# How to securely isolate and execute legion from Kali Linux

Version 0.1, Last Updated: 2024-05-25

This site is dedicated to sharing information about the practice, ideas, concepts and patterns regarding computer security.

# Table of Contents

# Chapter 1. Introduction

The motivation behind this paper is to explore using the tool legion that comes with Kali Linux.

Legion is a free and easy-to-use tool for network security testing. It helps the cybersecurity professional find, gather information, and identify weaknesses in computer systems, making it easier to test the security of the system and then to secure said system(s).

## 1.1. Included Features:

- Automatic detection of CPEs (Common Platform Enumeration) and CVEs (Common Vulnerabilities and Exposures).
- Automatic recon and scanning with NMAP, whataweb, nikto, Vulners, Hydra, SMBenum, dirbuster, sslyzer, webslayer and more (with almost 100 auto-scheduled scripts).
- Easy to use graphical interface with rich context menus and panels that allow pentesters to quickly find and exploit attack vectors on hosts.
- Highly customizable stage scanning for ninja-like IPS evasion.
- Modular functionality allows users to easily customize Legion and automatically call their own scripts/tools.
- Multiple custom scan configurations ideal for testing different environments of various size and complexity.
- Realtime auto-saving of project results and tasks.
- Ties CVEs to Exploits as detailed in Exploit-Database.

The benefits of using Legion compared to other network scanning tools are:

- **User-Friendly**: Legion has a simple interface that's easy to navigate, making it great for beginners.
- **Automated Scanning**: It speeds up the process by automatically gathering information and finding vulnerabilities.
- **Built-In Tools**: Legion combines multiple security tools like Nmap and Nikto in one place, so you don't have to use them separately.
- **Extensible**: You can add new features or tools to Legion, giving it flexibility as your skills grow.
- **Detailed Reporting**: It organizes scan results in an easy-to-understand format, which can be saved for later analysis.

# Chapter 2. Requirements

## 2.1. Writing Conventions

If you see the following $ symbol on a command line to execute, what that means is that the command is executed as a regular user; meaning an account that does not have administrative privileges. Ignore the leading $ and execute the rest of the command.

```
$ command to execute as a regular user
```

If you see a command line lead with the # symbol, then that means that the command is executed as the root user. This implies you need to elevate to the root user before running the command, e.g. with: `sudo su - root`.

```
# command to execute as the root user
```

## 2.2. VirtualBox

Go to: https://www.virtualbox.org/wiki/Downloads and download VirtualBox.

The author is running on Ubuntu 18.04, so following to this URL: https://www.virtualbox.org/wiki/Linux_Downloads

For Ubuntu, double click on the .deb file, i.e. virtualbox-5.2_5.2.0-118431-Ubuntu-zesty_amd64.deb, and install VirtualBox on your local workstation.

### 2.2.1. Clean VirtualBox Networking

This section is here in case you already had virtualbox installed from before. The intent is to clean up the previous networking. If you do not need to do this, skip to Add VirtualBox Networking

Run these two commands from a Terminal:

```
$ VBoxManage list natnetworks
$ VBoxManage list dhcpservers
```

Output (example):

```
NetworkName:    192.168.139-NAT
IP:             192.168.139.1
Network:        192.168.139.0/24
IPv6 Enabled:   No
IPv6 Prefix:    fd17:625c:f037:2::/64
DHCP Enabled:   Yes
Enabled:        Yes
loopback mappings (ipv4)
        127.0.0.1=2


NetworkName:    192.168.139-NAT
Dhcpd IP:       192.168.139.3
LowerIPAddress: 192.168.139.101
UpperIPAddress: 192.168.139.254
NetworkMask:    255.255.255.0
Enabled:        Yes
Global Configuration:
    minLeaseTime:     default
    defaultLeaseTime: default
    maxLeaseTime:     default
    Forced options:   None
    Suppressed opts.: None
        1/legacy: 255.255.255.0
Groups:               None
Individual Configs:   None

NetworkName:    HostInterfaceNetworking-vboxnet0
Dhcpd IP:       172.20.0.3
LowerIPAddress: 172.20.0.101
UpperIPAddress: 172.20.0.254
NetworkMask:    255.255.255.0
Enabled:        Yes
Global Configuration:
    minLeaseTime:     default
    defaultLeaseTime: default
    maxLeaseTime:     default
    Forced options:   None
    Suppressed opts.: None
        1/legacy: 255.255.255.0
Groups:               None
Individual Configs:   None
```

