



**CS³
MESH⁴
EOSC**

Connecting European Data



Driving ScienceBox into the future

Samuel Alfageme Sainz
Enrico Bocchi
Jimil Desai



CS3MESH4EOSC has received funding from the European Union's Horizon 2020 Research and Innovation programme under **Grant Agreement No. 863353**.



 **jimil749**



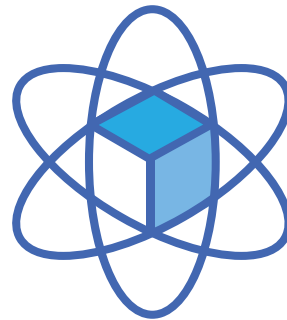
 **ebocchi**



 **SamuAlfageme**

But also deserve a mention: Aritz Brosa, Fabio Luchetti, Diogo Castro

The demonstrator package for some of CERN's storage and analysis services:



*We present this update as a **self-contained and easy-to-use package with minimal dependencies** and with the same goals as the original ScienceBox: Provide a sandbox to evaluate the storage, sharing, and analysis services we run at CERN on external premises to non-CERN users.*

*We believe there is not only a great value in releasing and contributing back to the open source projects that sustain these services, but also in **describing the configuration and artifacts that make operating such complex software systems at scale possible.***

EPJ Web of Conferences **245**, 07047 (2020)
CHEP 2019

<https://doi.org/10.1051/epjconf/202024507047>

ScienceBox

Converging to Kubernetes containers in production for on-premise and hybrid clouds for CERNBox, SWAN, and EOS

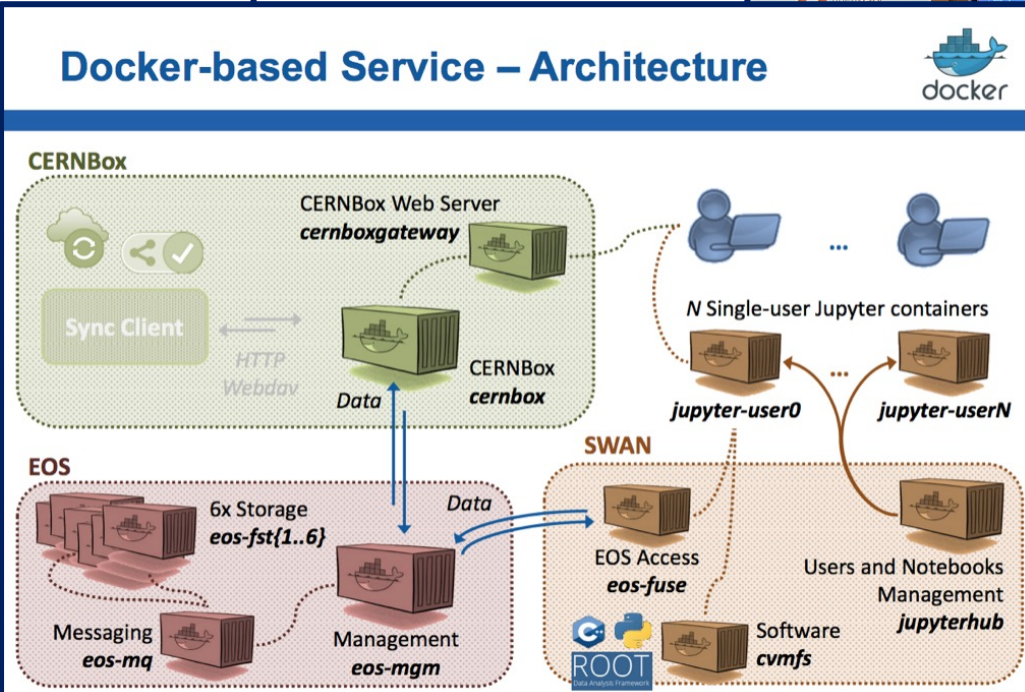
Enrico Bocchi^{1,*}, Luca Canali¹, Diogo Castro¹, Prasanth Kothur Labrador¹, Maciej Malawski², Jakub T. Mościcki¹, and Piotr Mrowczyński¹

¹CERN, Esplanade des Particules 1, 1211 Geneva 23, Switzerland

²AGH University of Science and Technology, Aleja Adama Mickiewicza 30, 30-059 Kraków, Poland

Abstract. Docker containers are the de-facto standard to package and run applications on cloud-based infrastructures. Commercial private clouds expand their offer with container orchestration engines, and the management of resources and containerized applications tightly. The Storage Group of CERN IT leverages on container technologies to create ScienceBox: An integrated software bundle with storage and computing services for general purposes and scientific use. ScienceBox features scalable storage, sync&share functionalities, and a web-based data access service, and can be deployed on a single machine or scaled-out across servers. ScienceBox has proven to be helpful in different contexts, from Energy Physics analysis to education for high schools, and has been fully deployed on different cloud infrastructure and heterogeneous hardware.

