Kubernetes-Goat Walkthrough

Kubernetes-Goat Overview and Walkthrough

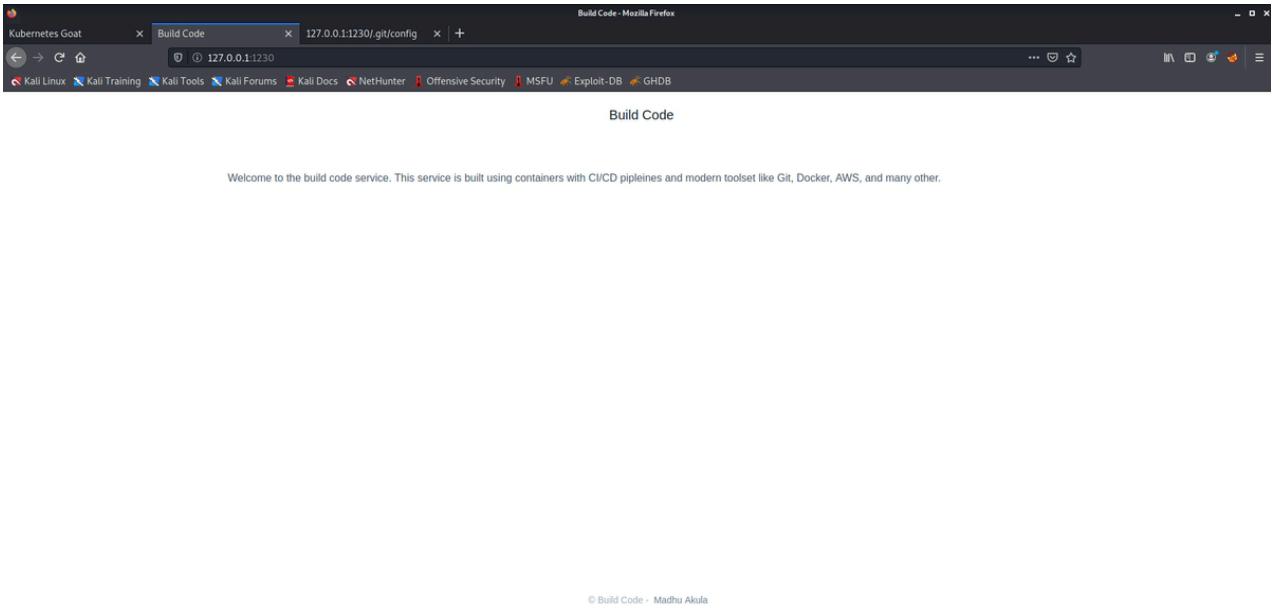
The following is the walkthrough of the Kubernetes-Goat project which is an intentional vulnerable Kubernetes cluster you can setup in your own environment. It is assumed that you have already followed the installation instructions on the ReadMe of the GitHub project and have the vulnerable Kubernetes cluster already running locally. Furthermore, throughout each section of the walkthrough an in depth analysis and review will be given along with references to supplemental documentation for understanding.

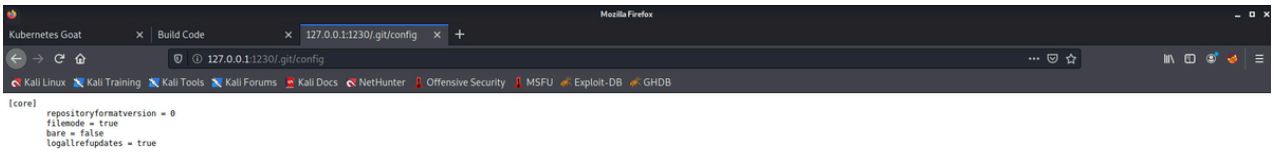
Sensitive Keys in Code Bases

On the initial Kubernetes-Goat setup it is mentioned that ports 1230-6 are open to us over localhost which we can investigate further using a browser or curl client.

