Lab - Installing w3af in Kali Linux Using Docker

Overview

In this lab, the student will learn how to install and update the w3af (Web Application audit and attack framework) using Docker. w3af is a framework for auditing and exploitation of web applications. Starting with Kali 2017, w3af was no longer included as part of the default install.

Using Docker, we will be able to install w3af and all its dependencies without having to install any additions updates for Kali.

About docker

Docker is a computer program that performs operating-system-level virtualization also known as containerization. It was first released in 2013 and is developed by Docker, Inc.

Docker is used to run software packages called "containers". In a typical example use case, one container runs a web server and web application, while a second container runs a database server that is used by the web application. Containers are isolated from each other and use their own set of tools and libraries; they can communicate through well-defined channels. All containers use the same kernel and are therefore more lightweight than virtual machines. Containers are created from "images" which specify their precise contents. Images are often created by combining and modifying standard images downloaded from repositories.

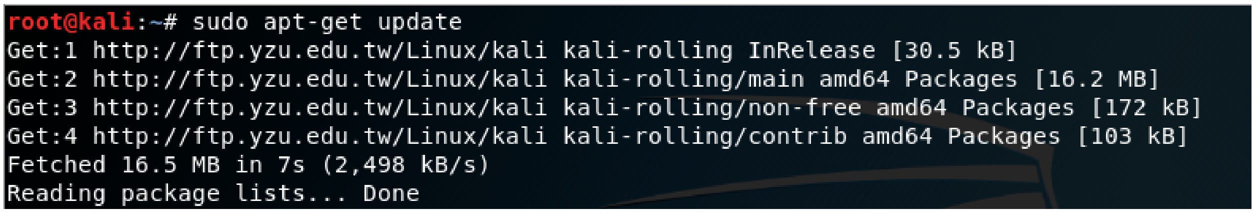
Requirements

- One virtual install of Kali Linux.
- Kali has been recently updated and upgraded with the latest packages.
- Internet connection

Begin the lab

Ensure Kali has been updated.

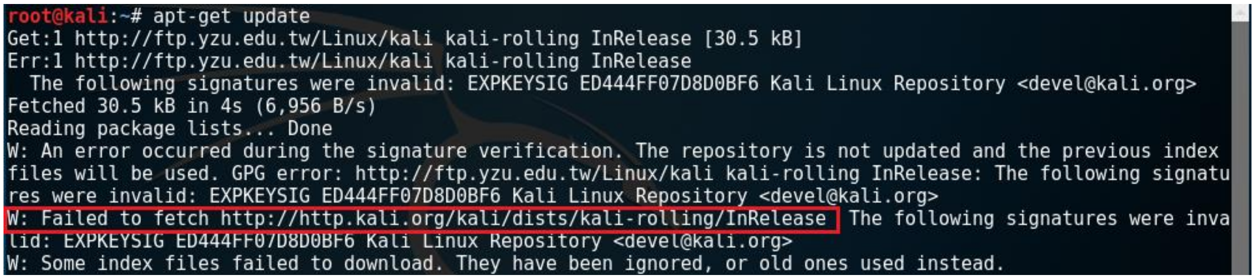
```
apt-get update
```



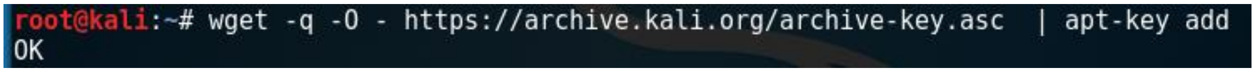
```
root@kali:~# sudo apt-get update
Get:1 http://ftp.yzu.edu.tw/Linux/kali kali-rolling InRelease [30.5 kB]
Get:2 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/main amd64 Packages [16.2 MB]
Get:3 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/non-free amd64 Packages [172 kB]
Get:4 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/contrib amd64 Packages [103 kB]
Fetched 16.5 MB in 7s (2,498 kB/s)
Reading package lists... Done
```

If you get error referencing an invalid key signature, you need to update your key signature using the following command:

```
wget -q -O - https://archive.kali.org/archive-key.asc  | apt-key add
```

