

# Setting Up a Penetration Testing Environment with Essential

## Tools

*Series of instructions for installing and configuring various tools and services on a Linux-based system (presumably Kali Linux or Ubuntu). Here's a breakdown of what each part does:*

### 1. Postman Installation

- **Download Postman Tarball:** This downloads the latest version of Postman for Linux.
- **Extract the Tarball:** Extracts the downloaded .tar.gz file to /opt directory.
- **Create a Symlink:** Creates a symlink so that you can run Postman from the terminal using the postman command.
- **Run Postman:** Opens Postman by typing Postman in the terminal.

### 2. Install Git

- Installs Git, a version control tool, using the package manager (apt-get).

### 3. Install Docker

- Installs Docker and Docker Compose to manage containers on the system.

### 4. Install Go

- Installs Go (Golang), a programming language, using the apt package manager.

### 5. Create a New User

- **Create User:** Adds a new user with a home directory and a bash shell.
- **Set Password:** Assigns a password to the newly created user.

### 6. Setup for jwt\_tool

- **Create Virtual Environment:** Creates a new Python virtual environment in the user's home directory.

- **Clone Repository:** Clones the jwt\_tool repository from GitHub and installs dependencies using pip.
- **Run the Tool:** Executes jwt\_tool.py from the virtual environment.

## 7. Install Kiterunner

- **Clone Kiterunner:** Clones the Kiterunner repository from GitHub.
- **Build:** Builds Kiterunner.
- **Create Symlink:** Creates a symlink for easy execution of the tool.

## 8. Setup for Arjun

- **Create Virtual Environment:** Creates a virtual environment for the Arjun tool.
- **Install Arjun:** Installs the Arjun tool in the virtual environment.
- **Change Directory Permissions:** Modifies the permissions for the /opt/Arjun directory if necessary to allow user modifications.

## 9. Install OWASP ZAP

- Installs OWASP ZAP (Zed Attack Proxy), a tool used for security testing and vulnerability scanning.

These steps are typically followed for setting up various security and development tools on a Kali Linux or Ubuntu system.

- Download Postman Tarball

Command :

**sudo wget <https://dl.pstmn.io/download/latest/linux64> -O postman-linux-x64.tar.gz**

```
File Actions Edit View Help
(kali㉿kali)-[~]
$ sudo wget https://dl.pstmn.io/download/latest/linux64 -O postman-linux-x64.tar.gz
[sudo] password for kali:
--2024-12-15 03:13:59-- https://dl.pstmn.io/download/latest/linux64
Resolving dl.pstmn.io (dl.pstmn.io)... 18.161.97.26, 18.161.97.90, 18.161.97.91, ...
Connecting to dl.pstmn.io (dl.pstmn.io)|18.161.97.26|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 149596640 (143M)
Saving to: 'postman-linux-x64.tar.gz'

postman-linux-x64.tar.gz 100%[=====>] 142.67M 6.08MB/s in 23s

2024-12-15 03:14:22 (6.30 MB/s) - 'postman-linux-x64.tar.gz' saved [149596640/149596640]

(kali㉿kali)-[~]
```

- Extract the Tarball

Command :

**sudo tar -xvzf postman-linux-x64.tar.gz -C /opt**

```
(kali㉿kali)-[~]
$ sudo tar -xvzf postman-linux-x64.tar.gz -C /opt
Postman/
Postman/Postman
Postman/app/
Postman/app/libffmpeg.so
Postman/app/Postman
Postman/app/icudtl.dat
Postman/app/chrome_100_percent.pak
Postman/app/resources/
```

- Create a Symlink

Command :

**sudo ln -s /opt/Postman/Postman /usr/bin/postman**

```
(kali㉿kali)-[~]
$ sudo ln -s /opt/Postman/Postman /usr/bin/postman
```

- Run postman

Command :

**Postman**

```
(kali㉿kali)-[~]  
$ postman
```

---

- Install Git

Command :

**sudo apt-get install git**

```
(kali㉿kali)-[~]
└─$ sudo apt-get install git
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.45.2-1).
The following packages were automatically installed and are no longer required:
  imagemagick-6-common libbfio1 libfmt9 libmagickcore-6.q16-7-extra libmagickcore-6.q16-7t64 libmagickwand-6.q16-7t64 libsperl6
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
```

- Install Docker

Command :