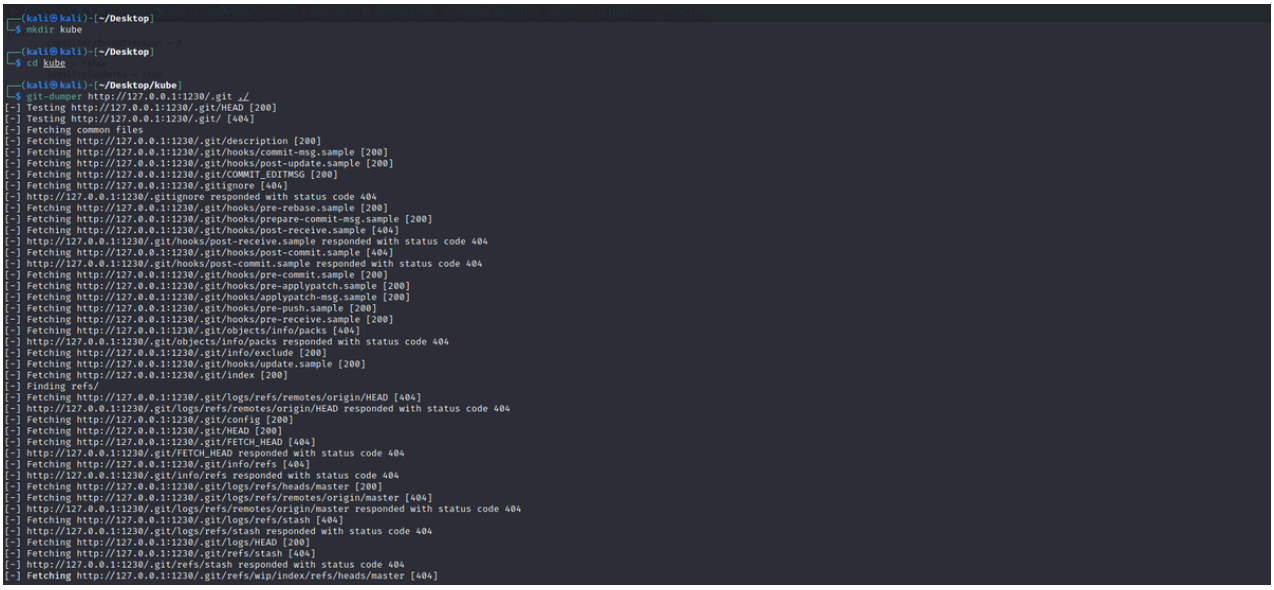
```
┌──(kali㉿kali)-[~/Desktop/kubernetes-goat]
└─$ bash access-kubernetes-goat.sh
kubectl setup looks good.
Creating port forward for all the Kubernetes Goat resources to locally. We will be using 1230 to 1236 ports locally!
Visit http://127.0.0.1:1234 to get started with your Kuberenetes Goat hacking!
```

Furthermore, when we look further at our local Kubernetes cluster on one of the open ports (port 1230) we are able to see more information to enumerate.

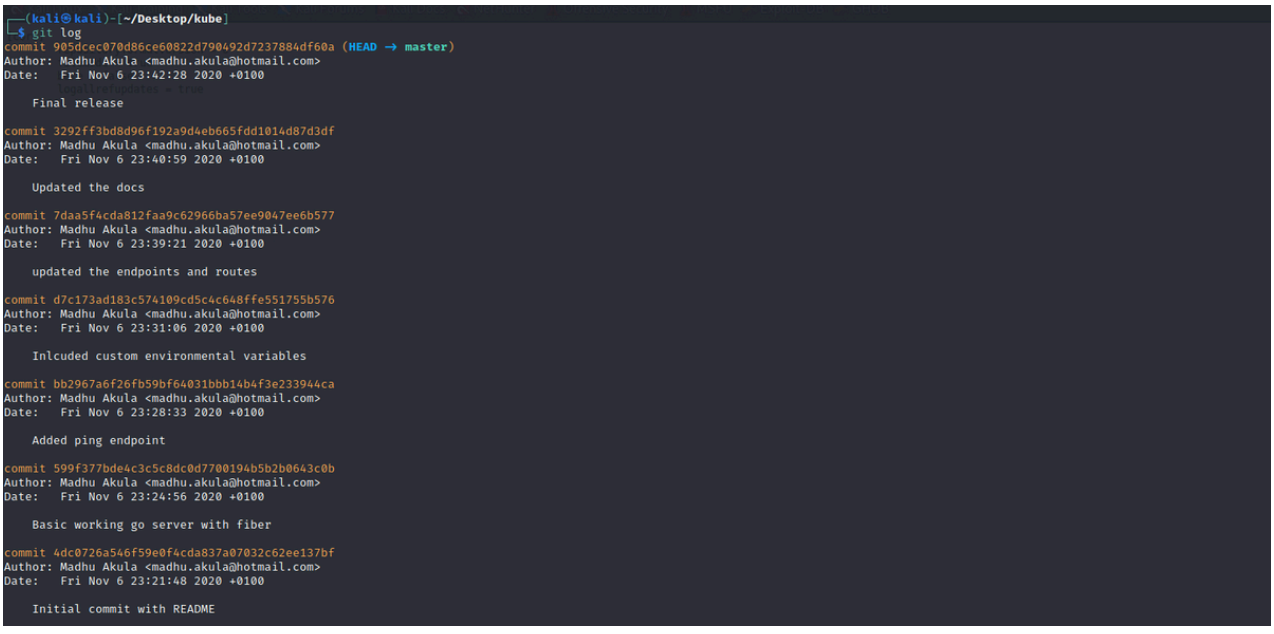




Upon enumeration a GitHub repository is found therefore git-dumper is used to download this found repository from the Kubernetes cluster into our own local directory to investigate further.



Important to realize, once a GitHub repository has been downloaded we can run a `git log` command to show the commit logs of that GitHub repository. Not to mention, once we have a given amount of commit numbers we can checkout each individual commit in the repository with the `git checkout <commit number>` command in order to enumerate for possible improper password use or any other security misconfigurations.