Now, delete ALL of the pre-installed VirtualBox networks (one at a time following the syntax below):

```
VBoxManage natnetwork remove --netname <NetworkName_from_above>
VBoxManage natnetwork remove --netname 192.168.139-NAT
```

Repeat as many times as necessary to delete all of them.

Now, delete ALL of the pre-installed DHCP services:

```
VBoxManage dhcpserver remove --netname <DHCP_Server_NetworkName_from_above>
VBoxManage dhcpserver remove --netname 192.168.139-NAT
```

Repeat as many times as necessary to delete all of them.

### 2.2.2. Add VirtualBox Networking

Now, add the new VirtualBox networks so the Kali Linux guides work.

```
VBoxManage natnetwork add \
    --netname 192.168.139-NAT \
    --network "192.168.139.0/24" \
    --enable --dhcp on

VBoxManage dhcpserver add \
    --netname 192.168.139-NAT \
    --ip 192.168.139.3 \
    --lowerip 192.168.139.101 \
    --upperip 192.168.139.254 \
    --netmask 255.255.255.0 \
    --enable

VBoxManage hostonlyif create

VBoxManage hostonlyif ipconfig vboxnet0 \
    --ip 172.20.0.1 \
    --netmask 255.255.255.0

VBoxManage dhcpserver add \
    --ifname vboxnet0 \
    --ip 172.20.0.3 \
    --lowerip 172.20.0.101 \
    --upperip 172.20.0.254 \
    --netmask 255.255.255.0

VBoxManage dhcpserver modify \
    --ifname vboxnet0 \
    --enable
```

VirtualBox install complete.

## 2.3. Vagrant

Go to: https://www.vagrantup.com/downloads.html, follow the appropriate link to your OS and 32 or 64 bit version representing your local workstation. Download.

For Ubuntu, double click on the .deb file, i.e. vagrant_2.0.1_x86_64.deb, and install Vagrant on your local system.

| | |
|---|---|
| Note | Update vagrant vm: `vagrant box update` |

## 2.4. Kali Linux and Damn Vulnerable Web Application (DVWA)

The author highly recommends to create a directory structure that is easy to navigate and find your code. As an example, you could use something similar to:

```
${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

Go ahead and make this structure with the following command (inside a Terminal):