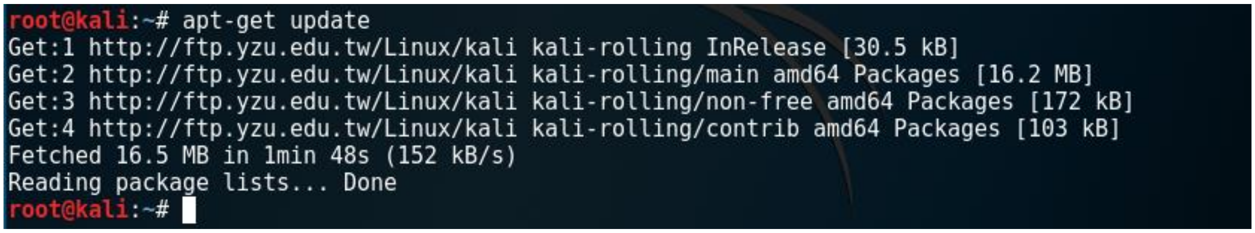


```
root@kali:~# apt-get update
Get:1 http://ftp.yzu.edu.tw/Linux/kali kali-rolling InRelease [30.5 kB]
Err:1 http://ftp.yzu.edu.tw/Linux/kali kali-rolling InRelease
  The following signatures were invalid: EXPKEYSIG ED444FF07D8D0BF6 Kali Linux Repository <devel@kali.org>
Fetched 30.5 kB in 4s (6,956 B/s)
Reading package lists... Done
W: An error occurred during the signature verification. The repository is not updated and the previous index
files will be used. GPG error: http://ftp.yzu.edu.tw/Linux/kali kali-rolling InRelease: The following signatu
res were invalid: EXPKEYSIG ED444FF07D8D0BF6 Kali Linux Repository <devel@kali.org>
W: Failed to fetch http://http.kali.org/kali/dists/kali-rolling/InRelease  The following signatures were inva
lid: EXPKEYSIG ED444FF07D8D0BF6 Kali Linux Repository <devel@kali.org>
W: Some index files failed to download. They have been ignored, or old ones used instead.
```



```
root@kali:~# wget -q -O - https://archive.kali.org/archive-key.asc  | apt-key add
OK
```

Run the update command once again.



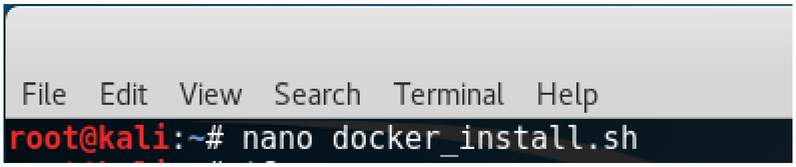
```
root@kali:~# apt-get update
Get:1 http://ftp.yzu.edu.tw/Linux/kali kali-rolling InRelease [30.5 kB]
Get:2 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/main amd64 Packages [16.2 MB]
Get:3 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/non-free amd64 Packages [172 kB]
Get:4 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/contrib amd64 Packages [103 kB]
Fetched 16.5 MB in 1min 48s (152 kB/s)
Reading package lists... Done
root@kali:~# 
```

Install the Docker Program

To install the Docker program, we will build a BASH script to automate the install.

Once Kali has been updated, at the terminal, type the following:

```
nano docker_install.sh
```



```
root@kali:~# nano docker_install.sh
```

This opens a blank text file using the nano text editor.

Copy and paste the following text between the dotted lines into the blank text editor.

The script is available at https://gist.github.com/apolloclark/f0e3974601346883c731

```
-------------------------Start of Script..................
```

```
#!/bin/bash

# update apt-get
export DEBIAN_FRONTEND="noninteractive"
sudo apt-get update

# remove previously installed Docker
sudo apt-get purge lxc-docker*
sudo apt-get purge docker.io*

# add Docker repo
sudo apt-get install -y apt-transport-https ca-certificates
sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-
keys 58118E89F3A912897C070ADBF76221572C52609D

cat > /etc/apt/sources.list.d/docker.list <<'EOF'
deb https://apt.dockerproject.org/repo debian-stretch main
EOF
sudo apt-get update

# install Docker
sudo apt-get install -y docker-engine
sudo service docker start
sudo docker run hello-world


# configure Docker user group permissions
sudo groupadd docker
sudo gpasswd -a ${USER} docker
sudo service docker restart

# set Docker to auto-launch on startup
sudo systemctl enable docker

------------------------------End of Script-----------------------------
```

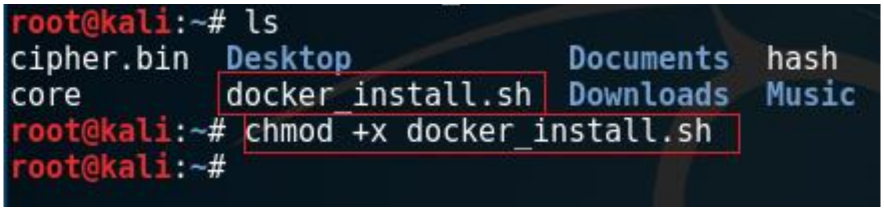
Save the file by pressing CTRL+x.