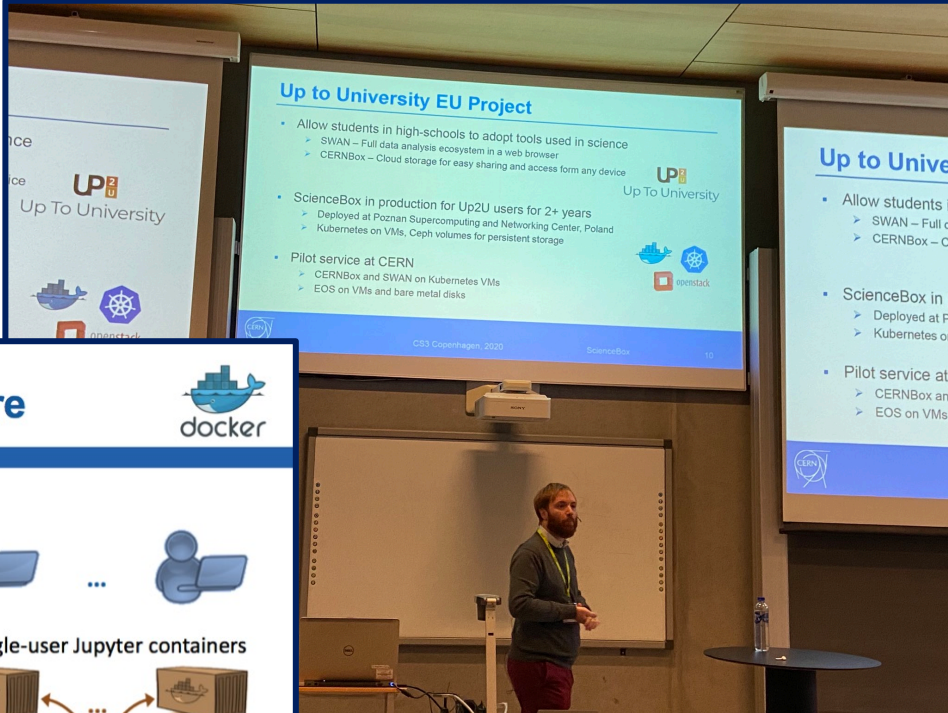
Docker-based Service – Architecture



The diagram illustrates the architecture of a Docker-based service. It is divided into three main sections: CERNBox, EOS, and SWAN.

- CERNBox:** Contains a Sync Client, CERNBox Web Server (cernboxgateway), and CERNBox (cernbox). The Sync Client and Web Server interact via HTTP Webdav. Data is stored in CERNBox.
- EOS:** Includes 6x Storage (eos-fst{1..6}), Messaging (eos-mq), and Management (eos-mgm). Data is shared between CERNBox and EOS.
- SWAN:** Features EOS Access (eos-fuse), Users and Notebooks Management (jupyterhub), and Software (cvmfs). It supports N Single-user Jupyter containers (jupyter-user0 to jupyter-userN).

Users interact with the system through the CERNBox Web Server and the Jupyter containers. The diagram also shows the Docker logo in the top right corner.



The photograph shows a presentation slide titled "Up to University EU Project". The slide lists several bullet points:

- Allow students in high-schools to adopt tools used in science
 - SWAN – Full data analysis ecosystem in a web browser
 - CERNBox – Cloud storage for easy sharing and access from any device
- ScienceBox in production for Up2U users for 2+ years
 - Deployed at Poznan Supercomputing and Networking Center, Poland
 - Kubernetes on VMs, Ceph volumes for persistent storage
- Pilot service at CERN
 - CERNBox and SWAN on Kubernetes VMs
 - EOS on VMs and bare metal disks

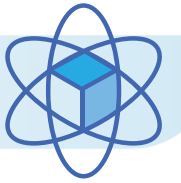
The slide also features logos for Up To University, ScienceBox, SWAN, and EOSC.



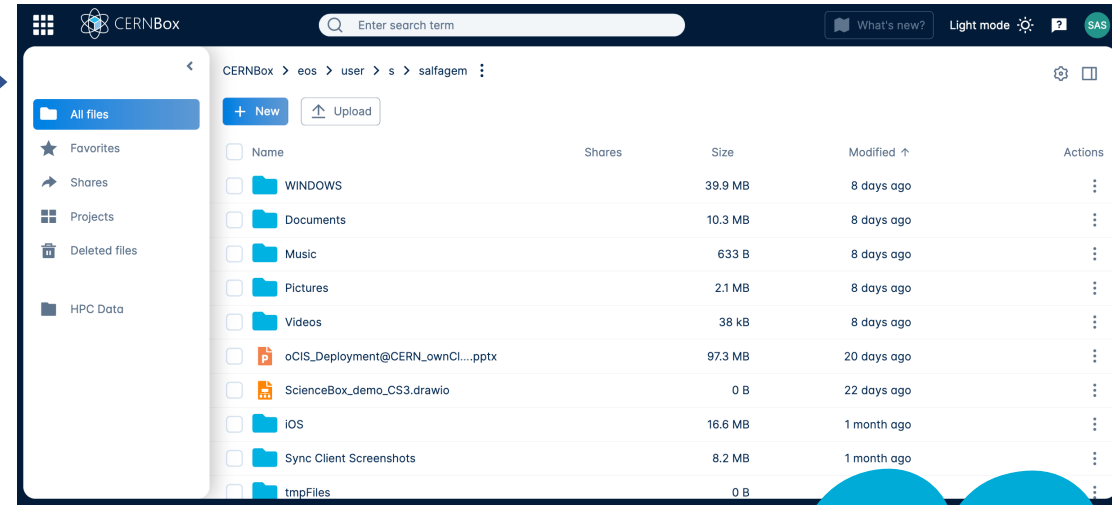
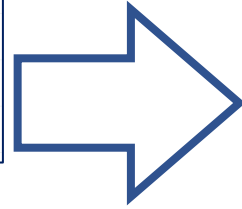
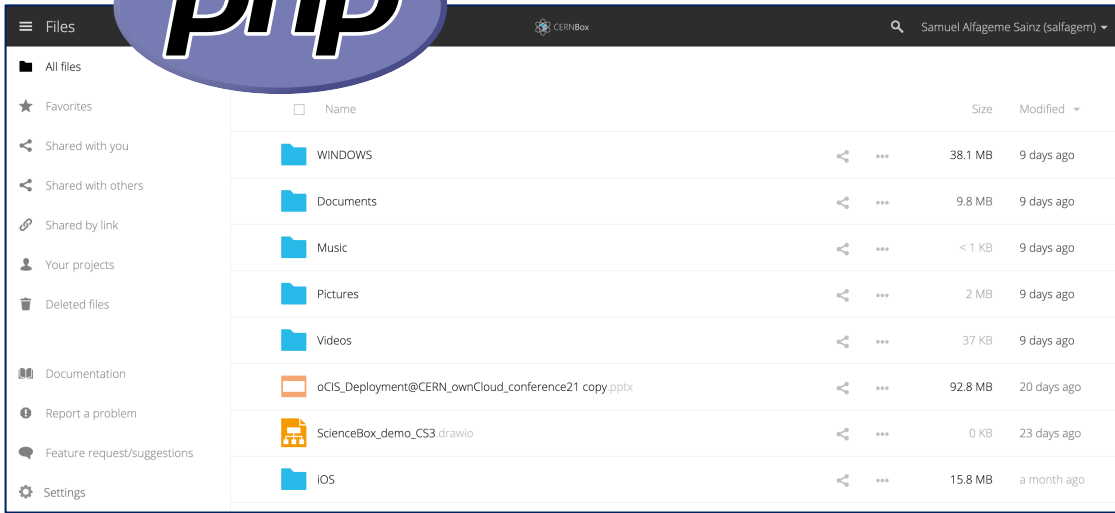
4 Conclusions and Future Outlook