```
$ mkdir -p ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

From a Terminal, change directory to:

```
$ cd ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

### 2.4.1. Vagrantfile

Inside of the kali-linux-vm directory, populate a new file with the exact name, “Vagrantfile”. Case matters, uppercase the “V”. This file will contain both virtual machines for Kali Linux as well as setting up the DVWA virtual machine. Aggregating both virtual machines into one file has saved the author a lot of time. The coolness here is setting up the variables at the top of the Vagrantfile mimicing shell scripting inside of a virtual machine (passed in with provision: shell ). I tested using: `apt-get update && apt-get upgrade -y`, but opted to take it out since it took over 45 minutes on my slower (old) hardware. See comment about downloading this file immediately preceding the code block.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

$os_update = <<SCRIPT
apt-get update
SCRIPT
```

```
VAGRANTFILE_API_VERSION = "2"


Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
    config.vm.define "kali-linux-vagrant" do |conf|
        conf.vm.box = "kalilinux/rolling"

        # For Linux systems with the Wireless network, uncomment the line:
        conf.vm.network "public_network", bridge: "wlo1", auto_config: true

        # For macbook/OSx systems, uncomment the line and comment out the Linux
Wireless network:
        #conf.vm.network "public_network", bridge: "en0: Wi-Fi (AirPort)",
auto_config: true

        conf.vm.hostname = "kali-linux-vagrant"
        conf.vm.provider "virtualbox" do |vb|
            vb.gui = true
            vb.memory = "4096"
            vb.cpus = "2"
            vb.customize ["modifyvm", :id, "--vram", "32"]
            vb.customize ["modifyvm", :id, "--accelerate3d", "off"]
            vb.customize ["modifyvm", :id, "--ostype", "Debian_64"]
            vb.customize ["modifyvm", :id, "--boot1", "dvd"]
            vb.customize ["modifyvm", :id, "--boot2", "disk"]
            vb.customize ["modifyvm", :id, "--audio", "none"]
            vb.customize ["modifyvm", :id, "--clipboard", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--draganddrop", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--paravirtprovider", "kvm"]
        end
        conf.vm.provision "shell", inline: $os_update
    end

    config.vm.define "dvwa-vagrant" do |conf|

        conf.vm.box = "ubuntu/xenial64"

        conf.vm.hostname = "dvwa-vagrant"

        # For Linux systems with the Wireless network, uncomment the line:
        conf.vm.network "public_network", bridge: "wlo1", auto_config: true

        # For macbook/OSx systems, uncomment the line and comment out the Linux
Wireless network:
        #conf.vm.network "public_network", bridge: "en0: Wi-Fi (AirPort)",
auto_config: true

        config.vm.network "forwarded_port", guest: 80, host: 8080, auto_correct:
true
        config.vm.network "forwarded_port", guest: 3306, host: 3306,
auto_correct: true

        conf.vm.provider "virtualbox" do |vb|
            vb.memory = "1024"
            vb.cpus = "2"
            vb.gui = false
            vb.customize ["modifyvm", :id, "--vram", "32"]
            vb.customize ["modifyvm", :id, "--accelerate3d", "off"]
            vb.customize ["modifyvm", :id, "--ostype", "Ubuntu_64"]
            vb.customize ["modifyvm", :id, "--boot1", "dvd"]
            vb.customize ["modifyvm", :id, "--boot2", "disk"]
            vb.customize ["modifyvm", :id, "--audio", "none"]
            vb.customize ["modifyvm", :id, "--clipboard", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--draganddrop", "hosttoguest"]
            vb.customize ["modifyvm", :id, "--paravirtprovider", "kvm"]
        end
        conf.vm.provision "shell", inline: $os_update
        conf.vm.provision :shell, path: "bootstrap.sh"
    end
end
```

Save and write this file.

You can also download from:

```
$ curl -o Vagrantfile
http://securityhardening.com/files/Vagrantfile_20200928.txt
```

