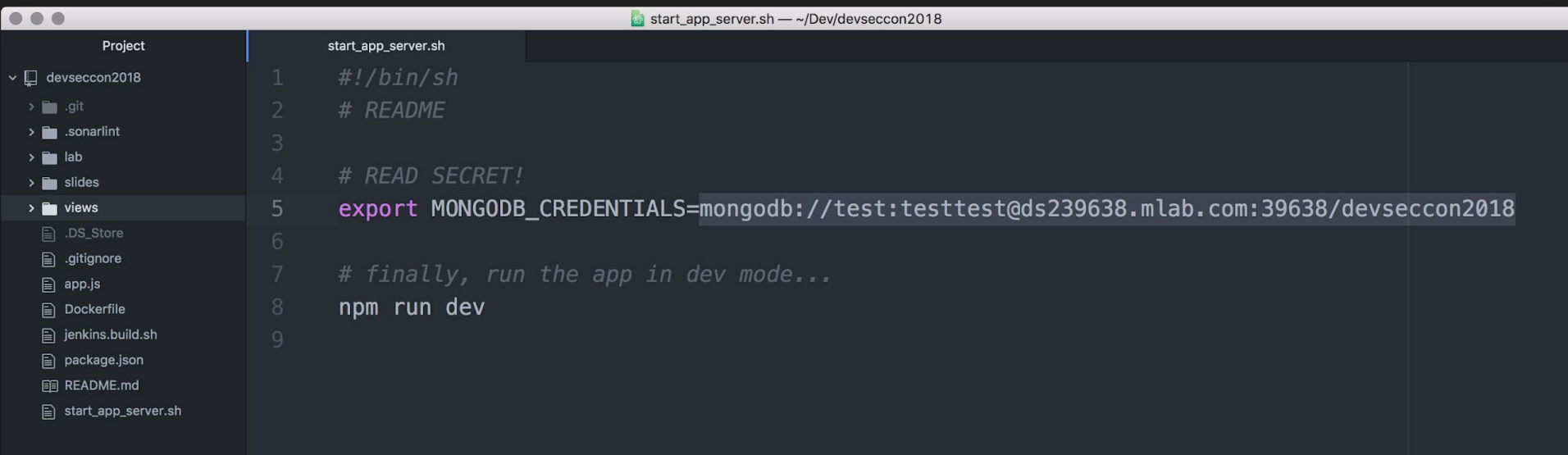
To connect using a driver via the standard MongoDB URI ([what's this?](#)):

```
mongodb://<dbuser>:<dbpassword>@ds241668.mlab.com:41668/coolname
```

In my example, mine is:

```
mongodb://triplejhacker:<dbpassword>@ds241668.mlab.com:41668/coolname
```

Paste mlab URL as MONGODB_CREDENTIALS



The screenshot shows a code editor window titled "start_app_server.sh — ~/Dev/devsecon2018". On the left is a "Project" sidebar showing a file tree for "devsecon2018" with folders ".git", ".sonarlint", "lab", "slides", and "views", and files ".DS_Store", ".gitignore", "app.js", "Dockerfile", "jenkins.build.sh", "package.json", "README.md", and "start_app_server.sh". The "views" folder is expanded. The main editor area shows the content of "start_app_server.sh" with line numbers 1 through 9. The script is a shell script that sets up environment variables and runs the application in development mode. Line 5 contains the command to export the MONGODB_CREDENTIALS environment variable with a value that is highlighted in a light blue box.

```
1  #!/bin/sh
2  # README
3
4  # READ SECRET!
5  export MONGODB_CREDENTIALS=mongodb://test:testtest@ds239638.mlab.com:39638/devsecon2018
6
7  # finally, run the app in dev mode...
8  npm run dev
9
```

Push changes to your fork

```
git add start_app_server.sh
```

```
git commit -m 'added mongo credentials'
```

```
git push origin master
```

Yes, check your secret into the code repository :P

We will remove and rotate it later, don't worry.

Access to Jenkins

Go to <http://13.228.110.97:8080>

Membership in
<https://github.com/DevSecOpsSG>
allows access to this Jenkins server.

Click “Authorize 3jmaster” (That’s me)



Authorize Jenkins DevSecCon2018



Jenkins DevSecCon2018 by **3jmaster**
wants to access your triplejhacker account



Personal user data
Email addresses (read-only)



Organizations and teams
Read-only access



Authorize 3jmaster

Authorizing will redirect to
<http://13.228.110.97:8080>



Not owned or
operated by GitHub



Created 4 days ago

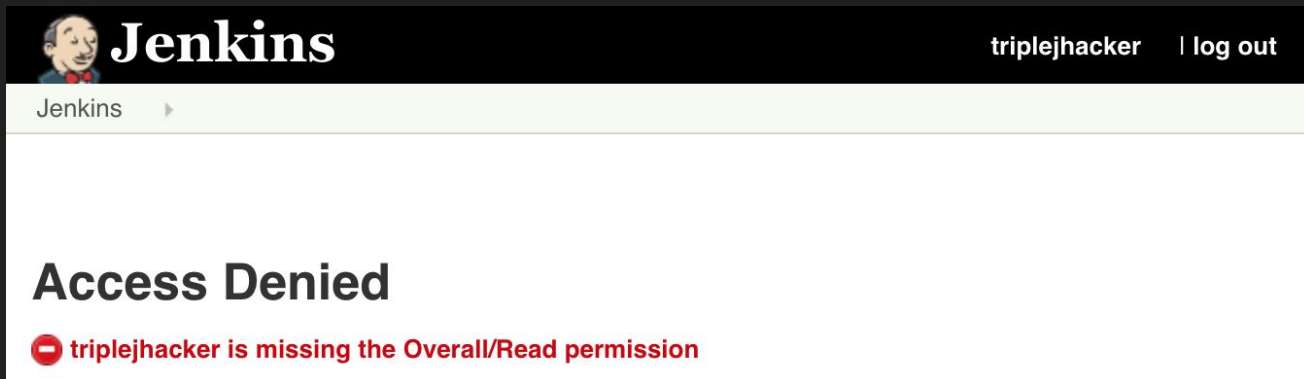


Fewer than 10
GitHub users

[Learn more about OAuth](#)

Denied Access to Jenkins

If you see this



Please perform steps in slide 5 and 6 to get access to the Github organization membership

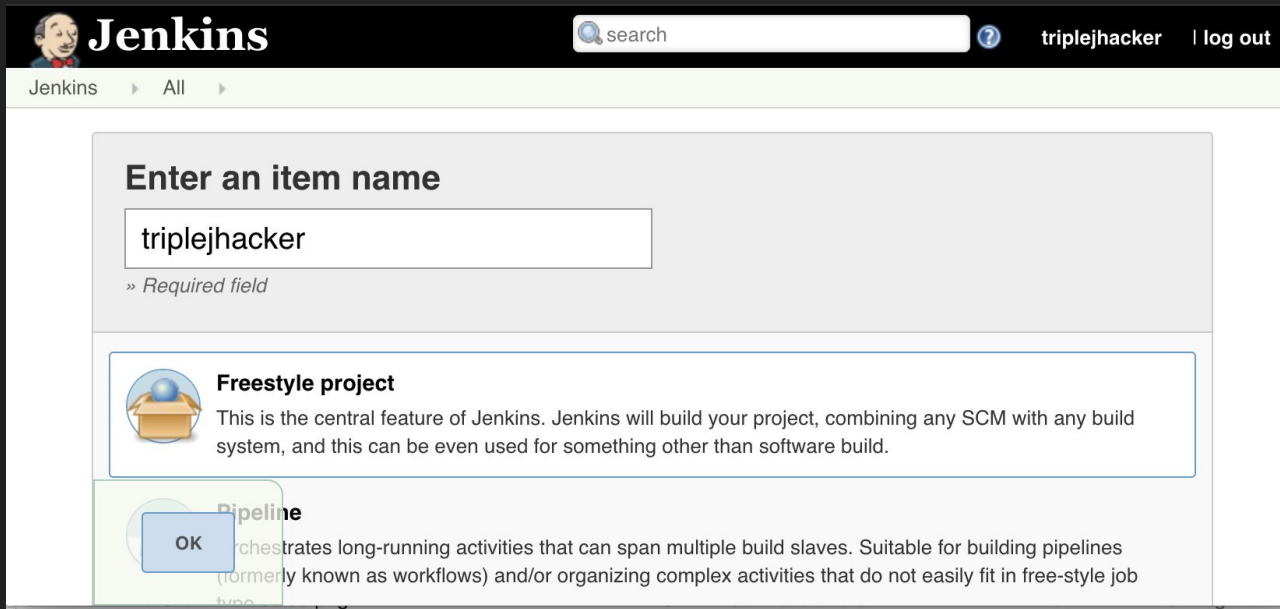
Build and Run with Jenkins

Click “New Item”



Build and Run with Jenkins

- Enter your username or any name as a project name
- Choose “Freestyle Project”
- Click “OK”



The screenshot shows the Jenkins web interface. At the top, there's a header with the Jenkins logo, a search bar, and the user 'triplejhacker' with a 'log out' link. Below the header, the breadcrumb 'Jenkins > All' is visible. The main content area is titled 'Enter an item name' and contains a text input field with the value 'triplejhacker'. Below the input field, it says '» Required field'. Underneath, there are two options: 'Freestyle project' (which is selected and highlighted with a blue border) and 'Pipeline'. The 'Freestyle project' option has a description: 'This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.' The 'Pipeline' option is partially visible below it, with a description: 'Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.' A blue 'OK' button is visible over the 'Pipeline' option.

Jenkins


search triplejhacker | log out

Jenkins > All


Enter an item name

triplejhacker

» Required field

**Freestyle project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**

Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

OK

Build and Run with Jenkins

- Go to your fork of the repository and copy the URL from “Clone or download”

The screenshot shows a web browser displaying the GitHub repository page for `triplejhacker / devsecon2018`. The repository is a fork of `DevSecOpsSG/devsecon2018`. The page includes navigation tabs for Code, Pull requests, Projects, Wiki, Insights, and Settings. Below these, it shows repository statistics: 14 commits, 2 branches, 0 releases, and 1 contributor. A 'Clone or download' button is visible, which has been clicked, opening a modal dialog. The modal offers two options: 'Clone with HTTPS' and 'Use SSH'. The HTTPS URL, `https://github.com/triplejhacker/devsecon`, is highlighted and copied to the clipboard, as indicated by a 'Copied!' notification. Below the URL, there are buttons for 'Open in Desktop' and 'Download ZIP'.

GitHub, Inc. [US] | <https://github.com/triplejhacker/devsecon2018>

BkMrk Mgr | ADDO2017 | Academics | OSCP | Finance | Buy&Sell | DevSecOps | Download | Prsnl | Entrpris Platforms

This repository | Search | Pull requests | Issues | Marketplace | Explore

triplejhacker / devsecon2018
forked from DevSecOpsSG/devsecon2018

Watch 0 | Star 0 | Fork 1

Code | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

for DevSecCon2018 workshop Edit

[Add topics](#)

14 commits | 2 branches | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

This branch is even with DevSecOpsSG:master.

Fabian Lim Merge branch 'master' of github.com:DevSecOpsSG/devsecon2018

lab update readmes

slides added devsecopssg org as a pre-req to access services

Clone with HTTPS ? [Use SSH](#)

Use Git or checkout with SVN using the web URL.

<https://github.com/triplejhacker/devsecon> Copied!

Open in Desktop | Download ZIP

Build and Run with Jenkins

- Paste it in Jenkins under “Configure”, under Source Code Management -> Git -> Repositories -> URL

The screenshot shows the Jenkins web interface for configuring a project named 'triplejhacker'. The 'Source Code Management' tab is selected, and the 'Git' option is chosen under 'Post-build Actions'. In the 'Repositories' section, the 'Repository URL' is set to 'https://github.com/triplejhacker/devsecon2018.git' and 'Credentials' is set to '- none -'. There are 'Advanced...' and 'Add Repository' buttons. In the 'Branches to build' section, the 'Branch Specifier (blank for 'any')' is set to '*/master', with an 'Add Branch' button. At the bottom, there are 'Save' and 'Apply' buttons. A red 'X' icon is visible next to the branch specifier field.

Jenkins > triplejhacker >

General **Source Code Management** Build Triggers Build Environment Build

Post-build Actions

Git

Repositories

Repository URL

Credentials Add

Advanced...