**sudo apt-get install docker.io docker-compose**

```
(kali㉿kali)-[~]
└─$ sudo apt-get install docker.io docker-compose
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
(kali㉿kali)-[~]
└─$ sudo apt install docker.io -y
Installing:
  docker.io

Installing dependencies:
  containerd criu docker-cli libcompel1 libintl-xs-perl libmodule-find-perl libproc-processtable-perl libsort-naturally-perl needrestart python3-pycrui runc tini

Suggested packages:
  containernetworking-plugins docker-doc aufs-tools btrfs-progs cgroupfs-mount debootstrap rinse rootlesskit xfsprogs zfs-fuse | zfsutils-linux

Summary:
  Upgrading: 0, Installing: 14, Removing: 0, Not Upgrading: 12
  Download size: 65.8 MB
  Space needed: 259 MB / 60.2 GB available

Get:1 http://http.kali.org/kali kali-rolling/main amd64 runc amd64 1.1.15+ds1-1 [2,950 kB]
Get:2 http://foreign.mirror.net-14/kali kali-rolling/main amd64 containerd amd64 1.0.3-1 [55.2 kB]
```

- Install Go

Command :

**sudo apt install golang-go**

```
(kali㉿kali)-[~]
└─$ sudo apt install golang-go
```

---

### **Create a New User in kali**

Run the following command to create a new user:

Command :

**sudo useradd -m -s /bin/bash <username>**

- Replace <username> with the desired username.
  - The -m flag creates the user's home directory.
  - The -s flag specifies the shell to use (/bin/bash in this case).
- 

### **Step 3: Set a Password for the New User**

Assign a password to the new user:

Command :

**sudo passwd <username>**

- Enter the password when prompted.

```
(kali㉿kali)-[~]
$ sudo useradd -m -s /bin/bash yay03

(kali㉿kali)-[~]
$ sudo passwd yay03

New password:
Retype new password:
passwd: password updated successfully

(kali㉿kali)-[~]
$ su yay03
Password:
(yay03㉿kali)-[/home/kali]
$
```

### **Option 1: Create Virtual Environment in Home Directory**

1. Navigate to your home directory:

Command :

```
cd ~
```

2. Create a new folder for your jwt\_tool and clone the repository there:

Command :

```
mkdir ~/jwt_tool
```

```
cd ~/jwt_tool
```

```
git clone https://github.com/ticarpi/jwt_tool
```

```
cd jwt_tool
```

3. Create the virtual environment in the new folder:

Command :

```
python3 -m venv venv
```

### **Activate the Virtual Environment**

1. Activate the virtual environment:

Command :

```
source venv/bin/activate
```

After activation, your prompt should change to indicate the environment is active (e.g., (venv)).

---

### **Install the Required Packages**

1. Use pip to install the required dependencies:

Command :

**python3 -m pip install termcolor cprint pycryptodomex requests**

These packages will be installed in the virtual environment, isolated from the system Python.

---

### **Step 4: Run the Tool**

1. Run the tool using the Python interpreter in the virtual environment:

Command :

**python3 jwt\_tool.py**



```

(yay03@kali)-[/opt/jwt_tool]
$ cd ~

(yay03@kali)-[~]
$ mkdir ~/jwt_tool
cd ~/jwt_tool
git clone https://github.com/ticarpi/jwt_tool
cd jwt_tool
Cloning into 'jwt_tool' ...
remote: Enumerating objects: 237, done.
remote: Counting objects: 100% (99/99), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 237 (delta 92), reused 73 (delta 73), pack-reused 138 (from 1)
Receiving objects: 100% (237/237), 137.75 KiB | 1.06 MiB/s, done.
Resolving deltas: 100% (117/117), done.

(yay03@kali)-[~/jwt_tool/jwt_tool]
$ python3 -m venv venv

(yay03@kali)-[~/jwt_tool/jwt_tool]
$ source venv/bin/activate

(venv)(yay03@kali)-[~/jwt_tool/jwt_tool]
$ python3 -m pip install termcolor cprint pycryptodomex requests
Collecting termcolor
  Downloading termcolor-2.5.0-py3-none-any.whl.metadata (6.1 kB)
Collecting cprint
  Downloading cprint-1.2.2.tar.gz (2.3 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting pycryptodomex
  Downloading pycryptodomex-3.21.0-cp36-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.4 kB)
Collecting requests
  Downloading requests-2.32.3-py3-none-any.whl.metadata (4.6 kB)
Collecting charset-normalizer<4, ≥2 (from requests)
  Downloading charset_normalizer-3.4.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (34 kB)
Collecting idna<4, ≥2.5 (from requests)
  Downloading idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting urllib3<3, ≥1.21.1 (from requests)
  Downloading urllib3-2.2.3-py3-none-any.whl.metadata (6.5 kB)
Collecting certifi≥2017.4.17 (from requests)
  Downloading certifi-2024.12.14-py3-none-any.whl.metadata (2.3 kB)
Downloading termcolor-2.5.0-py3-none-any.whl (7.8 kB)
Downloading pycryptodomex-3.21.0-cp36-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.3 MB)
  2.3/2.3 MB 5.2 MB/s eta 0:00:00
Downloading requests-2.32.3-py3-none-any.whl (64 kB)
Downloading certifi-2024.12.14-py3-none-any.whl (164 kB)
Downloading charset_normalizer-3.4.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (143 kB)
Downloading idna-3.10-py3-none-any.whl (70 kB)
Downloading urllib3-2.2.3-py3-none-any.whl (126 kB)
Building wheels for collected packages: cprint
  Building wheel for cprint (pyproject.toml) ... done