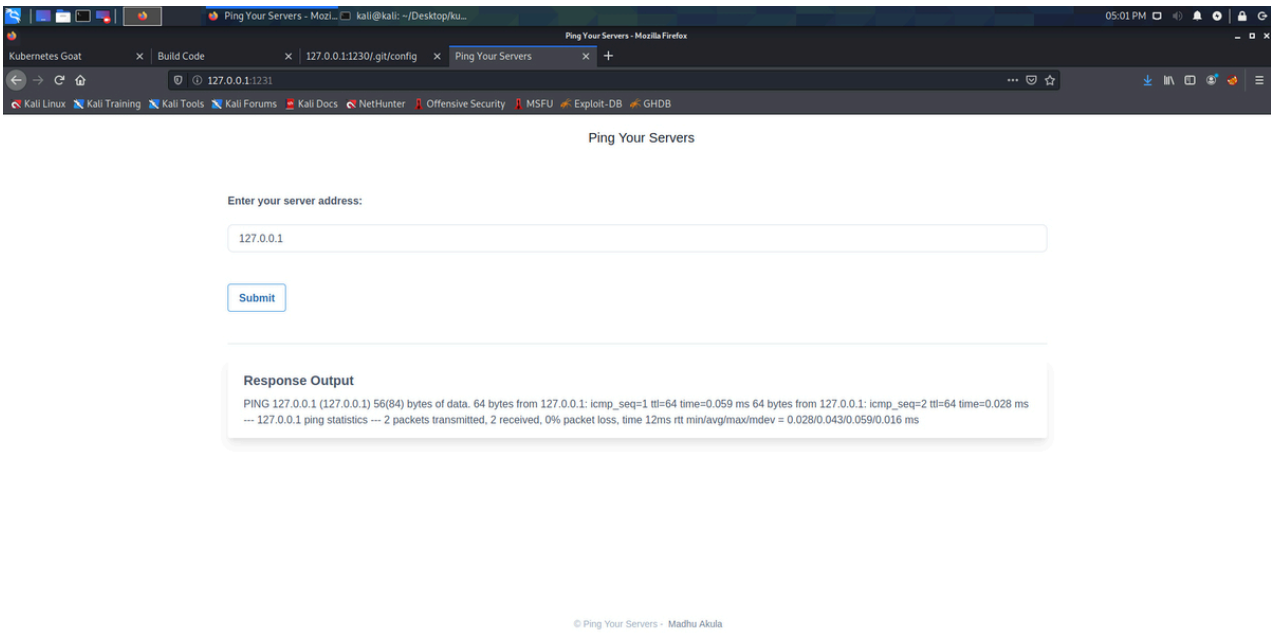


```
┌──(kali㉿kali)-[~/Desktop/kube]
└─$ git checkout d7c173ad183c574109cd5c4c648ffe551755b576
Note: switching to 'd7c173ad183c574109cd5c4c648ffe551755b576'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at d7c173a Inlcuded custom environmental variables

┌──(kali㉿kali)-[~/Desktop/kube]
└─$ ls -la
total 32
drwxr-xr-x 3 kali docker 4096 Aug 17 12:42 .
drwxr-xr-x 7 kali kali   4096 Aug 17 12:40 ..
-rw-r--r-- 1 kali docker  182 Aug 17 12:42 .env
drwxr-xr-x 7 kali docker 4096 Aug 17 12:42 .git
-rw-r--r-- 1 kali docker   76 Aug 17 12:42 go.mod
-rw-r--r-- 1 kali docker 2432 Aug 17 12:42 go.sum
-rw-r--r-- 1 kali docker  284 Aug 17 12:42 main.go
-rw-r--r-- 1 kali docker   95 Aug 17 12:42 README.md

┌──(kali㉿kali)-[~/Desktop/kube]
└─$ cat .env
[build-code-aws]
aws_access_key_id = AKIVSHD6243H22G1KIDC
aws_secret_access_key = cgGn4+gDgnriogn4g+34ig4bg34g44gg4Dox7c1M
k8s_goat_flag = k8s-goat-51bc78332065561b0c99280f62510bcc
```

Upon enumeration we use the `git checkout d7c173ad183c574109cd5c4c648ffe551755b576` to look at an individual commit where we look at the contents of a .env, which is a text file primarily used for custom user environment variables, and find AWS Credentials for an AWS principal IAM user along with a Kubernetes Goat flag.

Docker in Docker Exploitation

Continuing to enumerate the Kubernetes cluster we move to the next port in the port range being used which brings us to a form where we are able to input any machine to ping and get a response as such with an example of pinging localhost.



Afterwards, an idea came up on possibly using adding onto the server address to make the form run multiple bash commands assuming our input string is just being concatenated and not being validated. To much surprise it becomes apparent that the form on port 1231 does allow multiple bash commands with the right characters.