Add Repository

Branches to build

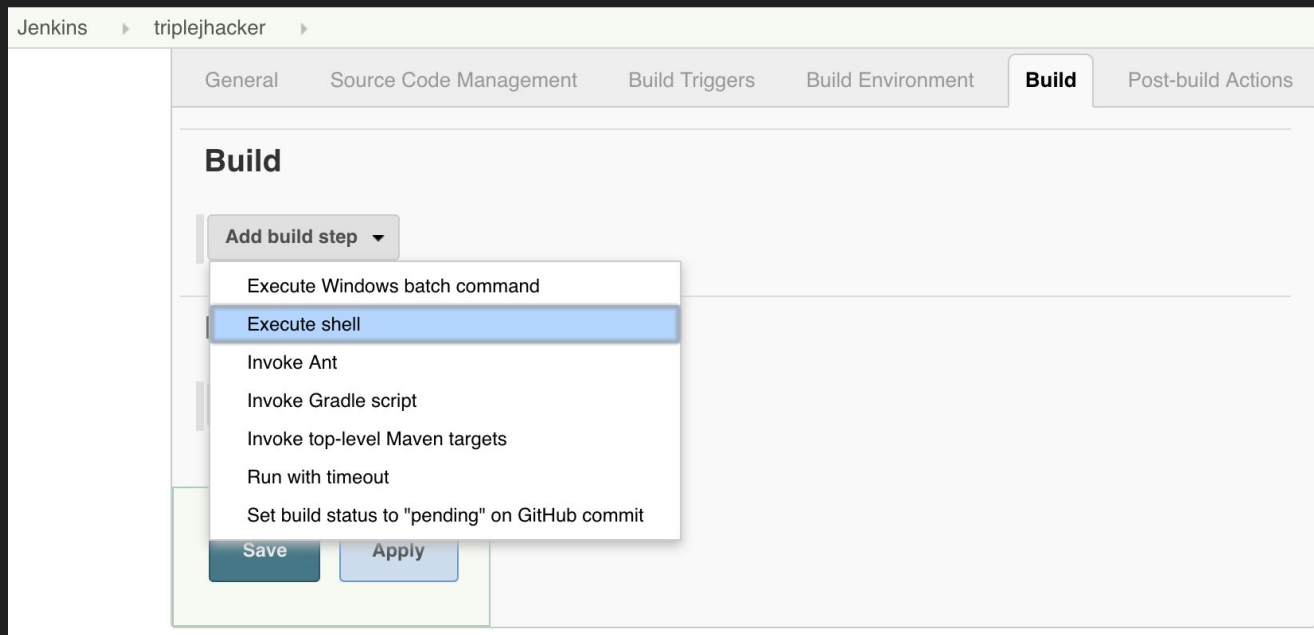
Branch Specifier (blank for 'any')

Add Branch

Save Apply

Build and Run with Jenkins

- Go to: Build -> Add Build Step -> Execute Shell



Build and Run with Jenkins

- Copy and paste the contents of jenkins.build.sh from the code repository
- Replace IMG_NAME (with your username), CTNR_NAME (with your username), PORT (with a random number between 9000 and 9999)
- ***Remember this PORT value, you will need to append it to the URL later***
- Click “Save”

Build

Execute shell

Command

```
#jenkins.build.sh
# COPY THIS TO JENKINS BUILD STEP

export IMG_NAME='triplejhacker/devseccon2018' # Name your Docker image
export CTNR_NAME='triplejhacker' # Name your Docker container (has to be unique)
export PORT='9999' # Choose a port between 9001-9999 (has to be unique)

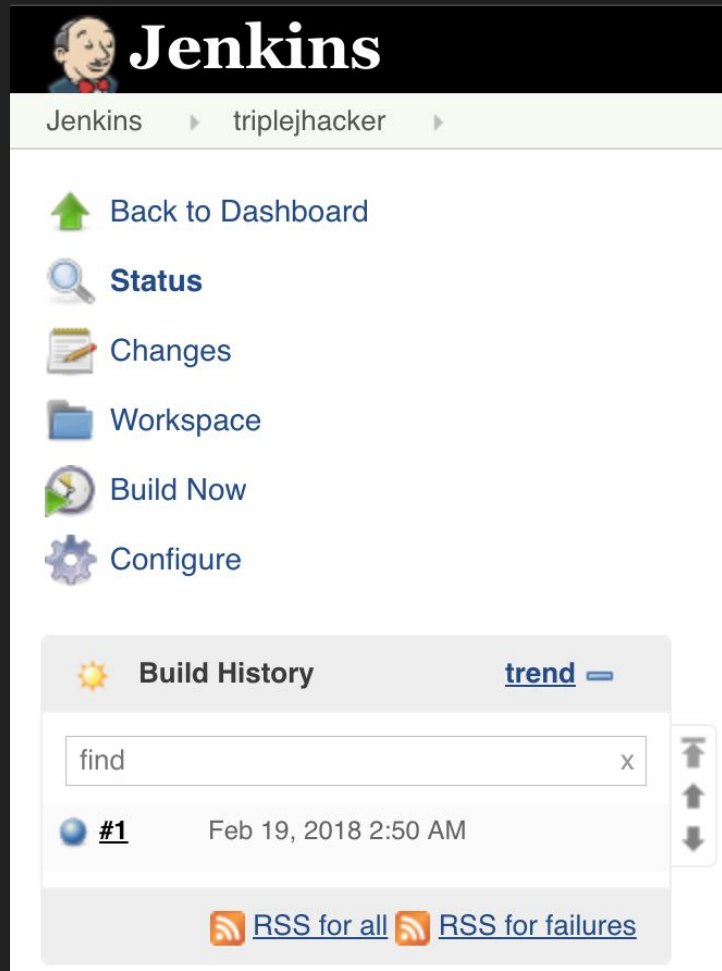
# Clean up
sudo docker kill ${CTNR_NAME} || true
sudo docker rm ${CTNR_NAME} || true
# Build
sudo docker build -t ${IMG_NAME} .
# Run
sudo docker run --network=isolated_nw \
-d \
-p ${PORT}:3000 \
--name ${CTNR_NAME} \
${IMG_NAME}
```

[See the list of available environment variables](#)

Save Apply Advanced...

Build and Run with Jenkins

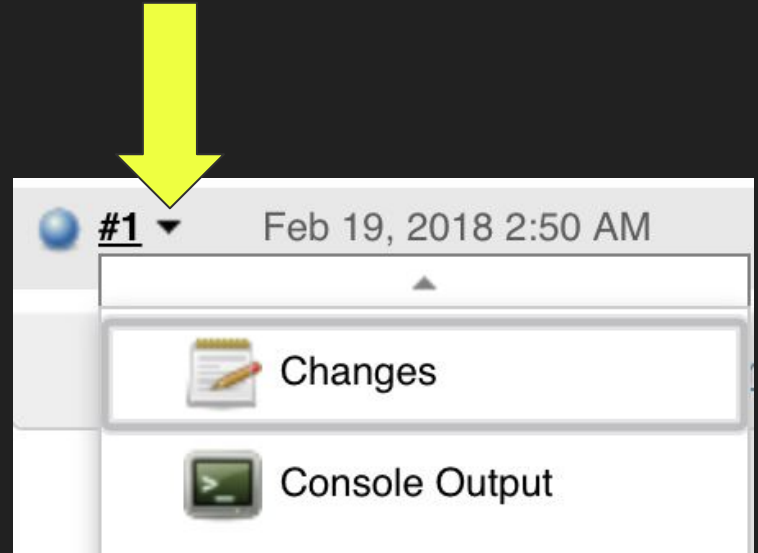
- Click “Build Now”
- Under “Build History”, there should be a build number like “#1”



The screenshot displays the Jenkins web interface. At the top, the Jenkins logo and name are visible. Below the header, a breadcrumb trail shows 'Jenkins' and 'triplejhacker'. A sidebar on the left contains several navigation links: 'Back to Dashboard' (with a green arrow icon), 'Status' (with a magnifying glass icon), 'Changes' (with a notepad icon), 'Workspace' (with a folder icon), 'Build Now' (with a play button icon), and 'Configure' (with a gear icon). The main content area features a 'Build History' section with a sun icon and a 'trend' link. Below this is a search bar with the text 'find' and a clear button 'x'. The build history table shows a single entry: a blue circle icon, the build number '#1', and the timestamp 'Feb 19, 2018 2:50 AM'. At the bottom of the build history section, there are two RSS feed links: 'RSS for all' and 'RSS for failures', each preceded by an RSS icon. On the right side of the build history table, there are three vertical arrow icons for scrolling.

Build and Run with Jenkins

- Click on the arrow beside the build number (shown here)
- Click on “Console Output” to show the logs from the build



Build and Run with Jenkins

- If all goes well, it should look something like this ending with **“Finished: SUCCESS”**
- *If you encounter error with container name, the CTNR_NAME that was already been used. So, change the value of CTNR_NAME, save project and re-run “Build Now”.*
- *If you encounter error with port number, a PORT that was already been used, so change the value of PORT, save project and re-run “Build Now”.*

Jenkins » triplejhacker » #1

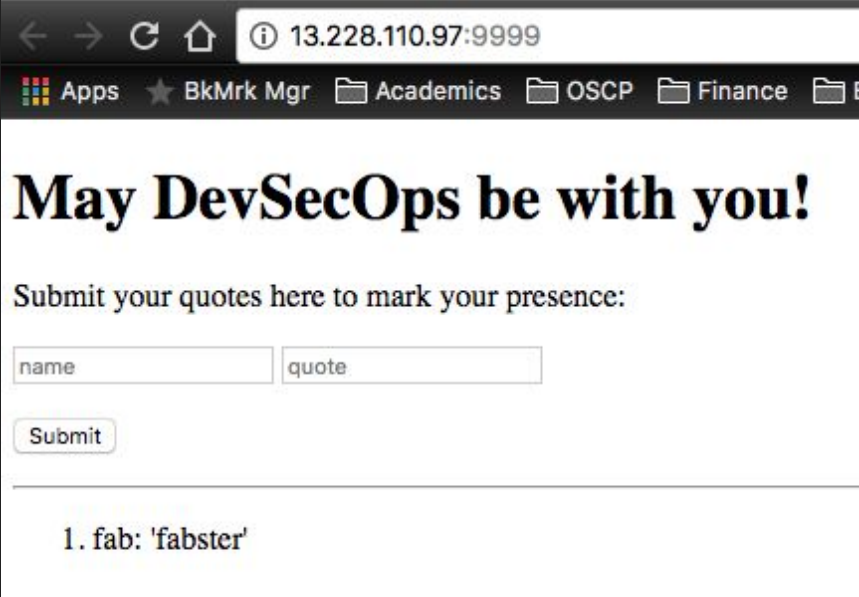
Console Output

```
Started by user triplejhacker
Building in workspace /var/lib/jenkins/workspace/triplejhacker
Cloning the remote Git repository
Cloning repository https://github.com/triplejhacker/devsecon2018.git
> git init /var/lib/jenkins/workspace/triplejhacker # timeout=10
Fetching upstream changes from https://github.com/triplejhacker/devsecon2018.git
> git --version # timeout=10
> git fetch --tags --progress https://github.com/triplejhacker/devsecon2018.git
+refs/heads/*:refs/remotes/origin/*
> git config remote.origin.url https://github.com/triplejhacker/devsecon2018.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/triplejhacker/devsecon2018.git # timeout=10
Fetching upstream changes from https://github.com/triplejhacker/devsecon2018.git
> git fetch --tags --progress https://github.com/triplejhacker/devsecon2018.git
+refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 5518845d1c9d2a08f96dffb7e87c060efb7d652 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 5518845d1c9d2a08f96dffb7e87c060efb7d652
Commit message: "Merge branch 'master' of github.com:DevSecOpsSQ/devsecon2018"
First time build. Skipping changelog.
[triplejhacker] $ /bin/sh -xe /tmp/jenkins876337294582057440.sh
+ export IMG_NAME=triplejhacker/devsecon2018
+ IMG_NAME=triplejhacker/devsecon2018
+ export CTNR_NAME=triplejhacker
+ CTNR_NAME=triplejhacker
+ export PORT=9999
+ PORT=9999
+ sudo docker kill triplejhacker
Error response from daemon: Cannot kill container: triplejhacker: No such container: triplejhacker
+ true
+ sudo docker rm triplejhacker
Error response from daemon: No such container: triplejhacker
+ true
+ sudo docker build -t triplejhacker/devsecon2018 .
Sending build context to Docker daemon 19.65MB

Step 1/8 : FROM node:alpine
--> b7e15c83cdaf
Step 2/8 : WORKDIR /usr/src
--> Using cache
--> fdad472886f
Step 3/8 : COPY package.json .
--> Using cache
--> 590500544dea
Step 4/8 : RUN npm install
--> Using cache
--> a89ace4cc5ad
Step 5/8 : COPY . .
--> 10612a60f4b7
Step 6/8 : EXPOSE 3000
--> Running in 7875e2b6c769
--> 176a023dda0e
Removing intermediate container 7875e2b6c769
Step 7/8 : COPY ./start_app_server.sh /usr/local/bin/start_app_server.sh
--> 441d495a728c
Step 8/8 : CMD start_app_server.sh
--> Running in c6562ca0c15b
--> d7f3d7cd1d1e
Removing intermediate container c6562ca0c15b
Successfully built d7f3d7cd1d1e
Successfully tagged triplejhacker/devsecon2018:latest
+ sudo docker run --network=isolated_nw -d -p 9999:3000 --name triplejhacker
triplejhacker/devsecon2018
a559e97ee9cd7b2d711a26d6f874f363f311005ebc153f161a5ef1a239af5f5
Finished: SUCCESS
```