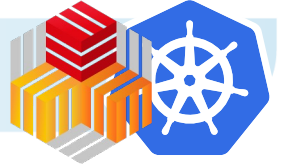
Also, we foresee ScienceBox to evolve into a modular and fully customizable bundle where each service component can be deployed independently from the others through Helm charts. This provides all-round configuration flexibility and the ability to install on-premises the complete stack of ScienceBox services or only a subset of them. In this scenario, the ease of deployment provided by container technologies and the modular architecture of ScienceBox would foster the distribution of open-source scientific software across multiple institutions to increase the interoperability beyond the borders of single clouds.

Lastly, ScienceBox has been chosen as the reference deployment model for CS3MESH4EOSC [16], a European project that aims at creating a federation of data and



php





<https://gitlab.cern.ch/eos/eos-on-k8s>

Instantiating a test

1. Create a K8s Namespace
2. Create all K8s Services, Pods (1 or more per EOS role)
 - MGM, MQ, QDB, KRBS, FSTs, CLIENTs
3. Configure each Pod
 - "Patch" running containers to operate in the new network space
4. Run the test you like most

create-all.sh
script

ref: <https://gitlab.cern.ch/eos/eos-on-k8s>

*From:
"Testing EOS on Kubernetes: CI and beyond"
4th EOS Workshop - 2020*

Harbor | English | Samuel_Alfageme_Sainz

< Projects < eos < Versions

server:0.1.6 | Project Admin

Summary | Dependencies | Values

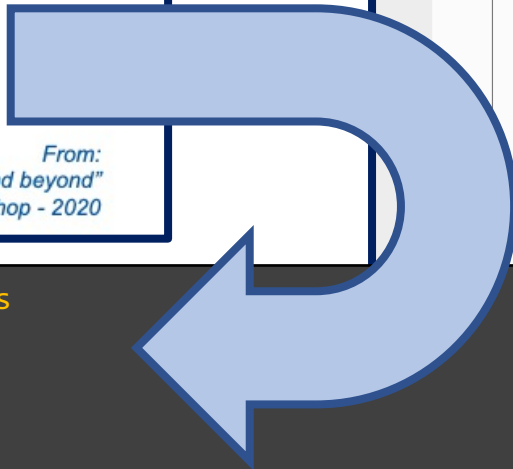
A fully-fledge EOS server instance with qdb, mgm, mq, and fst

server

A fully-fledge EOS server instance with qdb, mgm, mq, and fst

Version 0.1.5 | Type application | AppVersion 5.0.31

Requirements



```
helm repo add eos https://registry.cern.ch/chartrepo/eos
helm repo update
helm install eos eos/server

[...]

kubectl exec eos-mgm-0 -- eos fs ls
```

host	port	id	path	schedgroup	geotag	boot	configstatus	drain	active	health
eos-fst-3.eos-fst.ocis.svc.cluster.local	1095	1	/fst_storage	default.0	docker::k8s	booted	rw	nodrain	online	N/A
eos-fst-2.eos-fst.ocis.svc.cluster.local	1095	2	/fst_storage	default.1	docker::k8s	booted	rw	nodrain	online	N/A
eos-fst-1.eos-fst.ocis.svc.cluster.local	1095	3	/fst_storage	default.2	docker::k8s	booted	rw	nodrain	online	N/A
eos-fst-0.eos-fst.ocis.svc.cluster.local	1095	4	/fst_storage	default.3	docker::k8s	booted	rw	nodrain	online	N/A

SO, LONG
STORY
SHORT



WWYD ?