## 2.4.2. bootstrap.sh

Inside of the kali-linux-vm directory, populate a new file with the exact name, `bootstrap.sh`. Case matters, all lowercase. See comment about downloading this file immediately preceding the code block. `bootstrap.sh` (include the shebang in your file: the first line with `#!/usr/bin/env bash` ):

```
#!/usr/bin/env bash
PHP_FPM_PATH_INI='/etc/php/7.0/fpm/php.ini'
PHP_FPM_POOL_CONF='/etc/php/7.0/fpm/pool.d/www.conf'
MYSQL_ROOT_PW='Assword12345'
MYSQL_dvwa_user='dvwa'
MYSQL_dvwa_password='sunshine'
DVWA_admin_password='admin'
recaptcha_public_key='u8392ihj32kl8hujalkshuil32'
recaptcha_private_key='89ry8932873832lih32ilj32'
```

```
install_base() {
    add-apt-repository -y ppa:nginx/stable
    sudo apt-get update
    sudo apt-get dist-upgrade -y
    sudo apt-get install -y \
        nginx \
        mariadb-server \
        mariadb-client \
        php \
        php-common \
        php-cgi \
        php-fpm \
        php-gd \
        php-cli \
        php-pear \
        php-mcrypt \
        php-mysql \
        php-gd \
        git \
        vim
}

config_mysql(){
    mysqladmin -u root password "${MYSQL_ROOT_PW}"
## Config the mysql config file for root so it doesn't prompt for password.
## Also sets pw in plain text for easy access.
## Don't forget to change the password here!!

cat <<EOF > /root/.my.cnf
[client]
user="root"
password="${MYSQL_ROOT_PW}"
EOF
    mysql -BNe "drop database if exists dvwa;"
    mysql -BNe "CREATE DATABASE dvwa;"
    mysql -BNe "GRANT ALL ON *.* TO '"${MYSQL_dvwa_user}"'@'localhost'
IDENTIFIED BY '"${MYSQL_dvwa_password}"';"

    systemctl enable mysql
    systemctl restart mysql
    sleep 2
}


config_php(){
    ## Config PHP FPM INI to disable some security settings:

    sed -i 's/^;cgi.fix_pathinfo.*$/cgi.fix_pathinfo = 0/g' ${PHP_FPM_PATH_INI}
    sed -i 's/allow_url_include = Off/allow_url_include = On/g'
${PHP_FPM_PATH_INI}
    sed -i 's/allow_url_fopen = Off/allow_url_fopen = On/g' ${PHP_FPM_PATH_INI}
    sed -i 's/safe_mode = On/safe_mode = Off/g' ${PHP_FPM_PATH_INI}
    echo "magic_quotes_gpc = Off" >> ${PHP_FPM_PATH_INI}
    sed -i 's/display_errors = Off/display_errors = On/g' ${PHP_FPM_PATH_INI}

    ## explicitly set pool options
    ## (these are defaults in ubuntu 16.04 so i'm commenting them out.
    ## If they are not defaults for you try uncommenting these)
    #sed -i 's/^;security.limit_extensions.*$/security.limit_extensions = \
    #.php .php3 .php4 .php5 .php7/g' /etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^listen.owner.*$/listen.owner = www-data/g'
/etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^listen.group.*$/listen.group = www-data/g'
/etc/php/7.0/fpm/pool.d/www.conf
    #sed -i 's/^;listen.mode.*$/listen.mode = 0660/g'
/etc/php/7.0/fpm/pool.d/www.conf


    systemctl restart php7.0-fpm
}


config_nginx(){

cat << 'EOF' > /etc/nginx/sites-enabled/default
server
{
    listen  80;
    root /var/www/html;
    index index.php index.html index.htm;
    #server_name localhost
    location "/"
    {
        index index.php index.html index.htm;
        #try_files $uri $uri/ =404;
    }

    location ~ \.php$
    {
        include /etc/nginx/fastcgi_params;
        fastcgi_pass unix:/var/run/php/php7.0-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $request_filename;
    }
}
EOF

    systemctl restart nginx
}


install_dvwa(){

    if [[ ! -d "/var/www/html" ]];
    then
          mkdir -p /var/www;
          ln -s /usr/share/nginx/html /var/www/html;
          chown -R www-data. /var/www/html;
    fi

    cd /var/www/html
    rm -rf /var/www/html/.[!.]*
    rm -rf /var/www/html/*
    git clone https://github.com/ethicalhack3r/DVWA.git ./
    chown -R www-data. ./
    cp config/config.inc.php.dist config/config.inc.php

    ### chmod uploads and log file to be writable by nobody
    chmod 777 ./hackable/uploads/
    chmod 777 ./external/phpids/0.6/lib/IDS/tmp/phpids_log.txt

    ## change the values in the config to match our setup (these are what you
need to update!
    sed -i '/db_user/ s/root/'${MYSQL_dvwa_user}'/'
/var/www/html/config/config.inc.php
    sed -i '/db_password/ s/p@ssw0rd/'${MYSQL_dvwa_password}'/'
/var/www/html/config/config.inc.php
    sed -i "/recaptcha_public_key/ s/''/'"${recaptcha_public_key}"'/"
/var/www/html/config/config.inc.php
    sed -i "/recaptcha_private_key/ s/''/'"${recaptcha_private_key}"'/"
/var/www/html/config/config.inc.php

}


update_mysql_user_pws(){
## The mysql passwords are set via
/usr/share/nginx/html/dvwa/includes/DBMS/MySQL.php.
#  If you edit this every time they are reset it will reset to those.
#  Otherwise you can do a sql update statement to update them all (they are just
md5's of the string.
#  The issue is the users table doesn't get created until you click that button
T_T to init.

#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE
user = 'admin';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE
user = 'gordonb';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE
user = '1337';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE
user = 'pablo';"
#mysql -BNe "UPDATE dvwa.users SET password = md5('YOUR_MYSQL_PW_HERE') WHERE
user = 'smithy';"

sed -i '/admin/ s/password/'${DVWA_admin_password}'/g'
/var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/gordonb/ s/abc123/'${DVWA_admin_password}'/g'
/var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/1337/ s/charley/'${DVWA_admin_password}'/g'
/var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/pablo/ s/letmein/'${DVWA_admin_password}'/g'
/var/www/html/dvwa/includes/DBMS/MySQL.php
sed -i '/smithy/ s/password/'${DVWA_admin_password}'/g'
/var/www/html/dvwa/includes/DBMS/MySQL.php
}


install_base
config_mysql
install_dvwa
update_mysql_user_pws
config_php
config_nginx
```