Access your deployed app

- Append the port number you specified in jenkins to http://13.228.110.97:<specified_port> and go to this URL in your browser
- In my example, I go to <http://13.228.110.97:9999>
- A simple app should display
- Interact with the app by adding quotes
- These quotes are stored in your mongodb in mlab. You can go back to mlab and check the changes in database

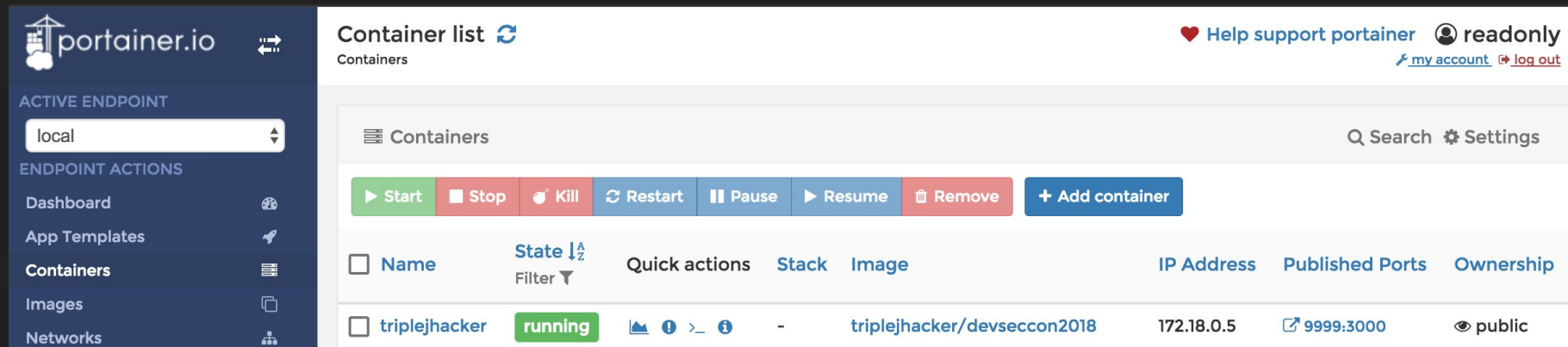


The screenshot shows a web browser window with the address bar displaying `13.228.110.97:9999`. The browser's bookmark bar includes links for 'Apps', 'BkMrk Mgr', 'Academics', 'OSCP', 'Finance', and 'E'. The main content area features a large heading **May DevSecOps be with you!**, followed by the instruction 'Submit your quotes here to mark your presence:'. Below this is a form with two input fields labeled 'name' and 'quote', and a 'Submit' button. At the bottom, a list of quotes is shown, starting with '1. fab: 'fabster''.

1. fab: 'fabster'

Debug your deployed app

- Docker UI is at <http://13.228.110.97:8100>
- Login with username and password “readonly” to view the state and logs of containers



The screenshot displays the Portainer.io web interface. On the left is a dark blue sidebar with the Portainer.io logo and navigation links: ACTIVE ENDPOINT (set to 'local'), ENDPOINT ACTIONS (Dashboard, App Templates, Containers, Images, Networks), and a user profile section. The main content area is titled 'Container list' and shows a table of containers. The first container, 'triplejhacker', is in a 'running' state. Above the table are various action buttons (Start, Stop, Kill, Restart, Pause, Resume, Remove) and an 'Add container' button. The top right of the interface includes links for help, support, and user account management.

portainer.io

ACTIVE ENDPOINT

local

ENDPOINT ACTIONS

- Dashboard
- App Templates
- Containers
- Images
- Networks

Container list

Containers

Help support portainer readonly

my account log out

Containers

Q Search Settings

Start Stop Kill Restart Pause Resume Remove Add container

<input type="checkbox"/>	Name	State <small>↓</small>	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
<input type="checkbox"/>	triplejhacker	running		-	triplejhacker/devseccon2018	172.18.0.5	9999:3000	public

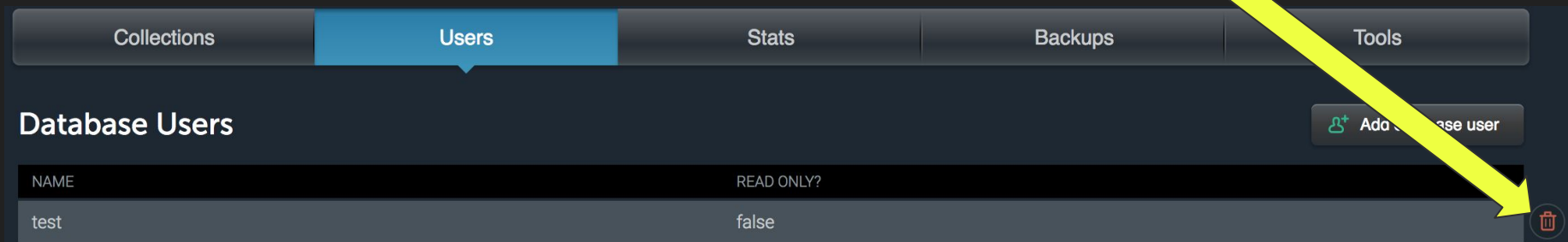
Back to Slide

https://docs.google.com/presentation/d/1jW0pPXheS2aZqsXvfPATQLbY5sDRyGVpuNpswS5Zv4l/edit#slide=id.g31d5e508b0_0_590

**Solution 1 - Remove secret from App,
secret stays in Jenkins**

Rotate mlab (MongoDB) credentials

- Back on the mlab page,
- Click on “Users”
- Click on the trash bin icon to delete the user

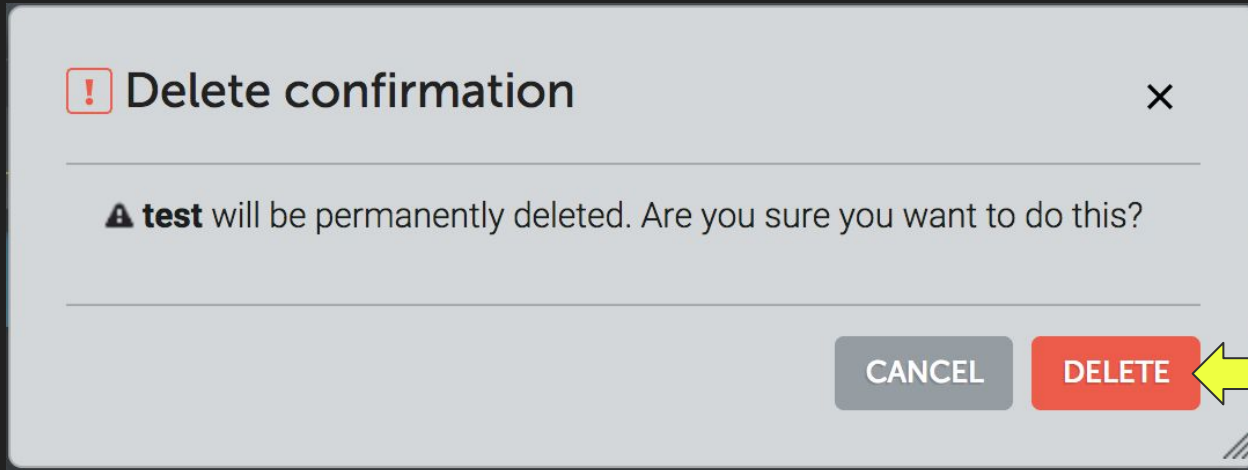


The screenshot shows the MongoDB Atlas interface. At the top, there is a navigation bar with tabs: Collections, Users (highlighted in blue), Stats, Backups, and Tools. Below the navigation bar, the title "Database Users" is displayed on the left, and a button labeled "Add database user" with a green plus icon is on the right. A table lists the database users. The table has two columns: "NAME" and "READ ONLY?". The first row shows a user named "test" with "false" in the "READ ONLY?" column. At the end of the "test" row, there is a red trash bin icon. A large yellow arrow points from the text "Click on the trash bin icon to delete the user" in the list above to this trash bin icon.

NAME	READ ONLY?
test	false

Rotate mlab (MongoDB) credentials

- Go ahead and Click on “DELETE”



Rotate mlab (MongoDB) credentials

- Click on “Add database user”

The screenshot displays the MongoDB Atlas console interface. At the top, the 'Database: coolname' is selected. A yellow arrow points from the 'Users' tab in the bottom navigation bar to the 'Add database user' button in the top right corner of the 'Database Users' section. The 'Database Users' section shows '[None at this time]'. The 'Users' tab is highlighted in blue. The console also displays connection instructions for the MongoDB shell and a standard MongoDB URI, along with a warning about sandbox databases.

Home
Database: coolname Delete database

To connect using the mongo shell:
`% mongo ds241668.mlab.com:41668/coolname -u <dbuser> -p <dbpassword>`

To connect using a driver via the standard MongoDB URI ([what's this?](#)):
`mongodb://<dbuser>:<dbpassword>@ds241668.mlab.com:41668/coolname`

mongod version: 3.4.13 (MMAPv1)

⚠ Sandbox databases do not have redundancy and therefore are not suitable for production. Read our documentation on [how to upgrade](#).

⚠ A database user is required to connect to this database. To create one now, visit the 'Users' tab and click the 'Add database user' button.

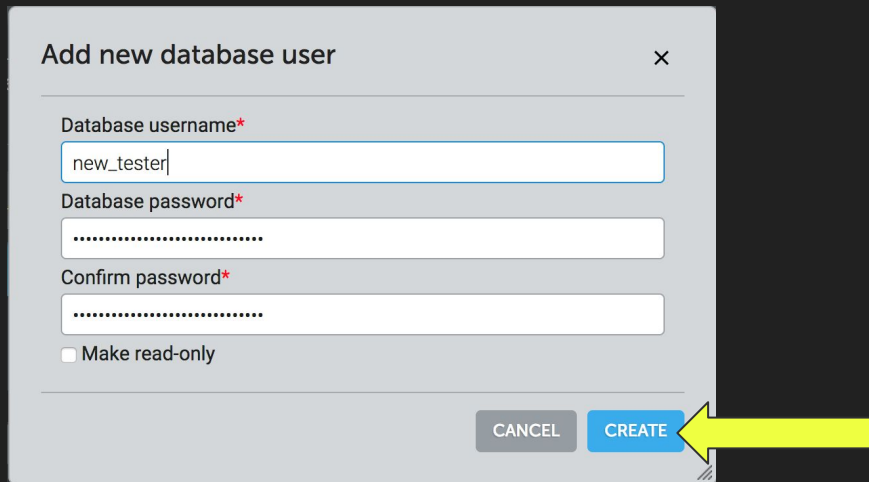
Collections **Users** Stats Backups Tools

Database Users Add database user

[None at this time]

Rotate mlab (MongoDB) credentials

- Enter a NEW username and password (don't use anything personal or sensitive)
 - To avoid syntax error in the later steps, do not use the “@” symbol at all; you can use it but you must [encode the character](#) in the script later
- Leave “Make read-only” box unchecked
- Click “Create”



Add new database user

Database username*

new_tester

Database password*

Confirm password*

☐ Make read-only

CANCEL CREATE

Rotate mlab (MongoDB) credentials

From the database page, construct your mlab mongodb instance URL (this contains secret) replacing <dbuser> and <dbpassword> that you previously entered.