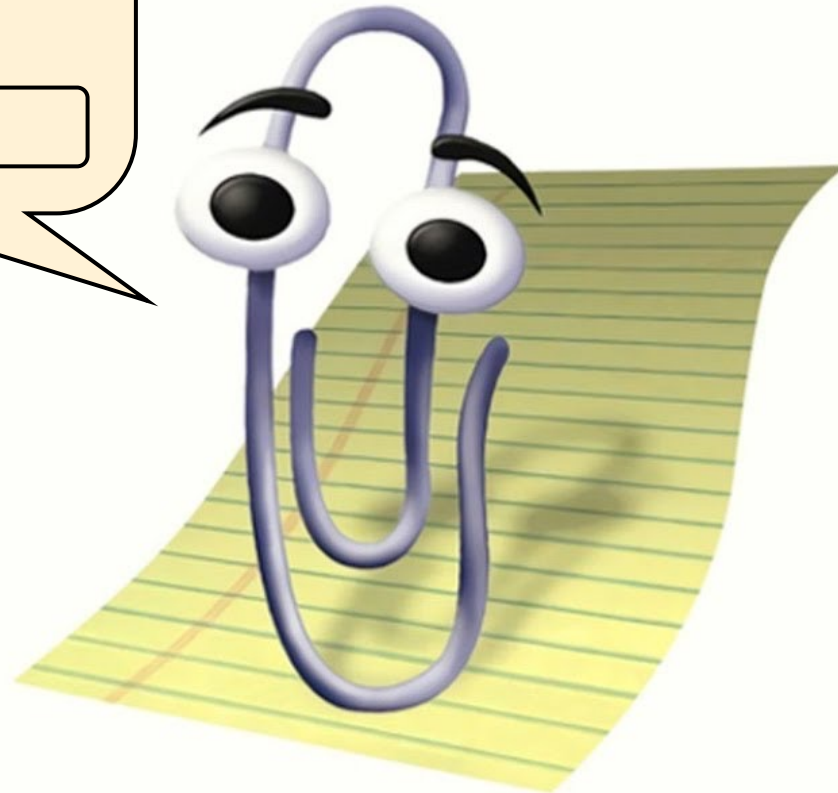


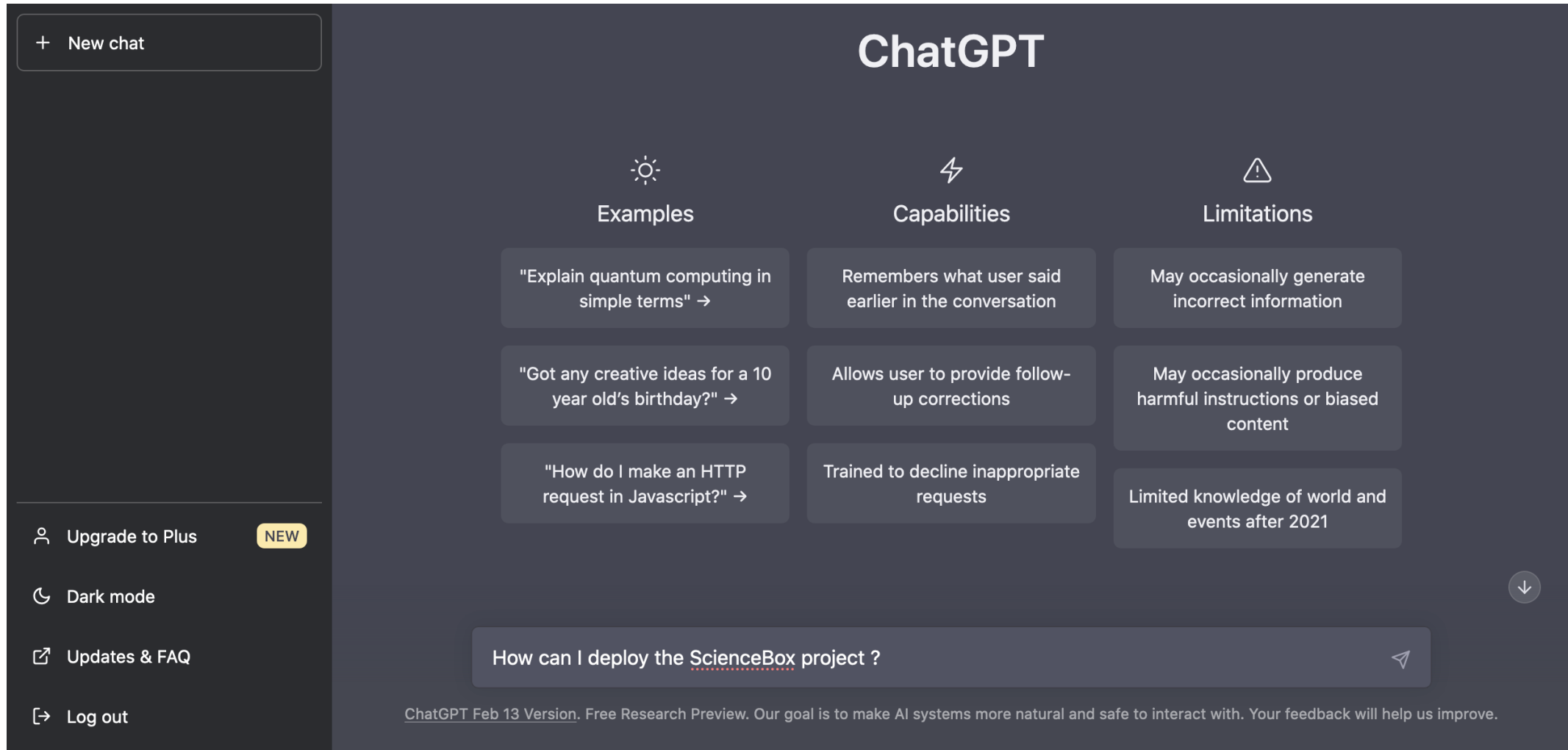
It looks like you're trying to
deploy ScienceBox today.

Would you like help?

Yes

No





The screenshot shows the ChatGPT web interface. On the left is a dark sidebar with a '+ New chat' button at the top. Below it are menu items: 'Upgrade to Plus' (with a 'NEW' badge), 'Dark mode', 'Updates & FAQ', and 'Log out'. The main content area is titled 'ChatGPT' and features a 3x3 grid of cards. The first column, 'Examples', shows prompts like 'Explain quantum computing in simple terms' and 'Got any creative ideas for a 10 year old's birthday?'. The second column, 'Capabilities', lists features like 'Remembers what user said earlier in the conversation' and 'Allows user to provide follow-up corrections'. The third column, 'Limitations', lists drawbacks such as 'May occasionally generate incorrect information' and 'Limited knowledge of world and events after 2021'. At the bottom of the main area is a chat input field with the text 'How can I deploy the ScienceBox project?' and a send button. A footer note reads: 'ChatGPT Feb 13 Version. Free Research Preview. Our goal is to make AI systems more natural and safe to interact with. Your feedback will help us improve.'



How can I deploy the ScienceBox project then?




Deploying the ScienceBox project requires some technical expertise and infrastructure setup. Here are the general steps involved in deploying ScienceBox:






1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.
2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, GitLab, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.
3. Configure the components: Each component of ScienceBox will need to be configured to work together and to meet the specific needs of your deployment. This may involve setting up user accounts, configuring storage, and adjusting security settings.
4. Test the deployment: Once the components are installed and configured, you will need to test the deployment to ensure that everything is working as expected. This may involve creating test accounts, uploading data, and testing collaboration features.
5. Maintain and update the deployment: ScienceBox is a complex system that requires ongoing maintenance and updates to ensure that it remains secure and functional. You will need to regularly update the software components, monitor for security vulnerabilities, and respond to user feedback and requests.