Save and write this file.

If you have issues with copying and pasting the above file because code blocks in PDFs always copy correctly [NOT!], you could use curl, i.e. Make sure the bootstrap.sh file ends up in the same directory as the Vagrantfile.

```
$ curl -o bootstrap.sh
http://securityhardening.com/files/bootstrap_sh_20200928.txt
```

From a Terminal, change directory to:

```
$ cd ${HOME}/Source_Code/Education/vagrant-machines/kali-linux-vm/
```

Then run (inside the directory kali-linux-vm):

```
$ vagrant up
```

This will download the appropriate images and start the virtual machines. Once running, through the VirtuaBox GUI, login as root. Password is “toor”, root backwards. Edit the following file: `/etc/ssh/sshd_config`

And change the line: `#PermitRootLogin prothibit-password` To: `PermitRootLogin yes` Meaning strip the comment out on the beginning of the line and alter `prohibit-password` to `yes`.

Then restart the ssh daemon:

```
# kill -HUP $(pgrep sshd)
```

Notice, you are on a Bridged adapter, this will open the instance to allow root to ssh in with the most unsecure password in the world. Only make this change (allowing root to login via SSH) if you require root SSH access. You can change the root user's password, which is highly recommended.

For the DVWA instance, I would first run 'vagrant status' to capture the name that vagrant is using for the running instance.

```
# vagrant status
```

Choose

```
Current machine states:
kali-linux-vagrant running (virtualbox)
dvwa-vagrant running (virtualbox)
```

This environment represents multiple VMs. The VMs are all listed above with their current state. For more information about a specific VM, run `vagrant status NAME`.

From there, log into the DVWA instance with:

```
$ vagrant ssh dvwa-vagrant
```

And then get the current IP address.

```
$ ip a
```

Choose the second network adapter, it should look like:

```
ubuntu@dvwa:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1
    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
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 02:53:17:3c:de:80 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::53:17ff:fe3c:de80/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 08:00:27:f0:77:2d brd ff:ff:ff:ff:ff:ff
    inet 172.20.156.76/24 brd 172.20.156.255 scope global enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fef0:772d/64 scope link
        valid_lft forever preferred_lft forever
```

The test network used for this paper uses 172.20.156.0/24 as the network range [shown here in section 3]. Therefore, the adapter, enp0s8 is what he is looking for. The IP to use as a target is 172.20.156.76. Write down your value.