[Home](#)

Database: coolname

To connect using the mongo shell:

```
% mongo ds241668.mlab.com:41668/coolname -u <dbuser> -p <dbpassword>
```

To connect using a driver via the standard MongoDB URI ([what's this?](#)):

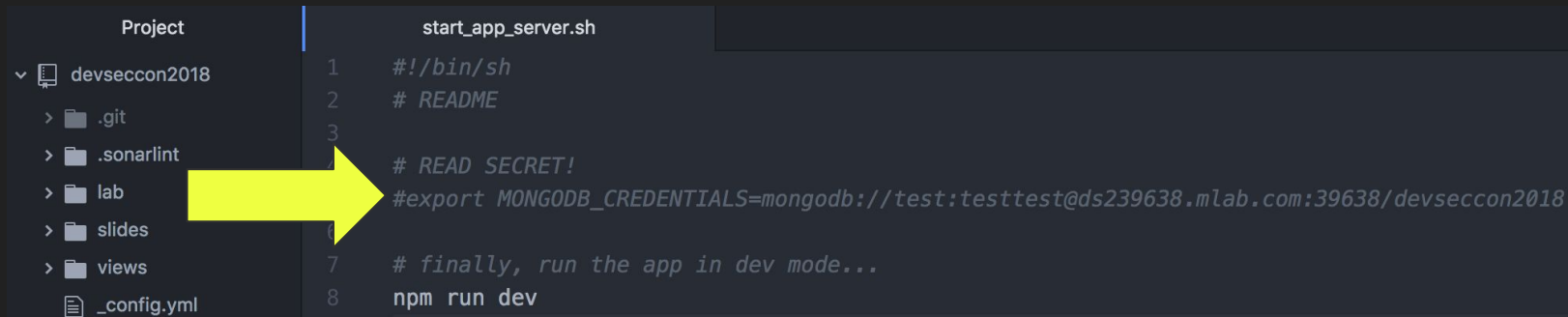
```
mongodb://<dbuser>:<dbpassword>@ds241668.mlab.com:41668/coolname
```

In my example, mine is:

```
mongodb://new_tester:<dbpassword>@ds241668.mlab.com:41668/coolname
```

Remove secret from code repository

From your local machine, delete or comment out the secret mongodb url:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'devseccon2018' with subfolders '.git', '.sonarlint', 'lab', 'slides', and 'views', and a file '_config.yml'. The code editor shows a file named 'start_app_server.sh' with the following content:

```
1  #!/bin/sh
2  # README
3
4  # READ SECRET!
5  #export MONGODB_CREDENTIALS=mongodb://test:testtest@ds239638.mlab.com:39638/devseccon2018
6
7  # finally, run the app in dev mode...
8  npm run dev
```

A yellow arrow points to the line containing the MongoDB URL secret: `#export MONGODB_CREDENTIALS=mongodb://test:testtest@ds239638.mlab.com:39638/devseccon2018`.

Remove secret from code repository

From your local machine in the directory where the git repository is, run:

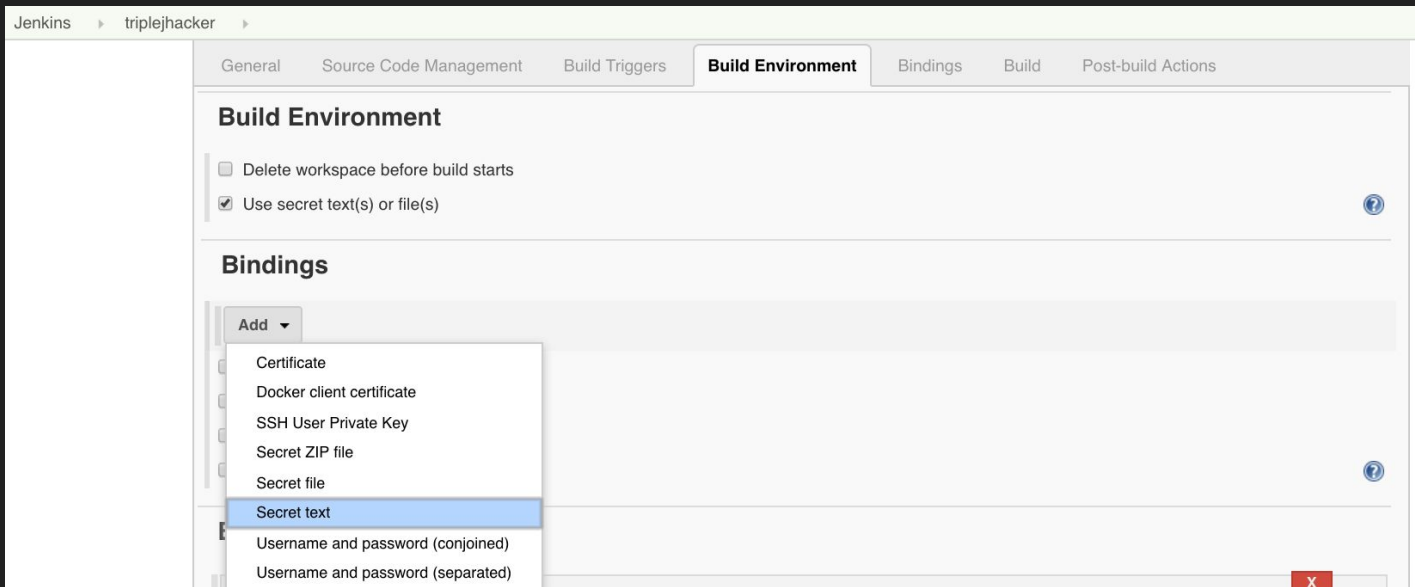
```
git add start_app_server.sh
```

```
git commit -m 'removed secret'
```

```
git push origin master
```

Store secrets within Jenkins

- Go to your previously created Jenkins project, under “Configure”
- Check “Use secret text(s) file(s)”
- Click “Secret Text”



Store secrets within Jenkins

- Fill Variable as “mongodb”
- Add to “Jenkins”

Bindings

Secret text

Variable

Credentials ☒ Specific credentials ☐ Parameter expression

⬆ ⬇ ⬆

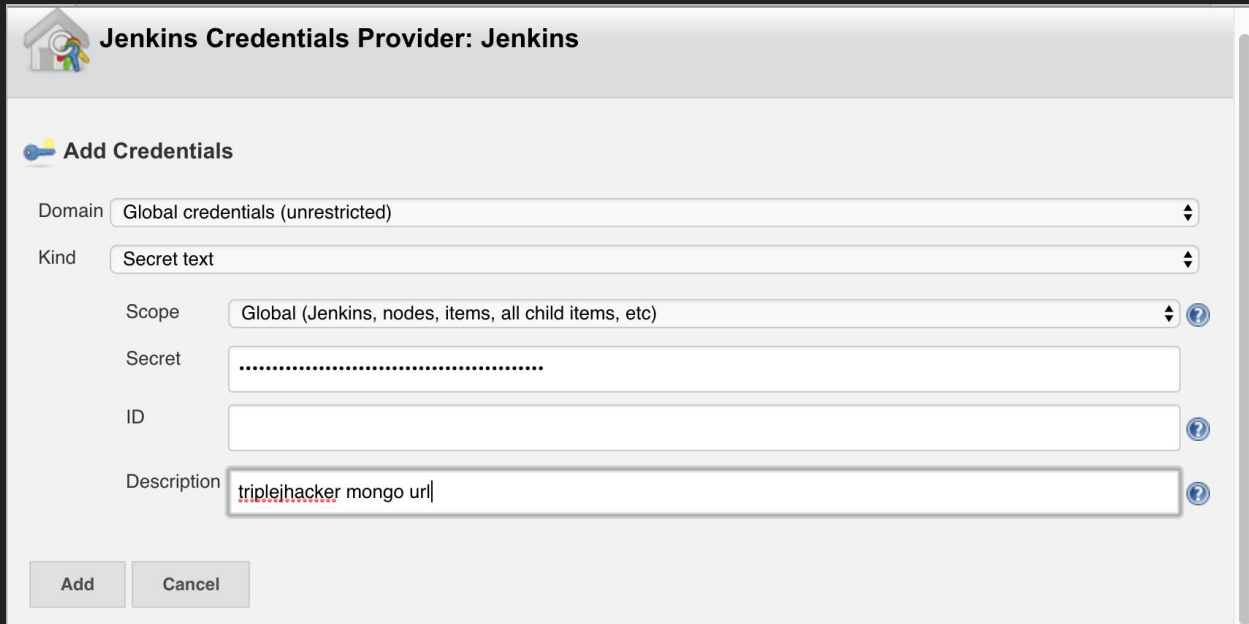
Add

Jenkins

Add ▾

Store secrets within Jenkins

- Choose “Secret text” as kind
- Fill Secret as your mongodb URL (from slide 7)
- Fill Description with your username (for easy identification)
- Click “Add”



The screenshot shows the 'Jenkins Credentials Provider: Jenkins' window with the 'Add Credentials' tab selected. The form contains the following fields:

- Domain:** A dropdown menu set to 'Global credentials (unrestricted)'.
- Kind:** A dropdown menu set to 'Secret text'.
- Scope:** A dropdown menu set to 'Global (Jenkins, nodes, items, all child items, etc)' with a help icon.
- Secret:** A text input field containing a series of dots to mask the text.
- ID:** An empty text input field with a help icon.
- Description:** A text input field containing the text 'tripleihacker mongo url' with a help icon.

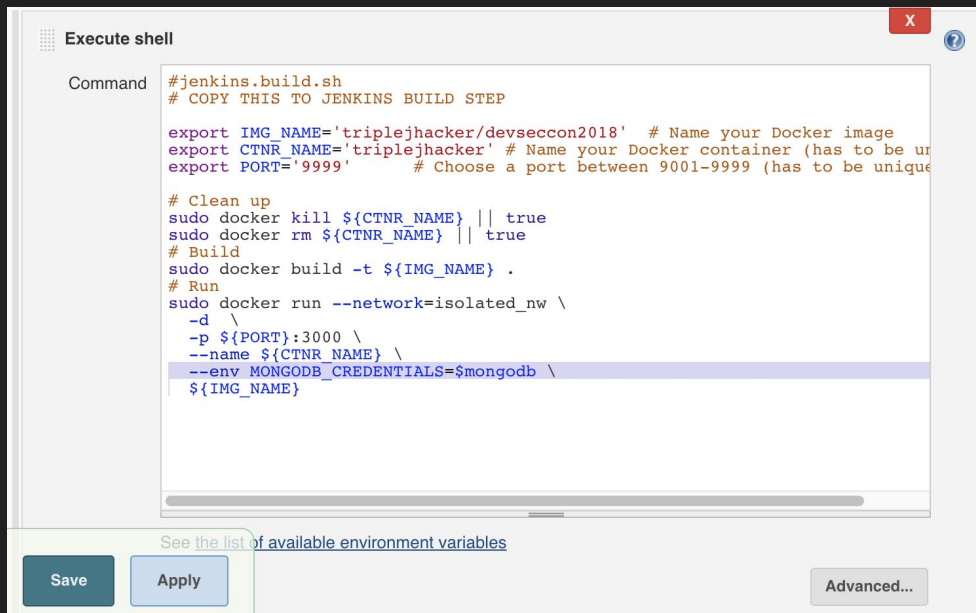
At the bottom of the window are two buttons: 'Add' and 'Cancel'.