```

## Activate

```

(venv)(yay03@kali)-[~/jwt_tool/jwt_tool]
$ python3 jwt_tool.py

  JWT Tool
  Version 2.2.7 @ticarpi

No config file yet created.
Running config setup.
Configuration file built - review contents of "jwtconf.ini" to customise your options.
Make sure to set the "httplistener" value to a URL you can monitor to enable out-of-band checks.

(venv)(yay03@kali)-[~/jwt_tool/jwt_tool]
$

```

## Install Kiterunner

- `sudo git clone https://github.com/assetnote/kiterunner.git`
- `cd kiterunner`
- `sudo make build`
- `sudo ln -s /opt/kiterunner/dist/kr /usr/bin/kr`

```
(yay03@kali)~/jwt_tool/jwt_tool
$ cd /opt

(yay03@kali)~/opt
$ sudo git clone https://github.com/assetnote/kiterunner.git
[sudo] password for yay03:
Cloning into 'kiterunner'...
remote: Enumerating objects: 230, done.
remote: Counting objects: 100% (28/28), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 230 (delta 16), reused 10 (delta 10), pack-reused 202 (from 1)
Receiving objects: 100% (230/230), 235.28 KiB | 1.02 MiB/s, done.
Resolving deltas: 100% (49/49), done.

(yay03@kali)~/opt
$ cd kiterunner

(yay03@kali)~/opt/kiterunner
$ sudo make build
mkdir -p dist
go build -ldflags "-extld 'g++' -extldflags '-static' -X 'github.com/assetnote/kiterunner/cmd/kiterunner/cmd.Version=hub.com/assetnote/kiterunner/cmd/kiterunner/cmd.Date=Sun Dec 15 05:26:29 AM EST 2024'" -o dist/kr ./cmd/kiterunner
go: downloading github.com/mitchellh/go-homedir v1.1.0
go: downloading github.com/spf13/cobra v1.1.3
go: downloading github.com/spf13/viper v1.7.1
go: downloading github.com/dustin/go-humanize v1.0.0
go: downloading github.com/hashicorp/go-multierror v1.0.0
go: downloading github.com/olekukonko/tablewriter v0.0.5
go: downloading github.com/rs/zerolog v1.20.0
go: downloading github.com/manifoldco/promptui v0.8.0
go: downloading github.com/andybalholm/brotli v1.0.0
go: downloading github.com/klauspost/compress v1.10.7
go: downloading github.com/lunixbochs/vtclean v1.0.0
go: downloading golang.org/x/term v0.0.0-20210220032956-6a3ed077a48d

(yay03@kali)~/opt/kiterunner
$ sudo ln -s /opt/kiterunner/dist/kr /usr/bin/kr

(yay03@kali)~/opt/kiterunner
$
```

```
(kali㉿kali)-[~]
$ cd ~

(kali㉿kali)-[~]
$ python3 -m venv arjun_venv

(kali㉿kali)-[~]
$ source arjun_venv/bin/activate

(arjun_venv)-(kali㉿kali)-[~]
$ cd /opt/Arjun
python3 setup.py install

Traceback (most recent call last):
  File "/opt/Arjun/setup.py", line 5, in <module>
    from setuptools import setup, find_packages
ModuleNotFoundError: No module named 'setuptools'

(arjun_venv)-(kali㉿kali)-[/opt/Arjun]
$ pip install setuptools

Collecting setuptools
  Using cached setuptools-75.6.0-py3-none-any.whl.metadata (6.7 kB)
Downloading setuptools-75.6.0-py3-none-any.whl (1.2 MB)
1.2/1.2 MB 3.7 MB/s eta 0:00:00
Installing collected packages: setuptools
Successfully installed setuptools-75.6.0

(arjun_venv)-(kali㉿kali)-[/opt/Arjun]
$ python3 setup.py install

running install
/home/kali/arjun_venv/lib/python3.12/site-packages/setuptools/_distutils/cmd.py:66: SetuptoolsDeprecationWarning: se
tup.py install is deprecated.
!!
```