Type in ‘y’ to save the changes and then press enter to exit.

At the terminal, type, ls and locate your newly created script file.

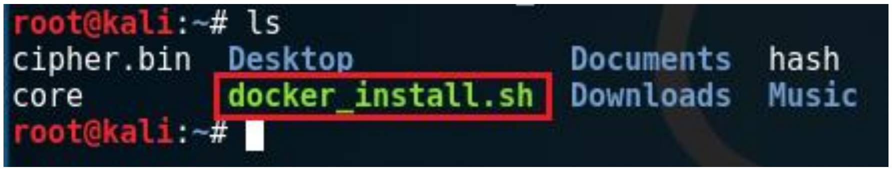
Type the following to make the script executable:

```
chmod +x docker_install.sh
```



```
root@kali:~# ls
cipher.bin  Desktop             Documents  hash
core        docker_install.sh   Downloads  Music
root@kali:~# chmod +x docker_install.sh
root@kali:~#
```

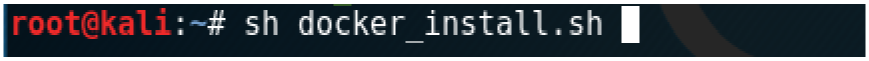
Type in ls and note the color of the file has changed to green annotating that the file is now an executable.



```
root@kali:~# ls
cipher.bin  Desktop            Documents  hash
core        docker_install.sh  Downloads  Music
root@kali:~# 
```

To run the script, at the terminal type:

```
Sh docker_install.sh
```



```
root@kali:~# sh docker_install.sh 
```

Hit enter

Allow to script to run and do not interrupt!

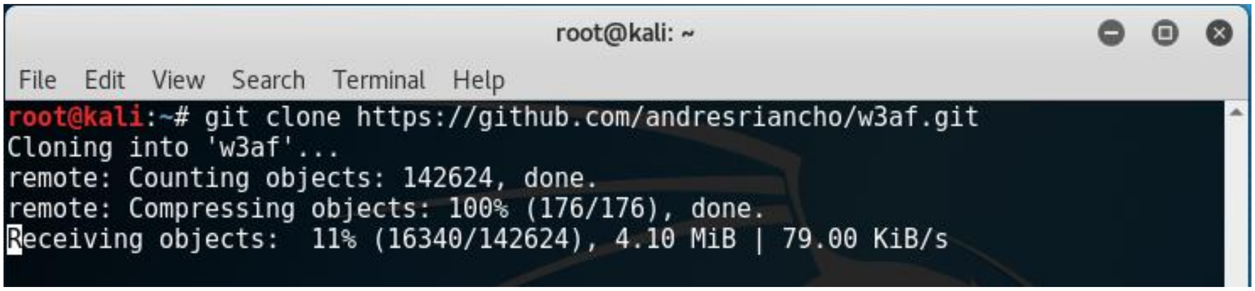
Check to see if Docker is properly installed

```
 docker run hello-world
```

Install the w3af Docker container

Download the w3af package from github using the following command.

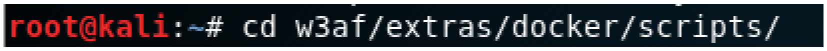
```
git clone https://github.com/andresriancho/w3af.git
```



```
root@kali:~# git clone https://github.com/andresriancho/w3af.git
Cloning into 'w3af'...
remote: Counting objects: 142624, done.
remote: Compressing objects: 100% (176/176), done.
Receiving objects:  11% (16340/142624), 4.10 MiB | 79.00 KiB/s
```

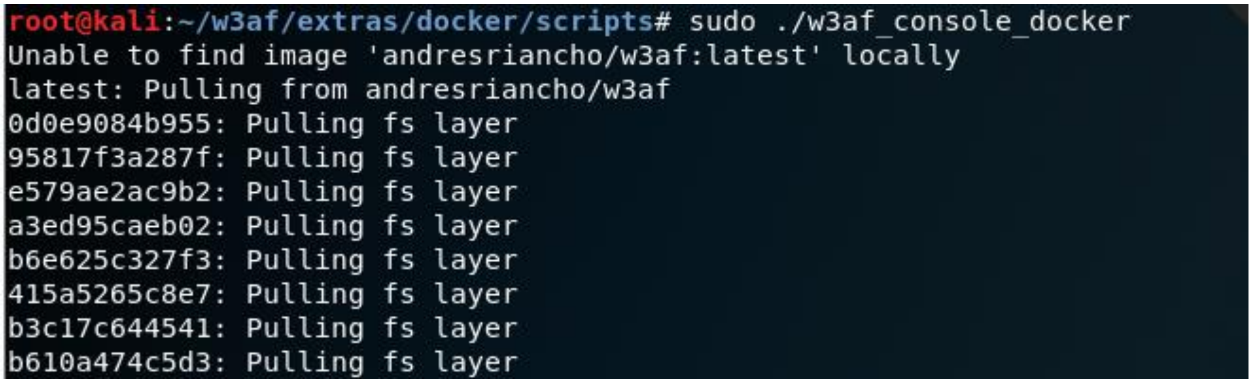
Change path over to the w3af/docker script folder. Inside is a script for installing the w3af scanner as a Docker container.

```
cd w3af/extras/docker/scripts/
```



```
root@kali:~# cd w3af/extras/docker/scripts/
```