Store secrets within Jenkins

- In the “Execute Shell”, add a line under `docker run`:

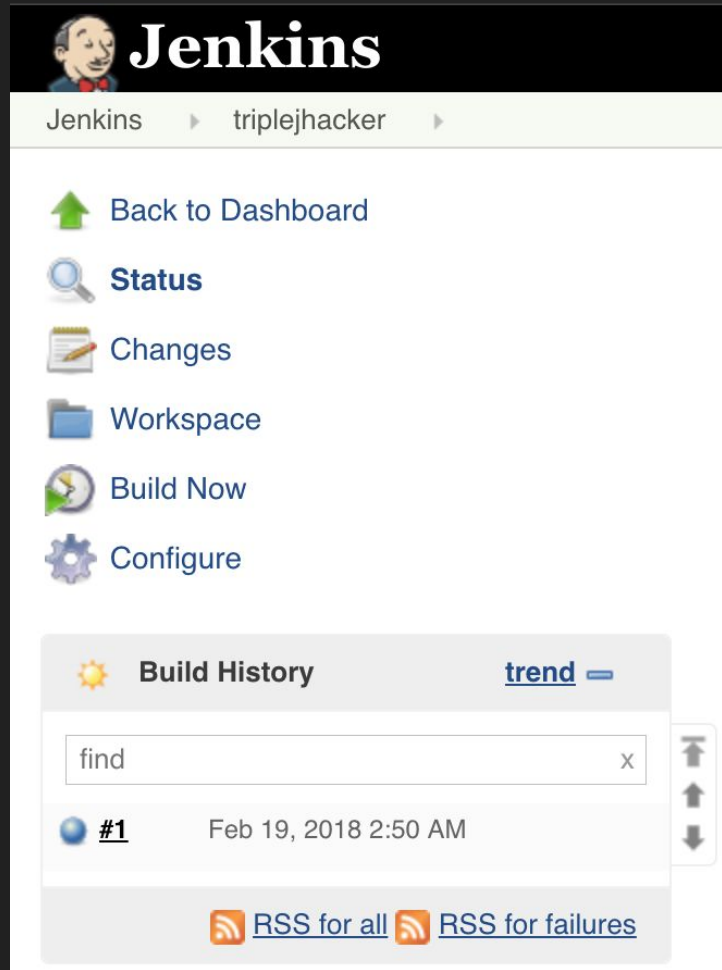
`--env MONGODB_CREDENTIALS=$mongodb \`

MUST add the backslash “\” at the end!



Build and Run with Jenkins


- Click “Build Now”
- Under “Build History”, there should be a build number like “#1” or “#2” or...





The screenshot shows the Jenkins web interface. At the top, there's a header with the Jenkins logo and the name 'Jenkins'. Below the header, there's a breadcrumb trail: 'Jenkins > triplejhacker >'. A sidebar on the left contains several icons and links: 'Back to Dashboard' (green arrow), 'Status' (magnifying glass), 'Changes' (notepad), 'Workspace' (folder), 'Build Now' (circular arrow), and 'Configure' (gear). The main content area has a 'Build History' section with a 'trend' link. Below this is a search bar with the text 'find' and a clear button 'x'. The build history table shows a single entry: a blue circle icon, the build number '#1', and the timestamp 'Feb 19, 2018 2:50 AM'. At the bottom, there are two RSS feed links: 'RSS for all' and 'RSS for failures'.


Jenkins


Jenkins > triplejhacker >


 Back to Dashboard


 Status

 Changes


 Workspace



 Build Now

 Configure

 Build History [trend](#)

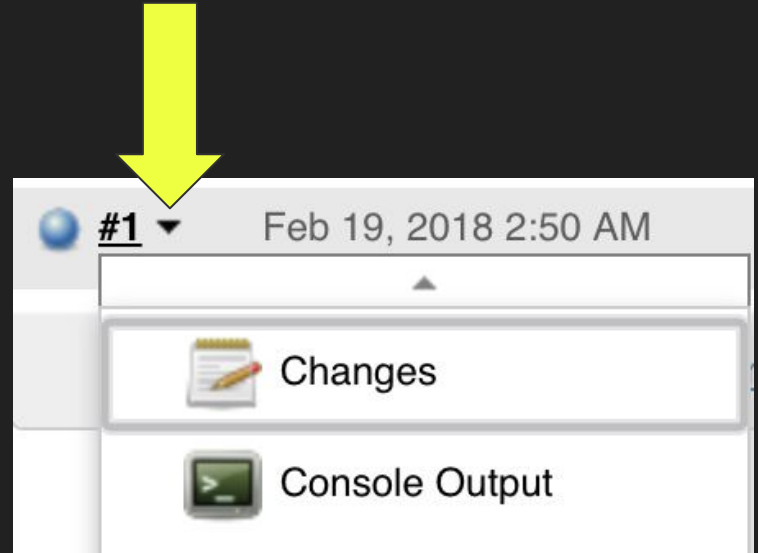
find x

 **#1** Feb 19, 2018 2:50 AM

 [RSS for all](#)  [RSS for failures](#)

Build and Run with Jenkins

- Click on the arrow beside the build number (shown here)
- Click on “Console Output” to show the logs from the build



Build and Run with Jenkins

- If all goes well, it should be pulling from your latest code commit, check the commit message.
 - It should be the same as the value in “`git commit -m 'remove secret'`” ran earlier

Jenkins ▶ triplejhacker ▶ #4

Commit message: "remove secret"

> git rev-list --no-walk 5518845d

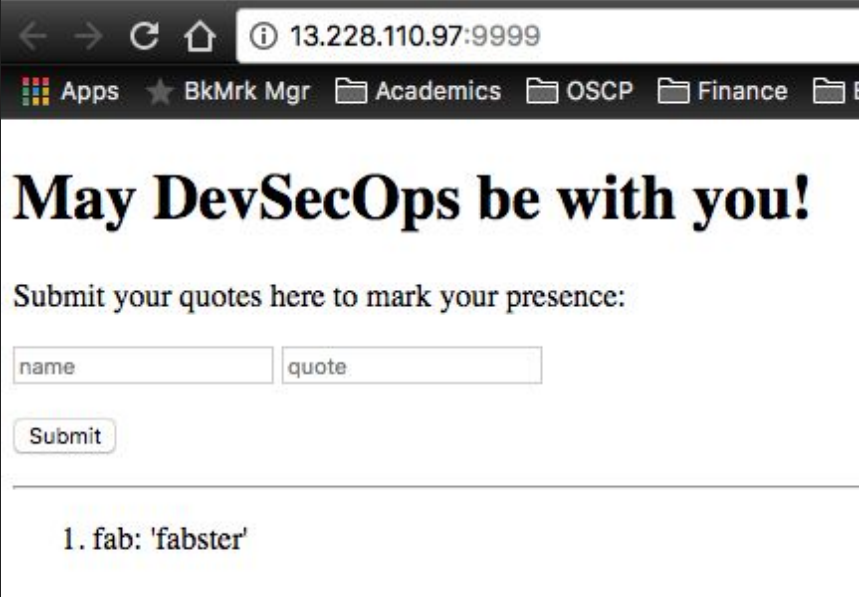
Build and Run with Jenkins

- If all goes well, it should look something like this ending with “**Finished: SUCCESS**”
- Note that the secret is also masked out. Good job Jenkins!

```
+ sudo docker run --network=isolated_nw -d -p 9999:3000 --name triplejhacker --env MONGODB_CREDENTIALS=****  
triplejhacker/devseccon2018  
2704aa320979f1c615df5db102fac4e0242bc5c6fbabada54c3809bbc4ba08a4  
Finished: SUCCESS
```

Access your deployed app

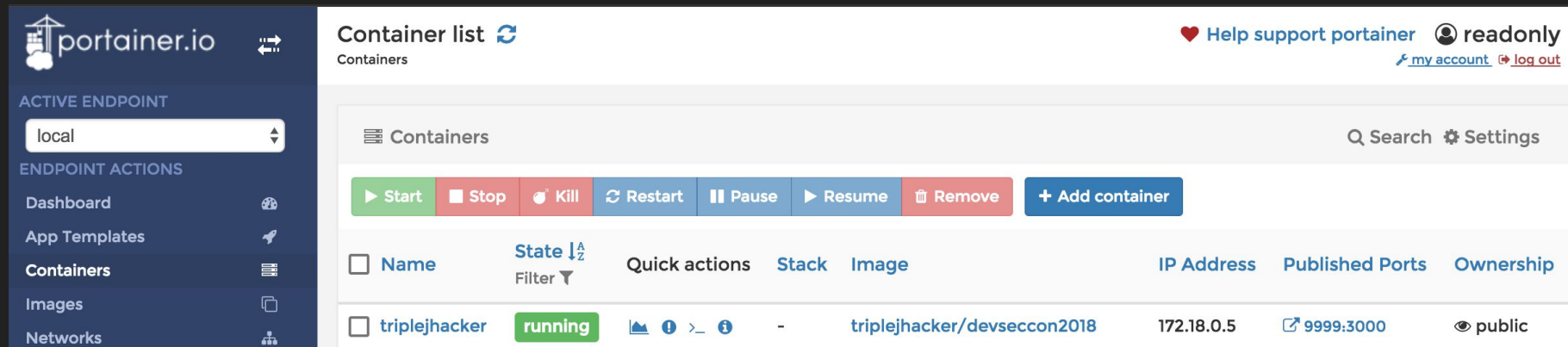
- Append the port number you specified in jenkins to http://13.228.110.97:<specified_port> and go to this URL in your browser
- In my example, I go to <http://13.228.110.97:9999>
- A simple app should display
- Interact with the app by adding quotes
- These quotes are stored in your mongodb in mlab. You can go back to mlab and check the changes in database



The screenshot shows a web browser window with the address bar displaying `13.228.110.97:9999`. The browser's bookmark bar includes links for 'Apps', 'BkMrk Mgr', 'Academics', 'OSCP', 'Finance', and 'E'. The main content of the page features the heading 'May DevSecOps be with you!' in a large, bold, black serif font. Below the heading, a text prompt reads 'Submit your quotes here to mark your presence:'. This is followed by a form with two input fields: 'name' and 'quote'. A 'Submit' button is positioned below these fields. At the bottom of the page, a single quote is listed: '1. fab: 'fabster''.

Debug your deployed app

- Docker UI is at <http://13.228.110.97:8100>
- Login with username and password “readonly” to view the state and logs of containers



The screenshot displays the Portainer.io web interface. On the left is a dark blue sidebar with the Portainer.io logo and navigation links: ACTIVE ENDPOINT (set to 'local'), ENDPOINT ACTIONS (Dashboard, App Templates, Containers, Images, Networks), and a user profile section. The main content area is titled 'Container list' and shows a table of containers. The table has columns for Name, State, Quick actions, Stack, Image, IP Address, Published Ports, and Ownership. One container, 'triplejhacker', is listed with a state of 'running'. Above the table are buttons for Start, Stop, Kill, Restart, Pause, Resume, Remove, and Add container. The top right of the interface includes links for Help, support, and a user profile for 'readonly'.

portainer.io

ACTIVE ENDPOINT

local

ENDPOINT ACTIONS

- Dashboard
- App Templates
- Containers
- Images
- Networks

Container list

Containers

Help support portainer readonly

my account log out

Containers

Q Search Settings

Start Stop Kill Restart Pause Resume Remove Add container

<input type="checkbox"/> Name	State ^A ₂ Filter ▼	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
<input type="checkbox"/> triplejhacker	running		-	triplejhacker/devseccon2018	172.18.0.5	9999:3000	public

Congrats!

**You have just removed (and rotated) a
shameful secret!**

But is this good enough?

Solution 2 - Remove Secrets from App and Jenkins, using Vault and its App Role

What is App Role?