Overall, deploying ScienceBox requires significant technical expertise and infrastructure resources. If you are interested in deploying ScienceBox, it may be helpful to consult with an experienced IT professional or seek guidance from the ScienceBox community.


Regenerate response

 How can I deploy the ScienceBox project then?

 Deploying the ScienceBox project requires some technical expertise and infrastructure setup. Here are the general steps involved in deploying ScienceBox:  

1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.
2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, GitLab, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.
3. Configure the components: Each component of ScienceBox will need to be configured to work together and to meet the specific needs of your deployment. This may involve setting up user accounts, configuring storage, and adjusting security settings.
4. Test the deployment: Once the components are installed and configured, you will need to test the deployment to ensure that everything is working as expected. This may involve creating test accounts, uploading data, and testing collaboration features.
5. Maintain and update the deployment: ScienceBox is a complex system that requires ongoing maintenance and updates to ensure that it remains secure and functional. You will need to regularly update the software components, monitor for security vulnerabilities, and respond to user feedback and requests.

Overall, deploying ScienceBox requires significant technical expertise and infrastructure resources. If you are interested in deploying ScienceBox, it may be helpful to consult with an experienced IT professional or seek guidance from the ScienceBox community.

 Regenerate response

1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.

OK 



How can I deploy the ScienceBox project then?



Deploying the ScienceBox project requires some technical expertise and infrastructure setup. Here are the general steps involved in deploying ScienceBox:



1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.
2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, GitLab, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.
3. Configure the components: Each component of ScienceBox will need to be configured to work together and to meet the specific needs of your deployment. This may involve setting up user accounts, configuring storage, and adjusting security settings.
4. Test the deployment: Once the components are installed and configured, you will need to test the deployment to ensure that everything is working as expected. This may involve creating test accounts, uploading data, and testing collaboration features.
5. Maintain and update the deployment: ScienceBox is a complex system that requires ongoing maintenance and updates to ensure that it remains secure and functional. You will need to regularly update the software components, monitor for security vulnerabilities, and respond to user feedback and requests.

Overall, deploying ScienceBox requires significant technical expertise and infrastructure resources. If you are interested in deploying ScienceBox, it may be helpful to consult with an experienced IT professional or seek guidance from the ScienceBox community.

Regenerate response

1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.

2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, **GitLab**, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.

O... wait, what? 🤔



How can I deploy the ScienceBox project then?



Deploying the ScienceBox project requires some technical expertise and infrastructure setup. Here are the general steps involved in deploying ScienceBox:



1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.
2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, GitLab, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.
3. Configure the components: Each component of ScienceBox will need to be configured to work together and to meet the specific needs of your deployment. This may involve setting up user accounts, configuring storage, and adjusting security settings.
4. Test the deployment: Once the components are installed and configured, you will need to test the deployment to ensure that everything is working as expected. This may involve creating test accounts, uploading data, and testing collaboration features.
5. Maintain and update the deployment: ScienceBox is a complex system that requires ongoing maintenance and updates to ensure that it remains secure and functional. You will need to regularly update the software components, monitor for security vulnerabilities, and respond to user feedback and requests.

Overall, deploying ScienceBox requires significant technical expertise and infrastructure resources. If you are interested in deploying ScienceBox, it may be helpful to consult with an experienced IT professional or seek guidance from the ScienceBox community.

Regenerate response

1. Choose a deployment environment: ScienceBox can be deployed on-premises or in the cloud. You will need to choose an environment that can support the necessary infrastructure and storage requirements.

2. Install the required dependencies: ScienceBox relies on several open-source components, including OwnCloud, GitLab, and JupyterHub. You will need to install these components and their dependencies on your deployment environment.

3. Configure the components: Each component of ScienceBox will need to be configured to work together and to meet the specific needs of your deployment. This may involve setting up user accounts, configuring storage, and adjusting security settings.



Way too hard 4 me!

```
[root@mboxed-demo-cs3-centos ~]# git clone https://github.com/sciencebox/mboxed.git && cd mboxed
[...]  
[root@mboxed-demo-cs3-centos ~]# ./SetupInstall.sh  
[...]  
[root@mboxed-demo-cs3-centos ~]# ./ScienceBox.sh  
[...]  
Release "sciencebox" does not exist. Installing it now.  
NAME: sciencebox  
LAST DEPLOYED: Thu Mar  2 16:28:26 2023  
NAMESPACE: default  
STATUS: deployed  
REVISION: 1  
TEST SUITE: None
```

ScienceBox is being installed!

Now it is a good time to grab a coffee...

The deployment should be ready in a few minutes.

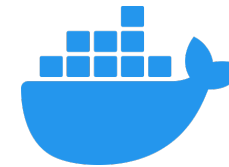
You can check the containers status by typing `kubectl get pods`.

Once all the containers are running (or completed), you can reach ScienceBox from your browser at <https://mboxed-demo-cs3-centos.cern.ch/sciencebox>