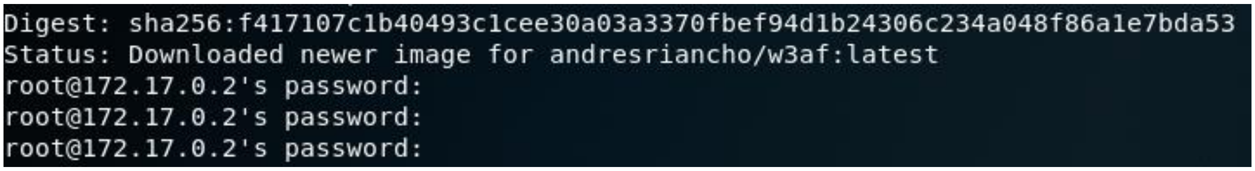
Install the w3af scanner using the pre-built docker script.



```
root@kali:~/w3af/extras/docker/scripts# sudo ./w3af_console_docker
Unable to find image 'andresriancho/w3af:latest' locally
latest: Pulling from andresriancho/w3af
0d0e9084b955: Pulling fs layer
95817f3a287f: Pulling fs layer
e579ae2ac9b2: Pulling fs layer
a3ed95caeb02: Pulling fs layer
b6e625c327f3: Pulling fs layer
415a5265c8e7: Pulling fs layer
b3c17c644541: Pulling fs layer
b610a474c5d3: Pulling fs layer
```

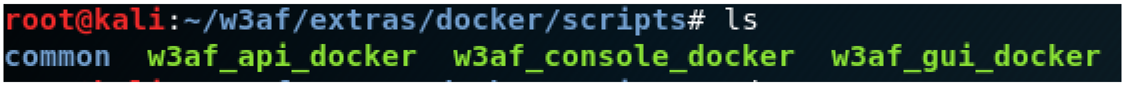
We need to modify the chmod permissions in the

`/w3af/extras/docker/scripts/common/`**`docker_helpers.py`**` script`



```
Digest: sha256:f417107c1b40493c1cee30a03a3370fbef94d1b24306c234a048f86a1e7bda53
Status: Downloaded newer image for andresriancho/w3af:latest
root@172.17.0.2's password:
root@172.17.0.2's password:
root@172.17.0.2's password:
```

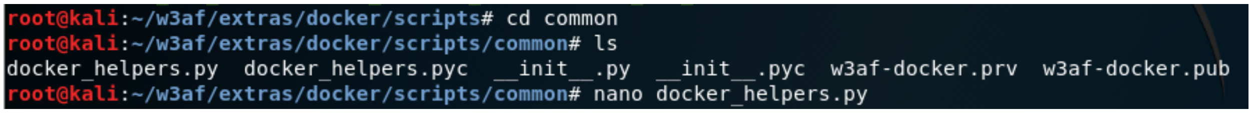
At the terminal type, `ls`



```
root@kali:~/w3af/extras/docker/scripts# ls
common  w3af_api_docker  w3af_console_docker  w3af_gui_docker
```

Change location over the common directory. Show the contents using the ls command.

Use nano to open the docker_helpers.py script.



```
root@kali:~/w3af/extras/docker/scripts# cd common
root@kali:~/w3af/extras/docker/scripts/common# ls
docker_helpers.py  docker_helpers.pyc  __init__.py  __init__.pyc  w3af-docker.prv  w3af-docker.pub
root@kali:~/w3af/extras/docker/scripts/common# nano docker_helpers.py
```

`nano docker_helpers.py`

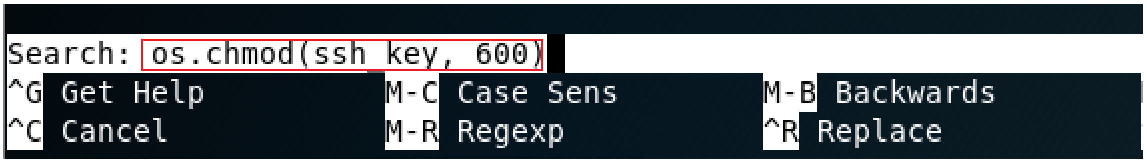
Warning! Follow these directions carefully!