<https://www.vaultproject.io/docs/auth/approle.html>

Access to Vault

- Membership in <https://github.com/DevSecOpsSG> allows access to Vault
- Generate a GitHub personal access token to login -> Follow Steps 1-9 on: <https://help.github.com/articles/creating-a-personal-access-token-for-the-command-line/>
- Scopes define the access for personal tokens: Check “**read:org**” only

<input type="checkbox"/> admin:org	Full control of orgs and teams
<input type="checkbox"/> write:org	Read and write org and team membership
<input checked="" type="checkbox"/> read:org	Read org and team membership

Access to Vault (GUI version)

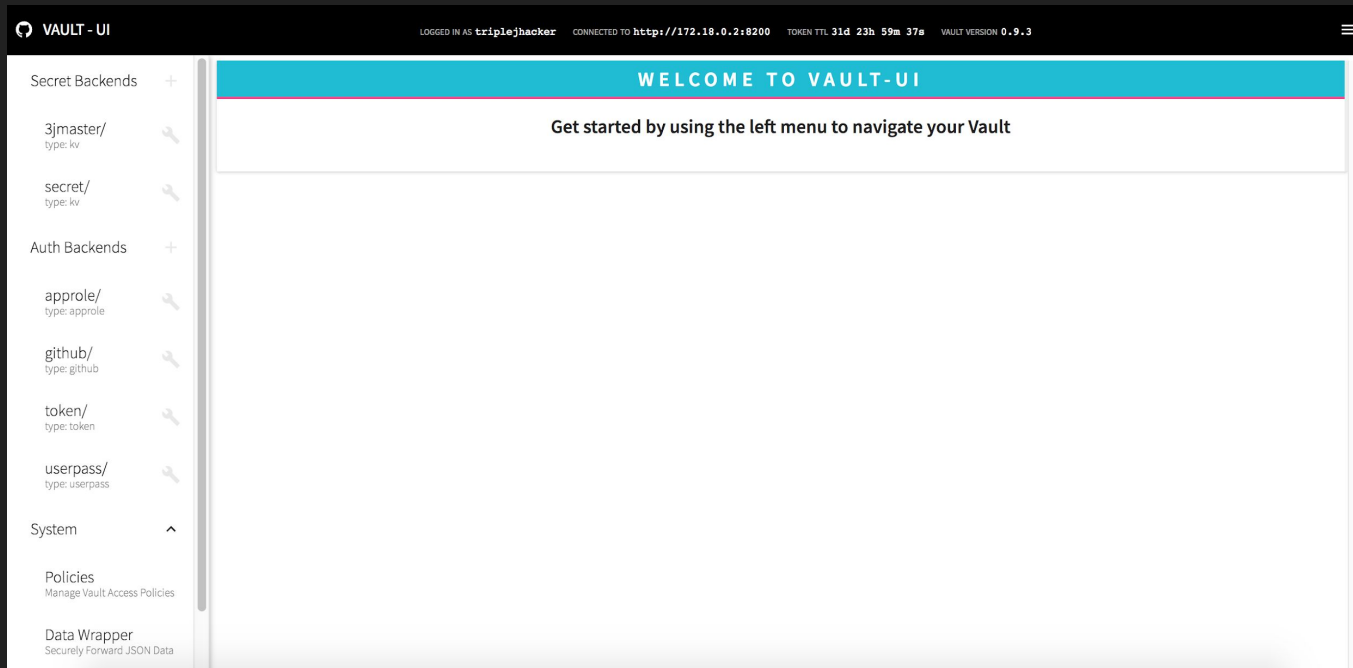
- This is Vault's UI (web container)

Note:

Vault UI

runs on port 8300

*While Vault Server
runs on port 8200*



Access to Vault (CLI version)

- Replace the GitHub Personal Token with your own and run:

```
$ export VAULT_ADDR=http://13.228.110.97:8200
$ vault auth -method=github token=<$YOUR_GITHUB_PERSONAL_TOKEN>
Successfully authenticated! You are now logged in.
The token below is already saved in the session. You do not
need to "vault auth" again with the token.
token: *****
token_duration: 2764799
token_policies: [default]
```

Access to Vault (CLI version)

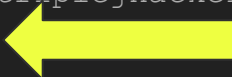
- Replace the username and secret with your own and run:

```
$ vault write secret/example/triplejhacker mongodb=<$MONGODB_URL_SECRET>
```

```
Success! Data written to: secret/example/triplejhacker
```

```
$ vault read secret/example/triplejhacker
```

Key	Value
---	-----
refresh_interval	768h0m0s
mongodb	<\$MONGODB_URL_SECRET>

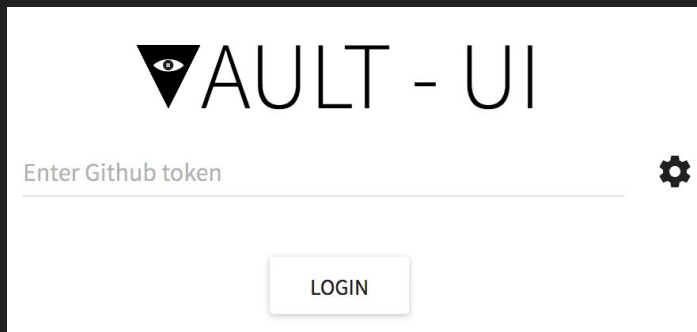


This command overwrites any existing values

You can now skip slides for Access to Vault (GUI version)

Access to Vault (GUI version)

- Go to <http://13.228.110.97:8300>
- Paste the GitHub Personal Token and login



VAULT - UI

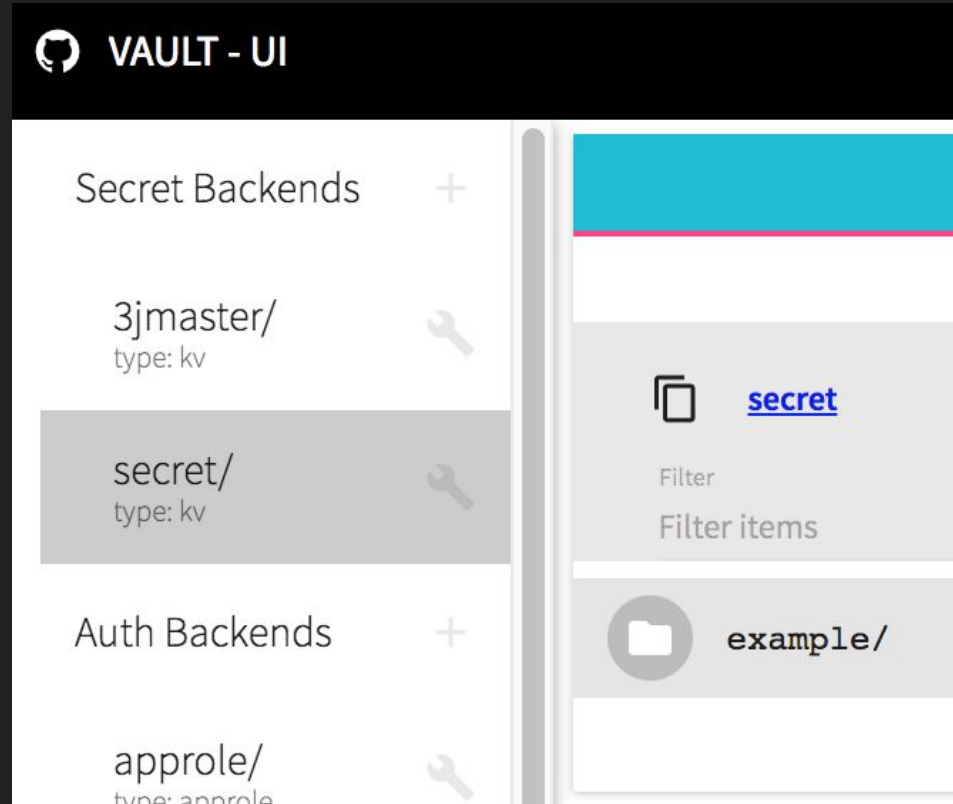
Enter Github token

LOGIN

- If it is not showing this, then Click on settings and choose "GitHub" as Login Method. *Do not change the Vault Server URL*
- Click "OK"
- Paste the GitHub Personal Token and login

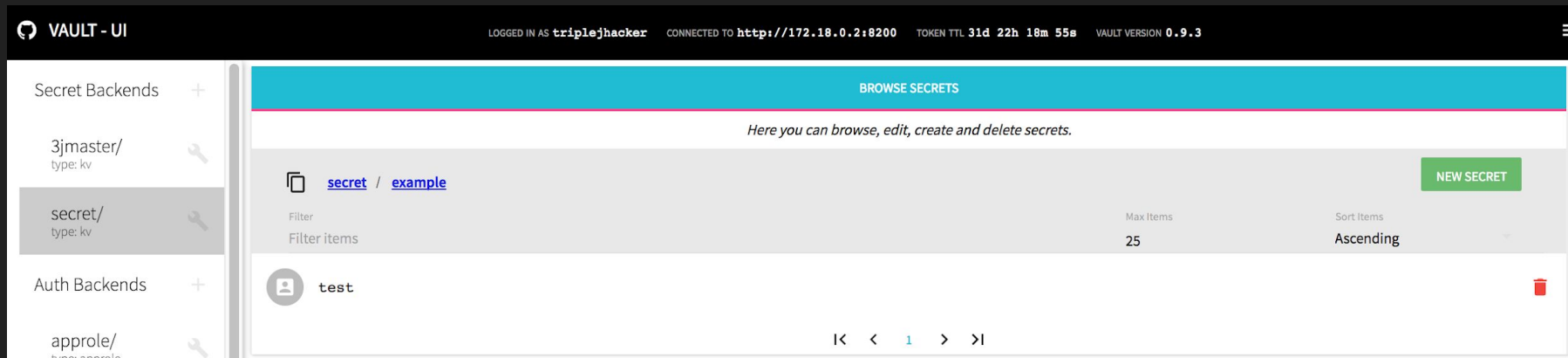
Access to Vault (GUI version)

- Under Secret Backend, click “secret/”
- Click “example/” folder



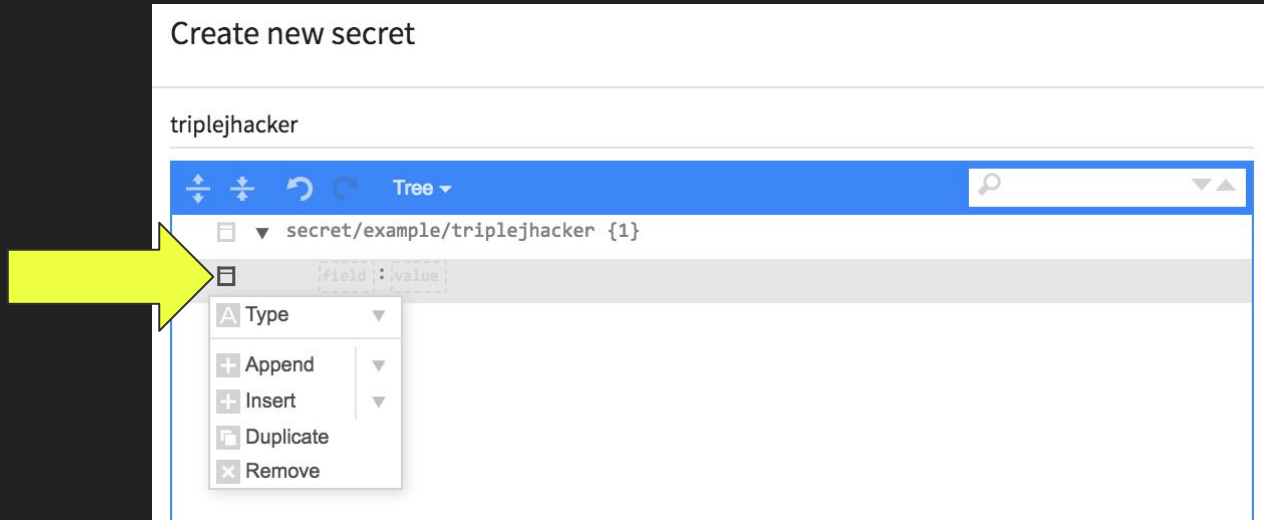
Access to Vault (GUI version)

- Click “test” item to view its key value
- Click “NEW SECRET” at the far right to create a new item