Hard dependencies



CentOS

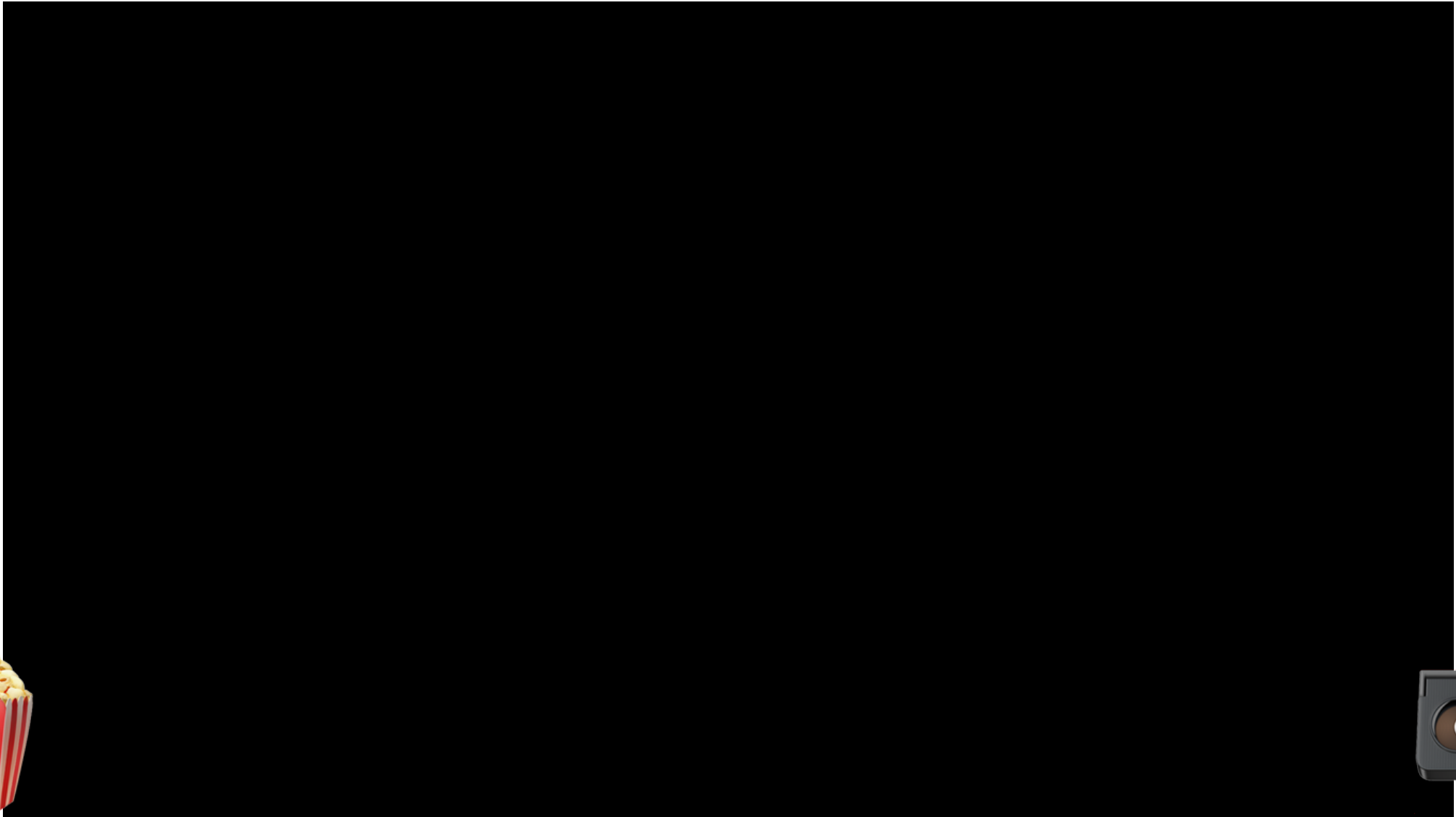


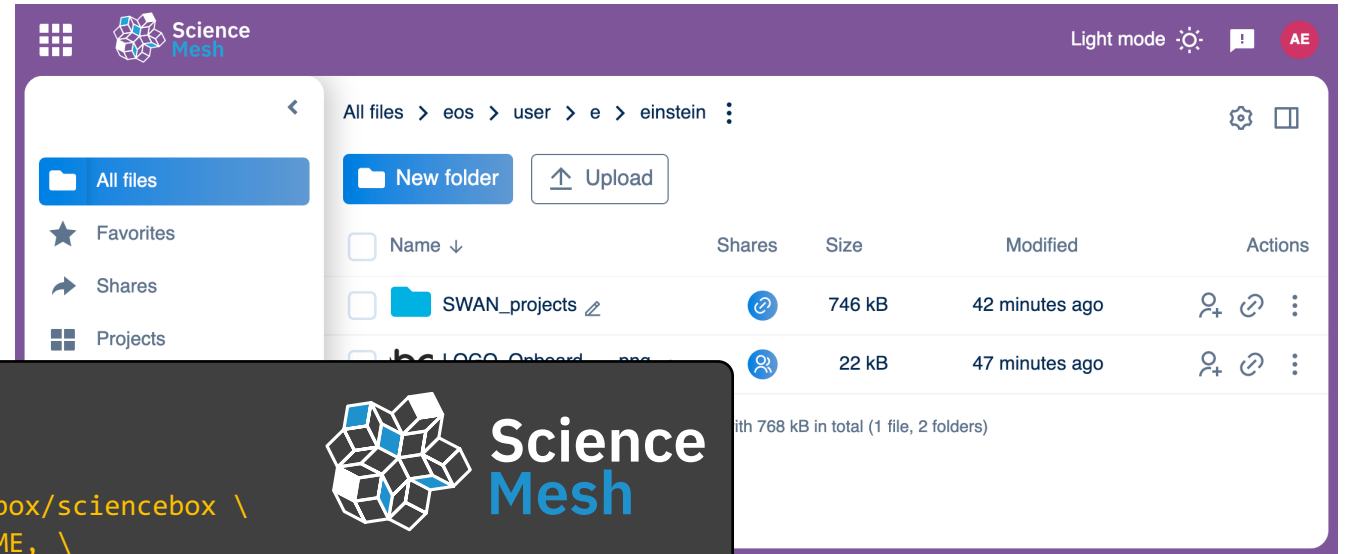
docker



minikube

Versions Fixed





Change the theme:

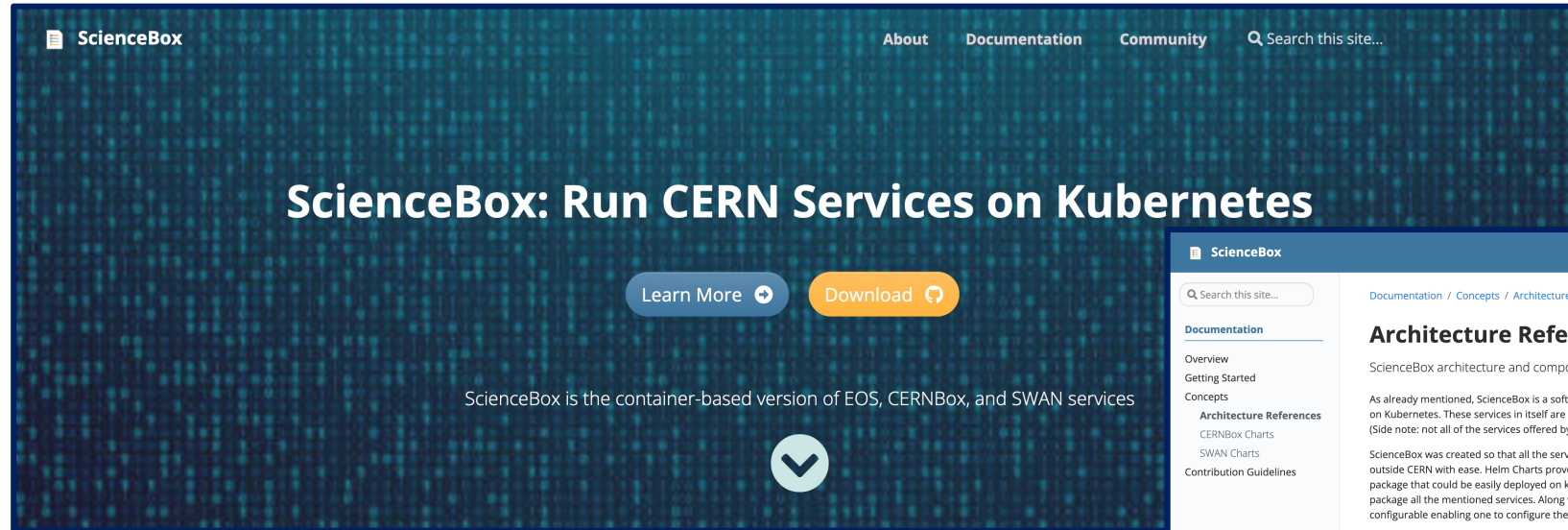
```
[root@mboxed-demo-cs3-centos ~]# helm upgrade -i sciencebox sciencebox/sciencebox \
--set cernbox-web.initContainers[0].env[1].name=WEB_THEME, \
cernbox-web.initContainers[0].env[1].value=https://alfage.me/statics/theme-sciencebox.tar.gz
```



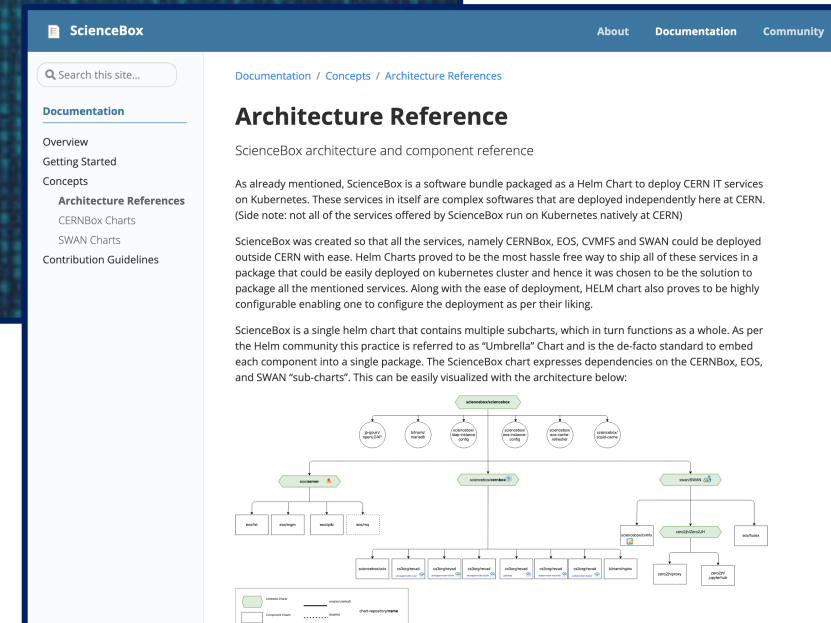
Configure KeyCloak as SSO:

```
[root@mboxed-demo-cs3-centos ~]# helm upgrade -i sciencebox sciencebox/sciencebox \
--set cernbox.ocis-idp.enable=false -f keycloak-values.yaml
```