We need to modify or change the chmod value for from decimal to octal.

With the nano editor open, press CTRL + w to search for the follow chmod value.

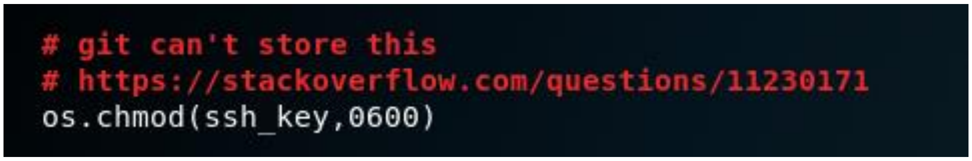
`os.chmod(ssh_key, 600)` The number reads, six-zero-zero.

Press enter.



```
Search: os.chmod(ssh_key, 600)
^G Get Help          M-C Case Sens        M-B Backwards
^C Cancel            M-R Regexp           ^R Replace
```

Change the value of the line to read,
`os.chmod(ssh_key,0600)`



```
# git can't store this
# https://stackoverflow.com/questions/11230171
os.chmod(ssh_key,0600)
```

This will prevent the prompt asking for the root password at the repository site. If you fat finger the change, when you try and run the w3af scanner, you will be prompted for a password. The password is **w3af.** That's how you know you fat fingered the change.

Press CTRL +x to exit.

Press 'y' to save you changes.

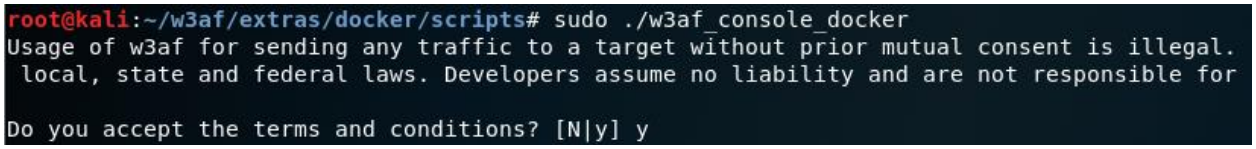
Press enter to exit the nano text editor.

At the terminal type, `cd ..` (there is a space). This returns you to the script folder.

Launch w3af inside the docker container

At the terminal type, `sudo ./w3af_console_docker`

Accept the terms and conditions.



```
root@kali:~/w3af/extras/docker/scripts# sudo ./w3af_console_docker
Usage of w3af for sending any traffic to a target without prior mutual consent is illegal.
 local, state and federal laws. Developers assume no liability and are not responsible for

Do you accept the terms and conditions? [N|y] y
```

You are now ready to proceed to the next lab, Conducting A Website Vulnerability Scan Using w3af.

Summary

Everything we do in a lab, course room or in the real world is a learning opportunity. No matter how much you think you know, the knowledge pool for technology is an inch deep and a mile wide so there will always be something for you to learn.

In this lab, you were introduced to the Docker program. Just like virtualization, containers are not going away. You need to become adept at using the Docker technology and not just reading about it. The only way you can become proficient at using technology, is through applied learning.

You may not be experienced enough to realize it, but by using a Docker container, we were able to download the w3af scanner with all its dependencies and not have to download, install, configure or troubleshoot Kali to accommodate the install. The entire container runs inside of the Docker program within its own space, isolated from the Kali operating system. The container for the w3af program behaves much like a sandbox never calling on Kali for a dependency or program. Everything the w3af scanner needs, came with the container.

Instead of kali having to pull down an updated version of the w3af scanner and having to pull down all the necessary updates that the scanner requires, the developer can create a new container which will have all the updated files and dependencies which the docker program can download from the Docker or Github.

Let’s say you been hired as a programmer to build a C++ application requiring several DOT Net packages along with several other Microsoft updates. Your client needs to track your progress and approve any changes to the application. As you add these packages and updates, you need to provide the updated application to your client. If you just send the application itself, the client will need to update his operating system with the required DOT Net packages and Microsoft updates. By building a Docker container with all required packages, needed to run the application, all the client needs installed is the Docker program itself. The container will have all the updates.

You can visit the Docker hub site and find thousands of prebuilt containers for operating systems and programs. Developers can also create and share containers on the Docker public site.

End of the lab!