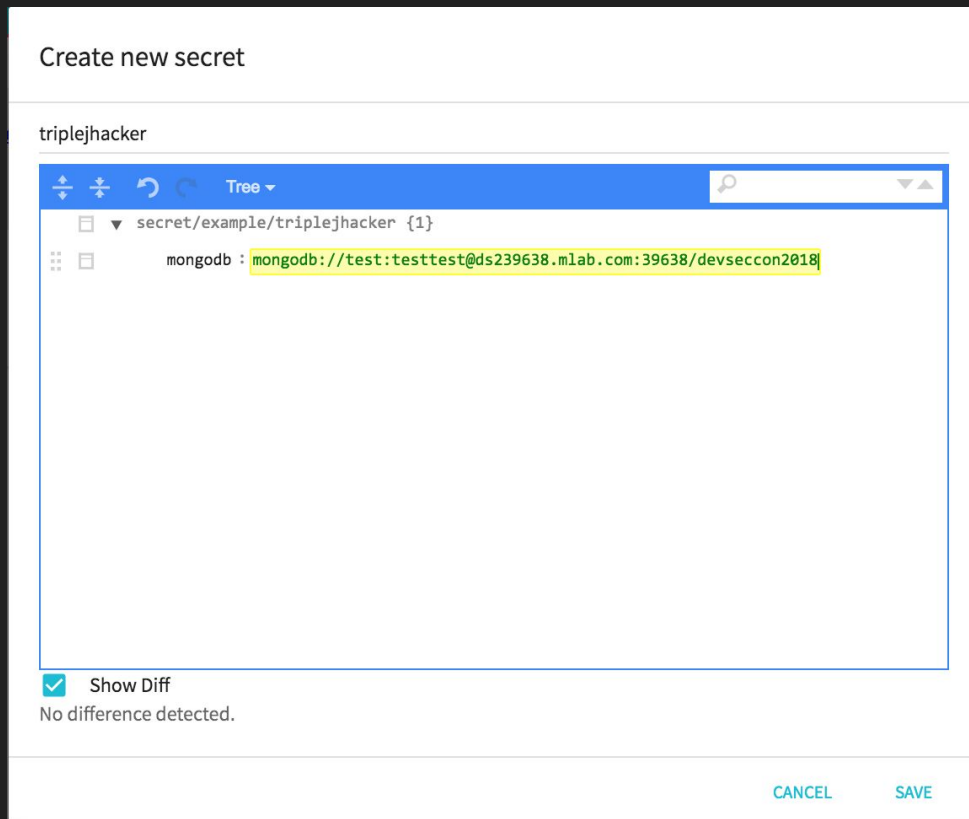
Access to Vault (GUI version)

- Fill <Insert object key> as your username
- Click on the box icon (arrow pointing) and choose “Append”, “field:value” boxes will appear



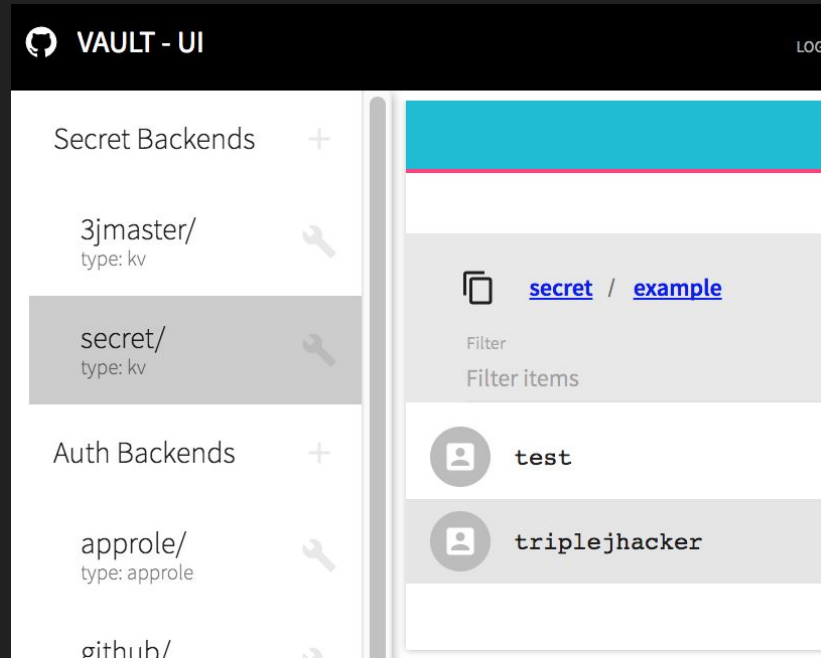
Access to Vault (GUI version)

- Fill “field” as ‘mongodb’
- Fill “value” as the secret mongodb URL in your app from slide 7
 - mongodb://...
- Click “Save”



Access to Vault (GUI version)

- Your secret is created in Vault, click on it to view
- Note its path: i.e. `secret/example/<username>`



Retrieve secret from Vault with Jenkins

- Navigate back to your Jenkins project, under “Configure”
- Uncheck previous “Use secret text(s) file(s)” box
- Check “Vault Plugin” box

Build Environment

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s)
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☒ Vault Plugin

Vault Plugin

Retrieve secret from Vault with Jenkins

- Fill Vault URL as <http://172.18.0.2:8200>
- Click “Add” -> “Jenkins”

Vault Plugin

Vault URL	<input type="text" value="http://172.18.0.2:8200"/>
Vault Credential	<div><div>- none -</div><div><div>Add</div><div>Jenkins</div></div></div>
<input type="checkbox"/> With Ant	

Jenkins Credentials Provider

Retrieve secret from Vault with Jenkins (CLI version)

- In the same terminal, run:

```
$ vault token-lookup
```

Key	Value
---	----
accessor	4fb12012-fb92-8d84-a4ed-bdb820532739
creation_time	1519349293
creation_ttl	2764800
display_name	github-triplejhacker
entity_id	ab3297fe-5fc2-5dca-f38e-c2716151774f
expire_time	2018-03-27T01:28:13.581448089Z
explicit_max_ttl	0
id	ce38db15-****-524f-482c-*****
issue_time	2018-02-23T01:28:13.581440761Z
meta	map[org:DevSecOpsSG username:triplejhacker]
num_uses	0
orphan	true
path	auth/github/login
policies	[default]
renewable	true
ttl	2764132

This is your vault token

Retrieve secret from Vault with Jenkins (CLI version)

- In the same terminal, run:

```
$ vault read auth/approle/role/example/role-id
```

```
Key          Value
```

```
---
```

```
role_id      e4964208-6fed-882b-7739-ace170ec5aba
```

```
$ vault write -f auth/approle/role/example/secret-id
```

```
Key          Value
```

```
---
```

```
secret_id      acb24ed4-1232-298a-abd4-4ad0ac77c461
```

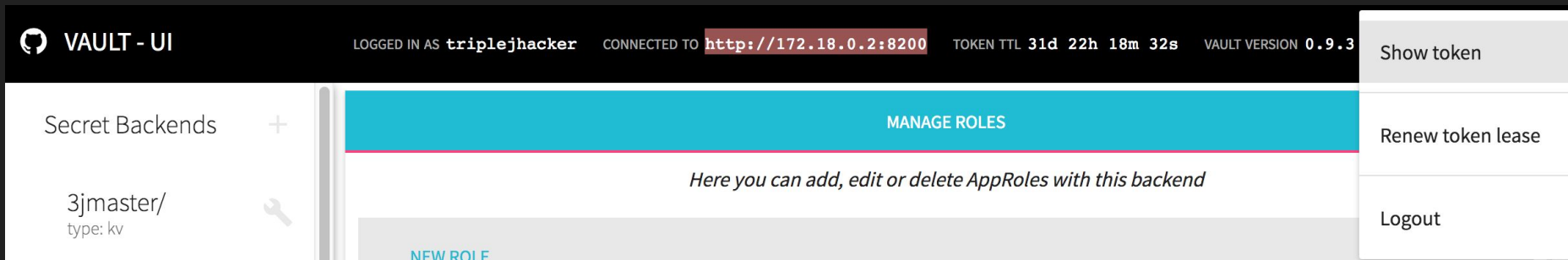
```
secret_id_accessor b84b516e-eb53-c5ff-8b5c-*****
```

Secret-id is uniquely
generated each time this
command is ran

You can now skip slides for Retrieve secret from Vault with Jenkins (GUI version)

Retrieve secret from Vault with Jenkins (GUI version)

- Go to <http://13.228.110.97:8300>
- Click on the top right corner and “Show token”
- Copy the vault token
- *Note: This is generated by Vault and is different from the GitHub Token*



Retrieve secret from Vault with Jenkins (GUI version)

- Sorry, there's no GUI for this!
- Paste your vault token from the previous slide
- Using curl, or <https://www.getpostman.com/apps> or any request tool, to GET request:

```
curl --header "X-Vault-Token: <REPLACE WITH VAULT TOKEN>"  
http://13.228.110.97:8200/v1/auth/approle/role/example/role-id
```

- Copy the **role-id**
 - {"request_id":"8d789757-ab4a-de80-9783-0927ac926f35","lease_id":"","renewable":false,"lease_duration":0,"data":{"role_id":"**e4964208-6fed-882b-7739-ace170ec5aba**"}, "wrap_info":null, "warnings":null, "auth":null}

Retrieve secret from Vault with Jenkins (GUI version)

- Sorry, there's no GUI for this!

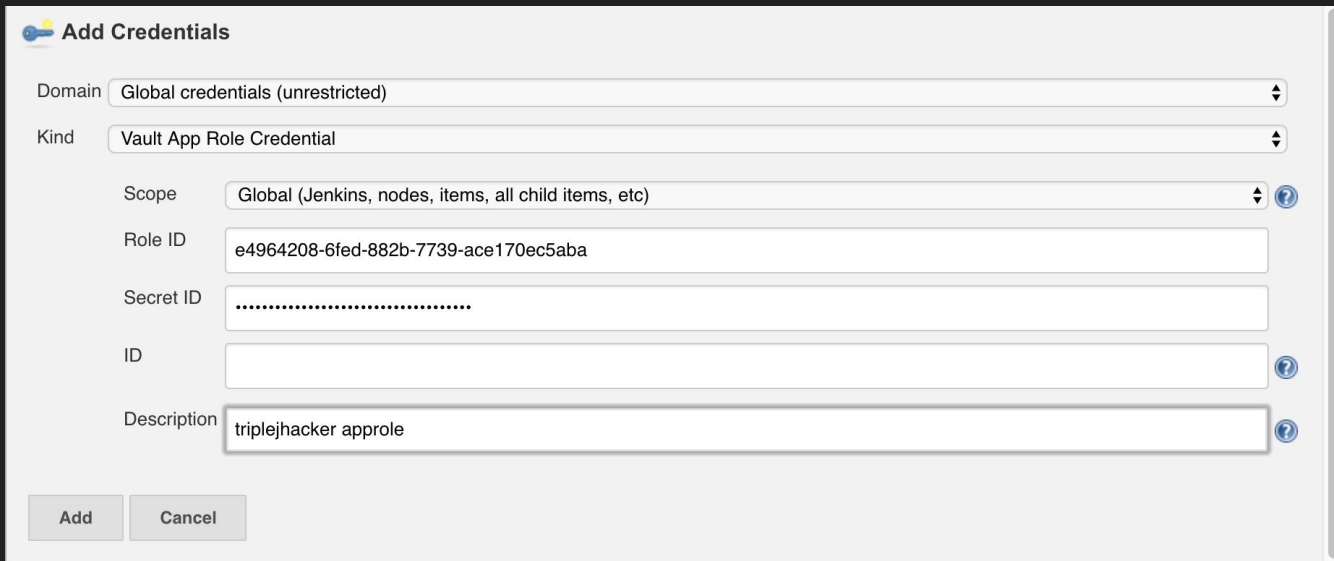
- Paste your vault token from the previous slide
- Using curl, or <https://www.getpostman.com/apps> or any request tool, to POST request:

```
curl --header "X-Vault-Token: <REPLACE WITH VAULT TOKEN>" --request POST  
http://13.228.110.97:8200/v1/auth/approle/role/example/secret-id
```

- Copy the **secret-id**
 - {"request_id":"a2bdb1d5-28d9-d7c8-da4e-94800ba496e3","lease_id":"","renewable":false,"lease_duration":0,"data":{"secret_id":"a6f53a92-96bd-9fc4-9d8a-*****","secret_id_accessor":"60e54752-c6d1-320f-574a-a1ee3f7a219b"},"wrap_info":null,"warnings":null,"auth":null}

Store secrets within Jenkins

- Choose “Vault App Role Credential” as kind
- Fill Role ID from previous, previous slide
- Fill Secret ID from previous slide
- Fill Description with your username (for easy identification)
- Click “Add”



The screenshot shows the 'Add Credentials' dialog in Jenkins. The 'Domain' is set to 'Global credentials (unrestricted)'. The 'Kind' is set to 'Vault App Role Credential'. The 'Scope' is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Role ID' is filled with 'e4964208-6fed-882b-7739-ace170ec5aba'. The 'Secret ID' field is filled with a series of dots. The 'ID' field is empty. The 'Description' is filled with 'triplejhacker approle'. There are 'Add' and 'Cancel' buttons at the bottom.