### Option 1: Create the Virtual Environment in Your Home Directory

1. Change to your home directory:

Command :

```
cd ~
```

2. Create a virtual environment inside your home directory:

Command :

```
python3 -m venv arjun_venv
```

3. Activate the virtual environment:

Command :

```
source arjun_venv/bin/activate
```

4. Then, proceed to install the package in this virtual environment:

Command :

```
cd /opt/Arjun
```

```
python3 setup.py install
```

#### **Option 1: Change the Directory Permissions**

You can change the permissions of the /opt/Arjun directory to allow your user to write to it:

1. Run this command to give write access to your user:

Command :

```
sudo chown -R $USER:$USER /opt/Arjun
```

2. After that, try running the installation command again in the virtual environment:

Command :

```
python3 setup.py install
```

```

(arjun_venv)-(kali@kali)-[/opt/Arjun]
$ sudo chown -R $USER:$USER /opt/Arjun

[sudo] password for kali:

(arjun_venv)-(kali@kali)-[/opt/Arjun]
$ python3 setup.py install

running install
/home/kali/arjun_venv/lib/python3.12/site-packages/setuptools/_distutils/cmd.py:66: SetuptoolsDeprecationWarning: se
tup.py install is deprecated.
!!

*****
Please avoid running ``setup.py`` directly.
Instead, use pypa/build, pypa/installer or other
standards-based tools.

See https://blog.ganssle.io/articles/2021/10/setup-py-deprecated.html for details.
*****

!!
self.initialize_options()
/home/kali/arjun_venv/lib/python3.12/site-packages/setuptools/_distutils/cmd.py:66: EasyInstallDeprecationWarning: e
asy_install command is deprecated.
!!

*****
Please avoid running ``setup.py`` and ``easy_install``.
Instead, use pypa/build, pypa/installer or other
standards-based tools.

See https://github.com/pypa/setuptools/issues/917 for details.
*****

!!
self.initialize_options()
running bdist_egg
running egg_info
creating arjun.egg-info
writing arjun.egg-info/PKG-INFO
writing dependency_links to arjun.egg-info/dependency_links.txt
writing entry points to arjun.egg-info/entry_points.txt
writing requirements to arjun.egg-info/requirements.txt

```

```

Installed /home/kali/arjun_venv/lib/python3.12/site-packages/idna-3.10-py3.12.egg
Searching for charset-normalizer<4, >=2
Reading https://pypi.org/simple/charset-normalizer/
Downloading https://files.pythonhosted.org/packages/bf/9b/08c0432272d77b04803958a4598a51e2a4b51c06640af8b8f0f908c18b
f2/charset-normalizer-3.4.0-py3-none-any.whl#sha256-fe9f97feb71aa9896b81973a7bbada8c49501dc73e58a10fcef6663af95e5079
Best match: charset-normalizer 3.4.0
Processing charset_normalizer-3.4.0-py3-none-any.whl
Installing charset_normalizer-3.4.0-py3-none-any.whl to /home/kali/arjun_venv/lib/python3.12/site-packages
Adding charset-normalizer 3.4.0 to easy-install.pth file
detected new path './idna-3.10-py3.12.egg'
Installing normalizer script to /home/kali/arjun_venv/bin

Installed /home/kali/arjun_venv/lib/python3.12/site-packages/charset_normalizer-3.4.0-py3.12.egg
Finished processing dependencies for arjun==2.2.7

```

```

(arjun_venv)-(kali@kali)-[/opt/Arjun]
$

```

```

(arjun_venv)-(kali@kali)-[/opt/Arjun]
$ arjun

```

- Install OWASP ZAP

Command : **sudo apt install zaproxy**

```
(arjun_venv)-(kali㉿kali)-[~]
$ sudo apt install zaproxy
Installing:
  zaproxy

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 12
  Download size: 213 MB
  Space needed: 266 MB / 59.5 GB available

Get:1 http://kali.download/kali kali-rolling/main amd64 zaproxy all 2.15.0-0kali1 [213 MB]
Fetched 213 MB in 39s (5,525 kB/s)
Selecting previously unselected package zaproxy.
(Reading database ... 425503 files and directories currently installed.)
Preparing to unpack .../zaproxy_2.15.0-0kali1_all.deb ...
Unpacking zaproxy (2.15.0-0kali1) ...
Setting up zaproxy (2.15.0-0kali1) ...
Processing triggers for kali-menu (2024.4.0) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

(arjun_venv)-(kali㉿kali)-[~]
$
(arjun_venv)-(kali㉿kali)-[~]
$
```

- Open zaproxy