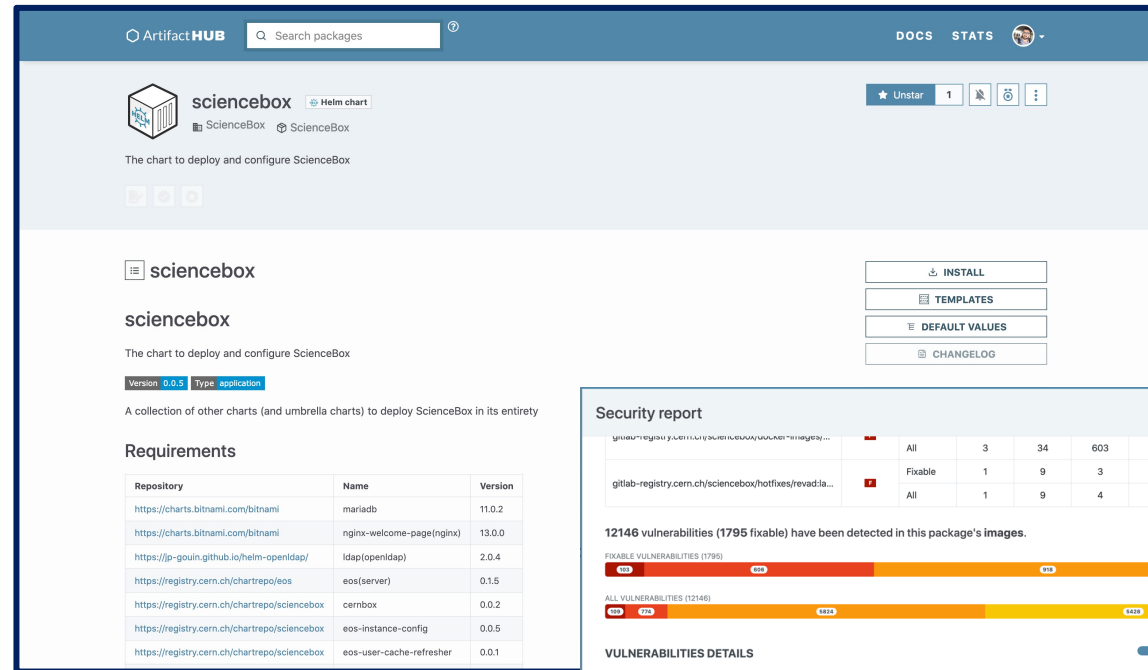


New cern.ch/sciencebox landing page & documentation



Reference of good practices for publishing Helm charts:

- # helm-docs generation
- # CI based in GH Actions
- # Integration with ArtifactHub
 - # Changelogs
 - # Security reports
 - # (...)



ArtifactHUB Search packages

sciencebox Helm chart

ScienceBox ScienceBox

The chart to deploy and configure ScienceBox

sciencebox

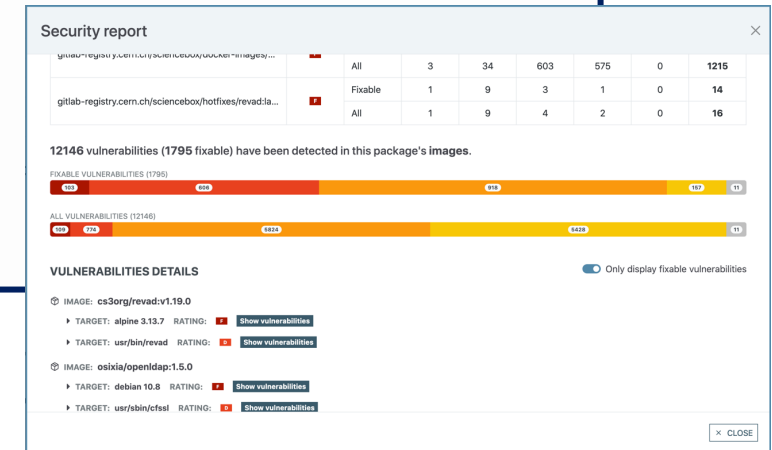
The chart to deploy and configure ScienceBox

Version 0.0.5 Type application

A collection of other charts (and umbrella charts) to deploy ScienceBox in its entirety

Requirements

Repository	Name	Version
https://charts.bitnami.com/bitnami	mariadb	11.0.2
https://charts.bitnami.com/bitnami	nginx-welcome-page/nginx	13.0.0
https://jp-gouin.github.io/helm-openldap/	ldap(openldap)	2.0.4
https://registry.cern.ch/chartrepo/eos	eos(server)	0.1.5
https://registry.cern.ch/chartrepo/sciencebox	cernbox	0.0.2
https://registry.cern.ch/chartrepo/sciencebox	eos-instance-config	0.0.5
https://registry.cern.ch/chartrepo/sciencebox	eos-user-cache-refresher	0.0.1



Security report

Image	All	3	34	603	575	0	1215
gitlab-registry.cern.ch/sciencebox/hotfixes/revad:ia...	Fixable	1	9	3	1	0	14
	All	1	9	4	2	0	16

12146 vulnerabilities (1795 fixable) have been detected in this package's images.

FIXABLE VULNERABILITIES (1795)

ALL VULNERABILITIES (12146)

VULNERABILITIES DETAILS

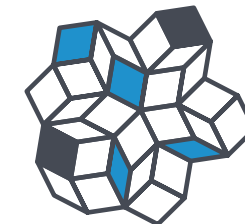
Only display fixable vulnerabilities

- IMAGE: cs3org/revad:v1.19.0
 - TARGET: alpine 3.13.7 RATING: ■ Show vulnerabilities
 - TARGET: usr/bin/revad RATING: ■ Show vulnerabilities
- IMAGE: osixia/openldap:1.5.0
 - TARGET: debian 10.8 RATING: ■ Show vulnerabilities
 - TARGET: usr/sbin/ctsd RATING: ■ Show vulnerabilities

Reference ScienceMesh Site:

institutions to increase the interoperability beyond the orders of single clouds

Lastly, ScienceBox has been chosen as the reference deployment model for CS3MESH4EOSC [16], a European project that aims at creating a federation of data and



Science Mesh

Grow the community around ScienceBox

Example contributions *up-for-grabs*:

- # Contribute configuration for the OCM Invitation Workflow to the Charts
- # Idem with the WOPI Server and Collaborative Applications (@Giuseppe)
- # Replace the **SetupInstall** script with config management (Ansible)
- # Remove the dependencies on CentOS 7 and a specific minikube driver

Adopt it for your use-cases: prospection, CI, testing ...



Q&A

 samuel.alfageme.sainz@cern.ch

 samuel@alfage.me



**CS³
MESH⁴
EOSC**

Connecting European Data

Thank you!
Discover more on...

 [cs3mesh4eosc.eu](https://www.cs3mesh4eosc.eu)

 [company/cs3mesh4eosc](https://www.linkedin.com/company/cs3mesh4eosc)

 [CS3org](https://twitter.com/CS3org)

 [CS3MESH4EOSC Project](https://www.youtube.com/channel/UCHKcZEKmqXjCvc3MLFjFxbw)

<https://www.youtube.com/channel/UCHKcZEKmqXjCvc3MLFjFxbw>



CS3MESH4EOSC has received funding from the European Union's Horizon 2020 Research and Innovation programme under **Grant Agreement No. 863353**.