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Vault App Role Credential

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

Role ID: e4964208-6fed-882b-7739-ace170ec5aba

Secret ID:

ID:



Description: triplejhacker approle

Add Cancel

Store secrets within Jenkins

- Choose your newly created item as the Vault Credential i.e. “triplejhacker approle”
- Click “Add a vault secret”

Vault Plugin

Vault URL	<input type="text" value="http://172.18.0.2:8200"/>	
Vault Credential	<div><input type="text" value="triplejhacker approle"/></div>	<div> Add</div>
<div>Add a vault secret</div>		

Store secrets within Jenkins

- Fill Environment Variable as “mongodb” (all small caps)
- Fill the rest as illustrated matching from the Vault UI
- Click “Save”

The diagram illustrates the configuration of a Jenkins environment variable to store secrets from HashiCorp Vault. It consists of two main parts: a screenshot of the Vault UI on the right and a screenshot of the Jenkins configuration page on the left.

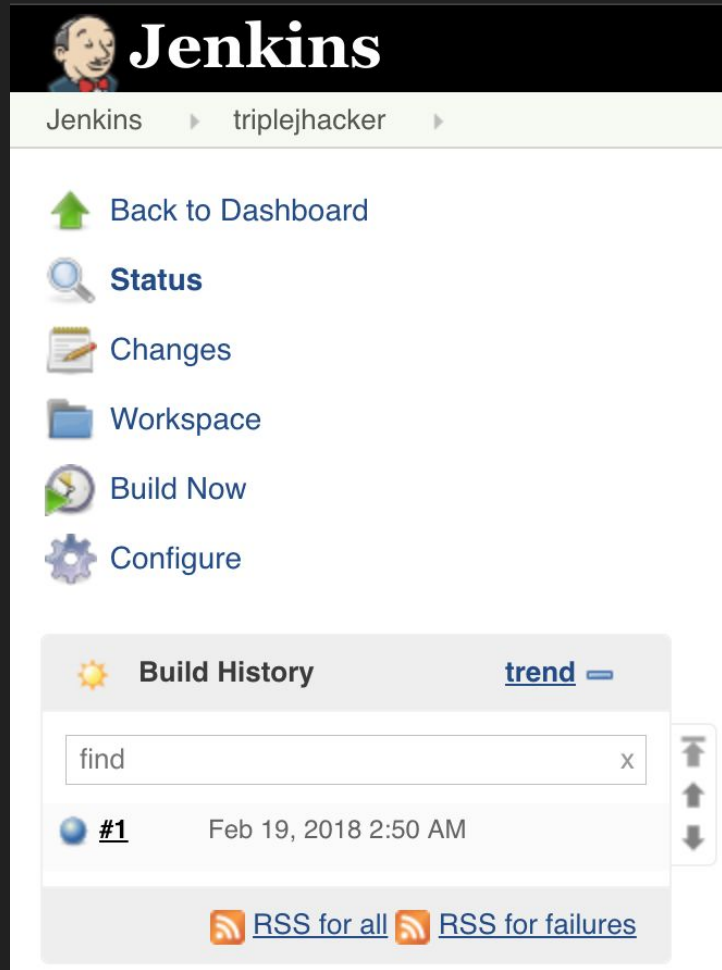
Vault UI (Right): The 'Vault Secret' page shows a 'Path' of `secret/example/triplejhacker`. Below this, the 'Environment Variable' is set to `mongodb`. The 'Key name' is also set to `mongodb`. Both fields include explanatory text: 'The environment variable to set with the value of the vault key. (from HashiCorp Vault Plugin)' and 'The vault key whose value with populate the environment variable. (from HashiCorp Vault Plugin)'.

Jenkins Configuration (Left): The 'Editing secret/example/triplejhacker' page shows a tree view with a folder `secret/example/triplejhacker {1}` and a key `mongodb : mongodb://test:testtest@...`. A yellow arrow points from the 'Path' field in the Vault UI to the folder name in the Jenkins tree. Another yellow arrow points from the 'Key name' field in the Vault UI to the key name `mongodb` in the Jenkins configuration.

Annotations: A yellow arrow points to the folder name in the Jenkins tree with the text 'Path (secret/example/triplejhacker)'. Another yellow arrow points to the key name in the Jenkins configuration with the text 'Key Name (mongodb)'.

Build and Run with Jenkins

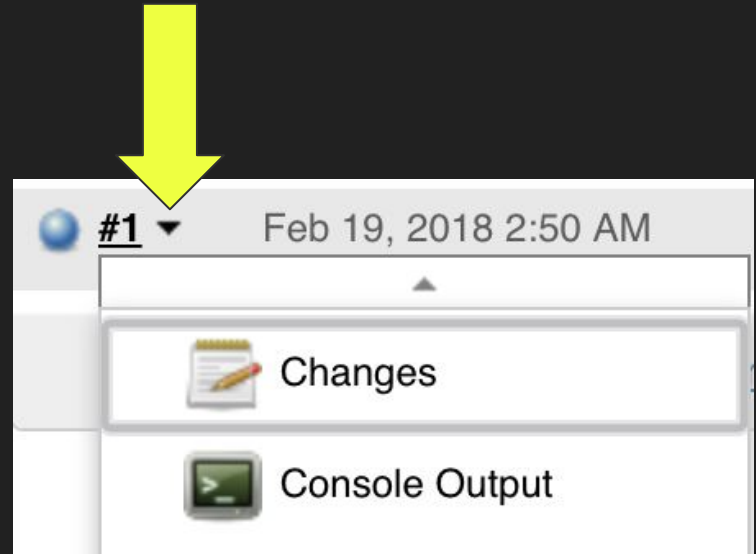
- Click “Build Now”
- Under “Build History”, there should be a build number like “#1” or “#2” or...



The screenshot displays the Jenkins web interface. At the top, the Jenkins logo is visible next to the text "Jenkins". Below this, a breadcrumb trail shows "Jenkins" followed by a right arrow and "triplejhacker" followed by another right arrow. A sidebar on the left contains several icons and labels: a green upward arrow for "Back to Dashboard", a magnifying glass for "Status", a notepad and pencil for "Changes", a folder for "Workspace", a clock with a green arrow for "Build Now", and a gear for "Configure". The main content area features a "Build History" section with a sun icon and a "trend" link with a minus sign. Below this is a search bar containing the text "find" and a close button "x". A table lists build history entries, with the first entry showing a blue circle icon, the build number "#1", and the timestamp "Feb 19, 2018 2:50 AM". At the bottom of the section, there are two RSS feed links: "RSS for all" and "RSS for failures", each preceded by an RSS icon. On the far right, a vertical scrollbar with up, down, and refresh arrows is visible.

Build and Run with Jenkins

- Click on the arrow beside the build number (shown here)
- Click on “Console Output” to show the logs from the build



Build and Run with Jenkins

- If all goes well, it should be pulling from your latest code commit, check the commit message.
 - It should be the same as the value in “`git commit -m 'remove secret'`” ran earlier
- We didn't push any code changes so this is correct.

Jenkins ▶ triplejhacker ▶ #4

Commit message: "remove secret"

> git rev-list --no-walk 5518845d

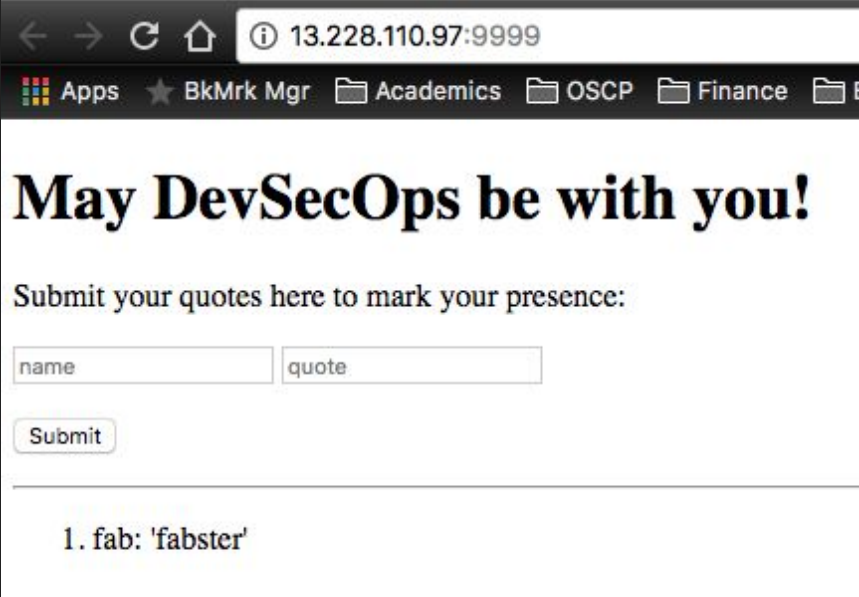
Build and Run with Jenkins

- If all goes well, it should look something like this ending with “**Finished: SUCCESS**”
- Note that the secret is also masked out. Good job Jenkins!

```
+ sudo docker run --network=isolated_nw -d -p 9999:3000 --name triplejhacker --env MONGODB_CREDENTIALS=****  
triplejhacker/devseccon2018  
2704aa320979f1c615df5db102fac4e0242bc5c6fbabada54c3809bbc4ba08a4  
Finished: SUCCESS
```

Access your deployed app

- Append the port number you specified in jenkins to http://13.228.110.97:<specified_port> and go to this URL in your browser
- In my example, I go to <http://13.228.110.97:9999>
- A simple app should still display
- Interact with the app by adding quotes
- These quotes are stored in your mongodb in mlab. You can go back to mlab and check the changes in database



The screenshot shows a web browser window with the address bar displaying `13.228.110.97:9999`. The browser's bookmark bar includes links for 'Apps', 'BkMrk Mgr', 'Academics', 'OSCP', 'Finance', and 'E'. The main content area features a large heading **May DevSecOps be with you!** followed by the instruction 'Submit your quotes here to mark your presence:'. Below this is a form with two input fields labeled 'name' and 'quote', and a 'Submit' button. At the bottom, a list of quotes is shown, starting with '1. fab: 'fabster''.

13.228.110.97:9999

Apps ★ BkMrk Mgr Academics OSCP Finance E

May DevSecOps be with you!

Submit your quotes here to mark your presence:

1. fab: 'fabster'

Debug your deployed app

- Docker UI is at <http://13.228.110.97:8100>
- Login with username and password “readonly” to view the state and logs of containers

The screenshot displays the Portainer.io web interface. On the left is a dark blue sidebar with the Portainer.io logo and navigation links: ACTIVE ENDPOINT (set to 'local'), ENDPOINT ACTIONS (Dashboard, App Templates, Containers, Images, Networks), and a user profile section. The main content area is titled 'Container list' and shows a table of containers. The first container, 'triplejhacker', is in a 'running' state. Above the table are various action buttons (Start, Stop, Kill, Restart, Pause, Resume, Remove) and an 'Add container' button. The top right of the interface includes links for help, support, and user account management.

portainer.io

ACTIVE ENDPOINT

local

ENDPOINT ACTIONS

- Dashboard
- App Templates
- Containers
- Images
- Networks

Container list

Containers

Help support portainer | readonly | my account | log out

Containers

Q Search ⚙ Settings

Start Stop Kill Restart Pause Resume Remove + Add container

<input type="checkbox"/>	Name	State <small>↓</small> Filter	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
<input type="checkbox"/>	triplejhacker	running		-	triplejhacker/devseccon2018	172.18.0.5	9999:3000	public

Congrats!

**You have just removed a shameful
secret AND use App Role to control
access to the secrets!**

But is this good enough?

<https://www.vaultproject.io/>

**Solution 3 - Remove Secrets from App
and Jenkins, using App Role and +++**

... until next time...

<https://www.vaultproject.io/docs/concepts/response-wrapping.html>

Or try

[https://medium.com/what-about-security/all-day-devops-2017-removing-dev-
elopers-shameful-secrets-f5aca3960316](https://medium.com/what-about-security/all-day-devops-2017-removing-dev-
elopers-shameful-secrets-f5aca3960316)

**Thank you for your attention, patience, and
enthusiasm during the workshop!**

**Happy Lunar New Year!
Cheers!**



Join the conversation

#DevSecCon