# Chapter 3. legion

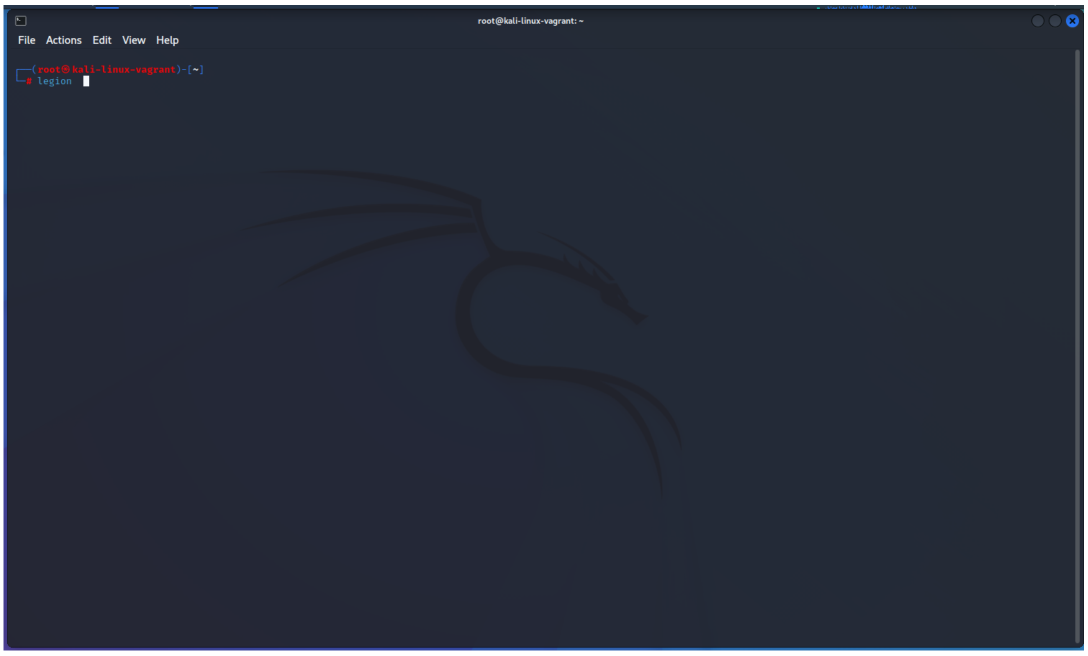
Typically the legion tool comes installed with Kali Linux. If it doesn't, you will need to install it by running the following command in a terminal:

```
apt install legion -y
```

To run the local tool, launch from a terminal by typing:

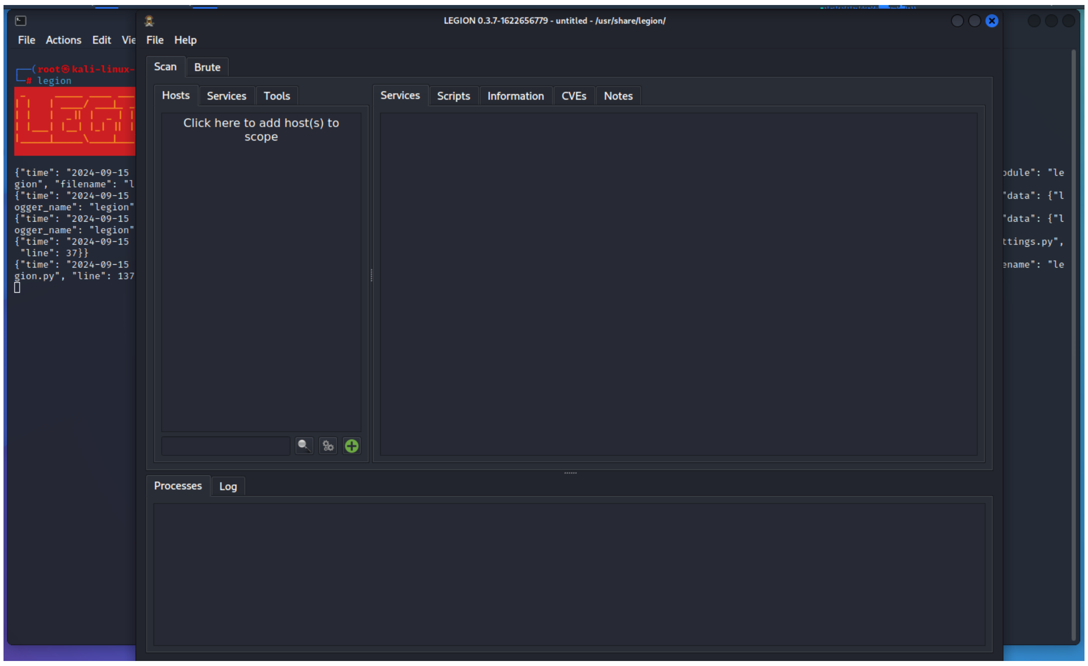
```
legion &
```

Example:



```
┌──(root㉿kali-linux-vagrant)-[~]
└─# legion
```

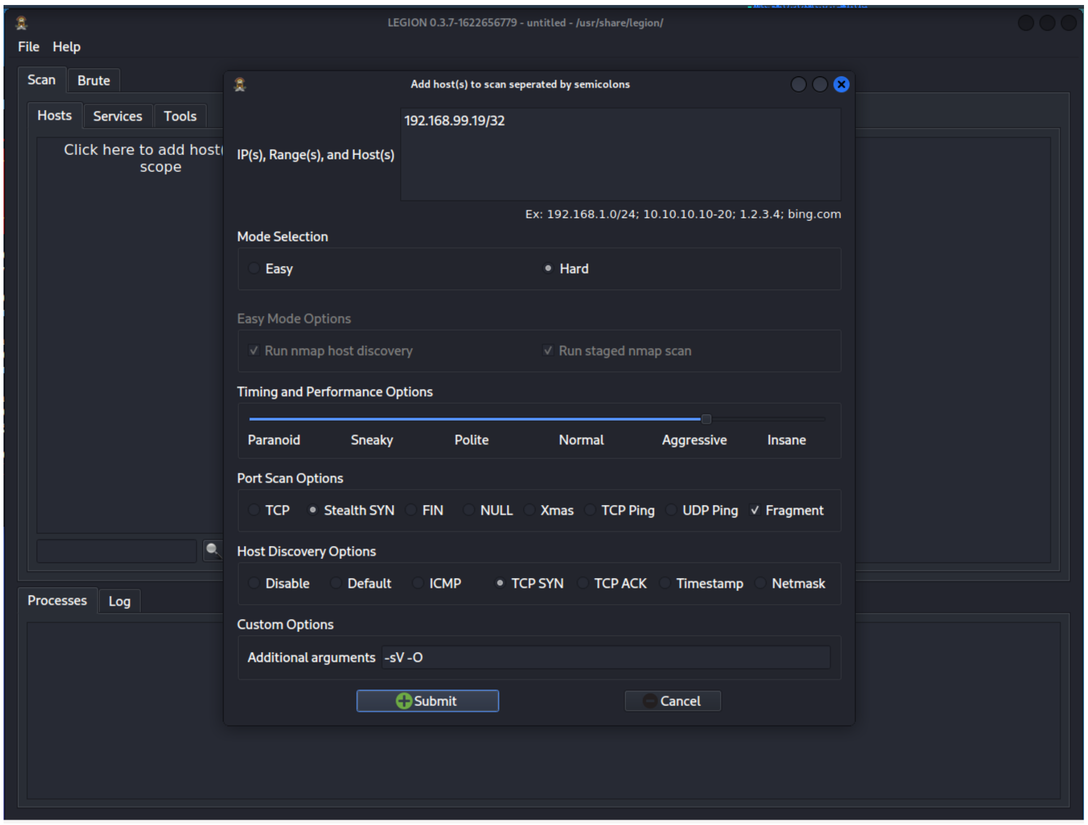
From the initial page, select under `Scan` the keyword `Hosts` tab. Where it says `Click here`, do so and enter in your target systems IP address in CIDR notation. e.g. `192.168.99.19/32` for a single IP on the subnet.



```
┌──(root㉿kali-linux-
└─# legion
{"time": "2024-09-15
gion", "filename": "l
{"time": "2024-09-15
ogger_name": "legion"
{"time": "2024-09-15
ogger_name": "legion"
{"time": "2024-09-15
 "line": 37}}
{"time": "2024-09-15
gion.py", "line": 137
```

```
odule": "le
"data": {"l
"data": {"l
ttings.py",
ename": "le
```

From the next screen, I like to choose the scan type of `Hard` and then leave the rest of the defaults. Click on the `Submit` button.


LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/
File Help
Scan
Brute
Hosts
Services
Tools
Click here to add host scope
Processes
Log

Add host(s) to scan seperated by semicolons

IP(s), Range(s), and Host(s): 192.168.99.19/32

Ex: 192.168.1.0/24; 10.10.10.10-20; 1.2.3.4; bing.com

Mode Selection: Easy [✓ Hard]

Easy Mode Options: ✓ Run nmap host discovery ✓ Run staged nmap scan

Timing and Performance Options: Paranoid Sneaky Polite Normal [✓ Aggressive] Insane

Port Scan Options: TCP [✓ Stealth SYN] FIN NULL Xmas TCP Ping UDP Ping [✓ Fragment]

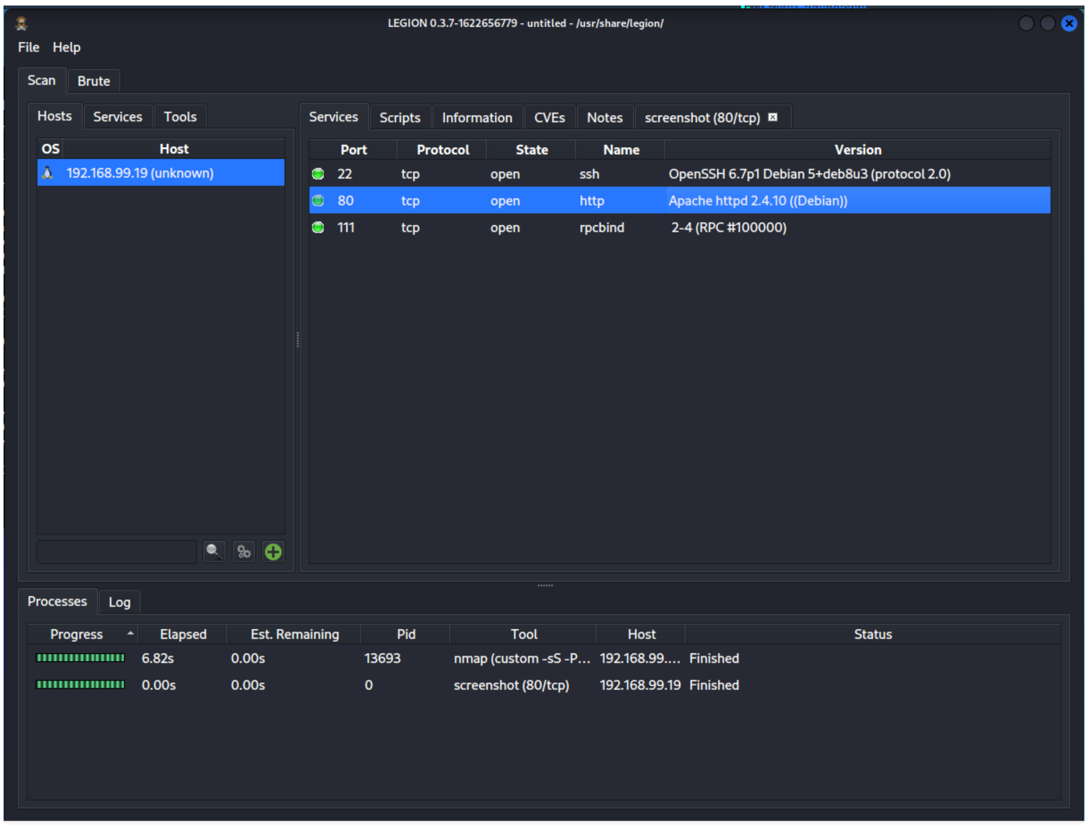
Host Discovery Options: Disable Default ICMP [✓ TCP SYN] TCP ACK Timestamp Netmask

Custom Options

Additional arguments: -sV -O

Submit Cancel

What we see from the output of the scan on the first screen.



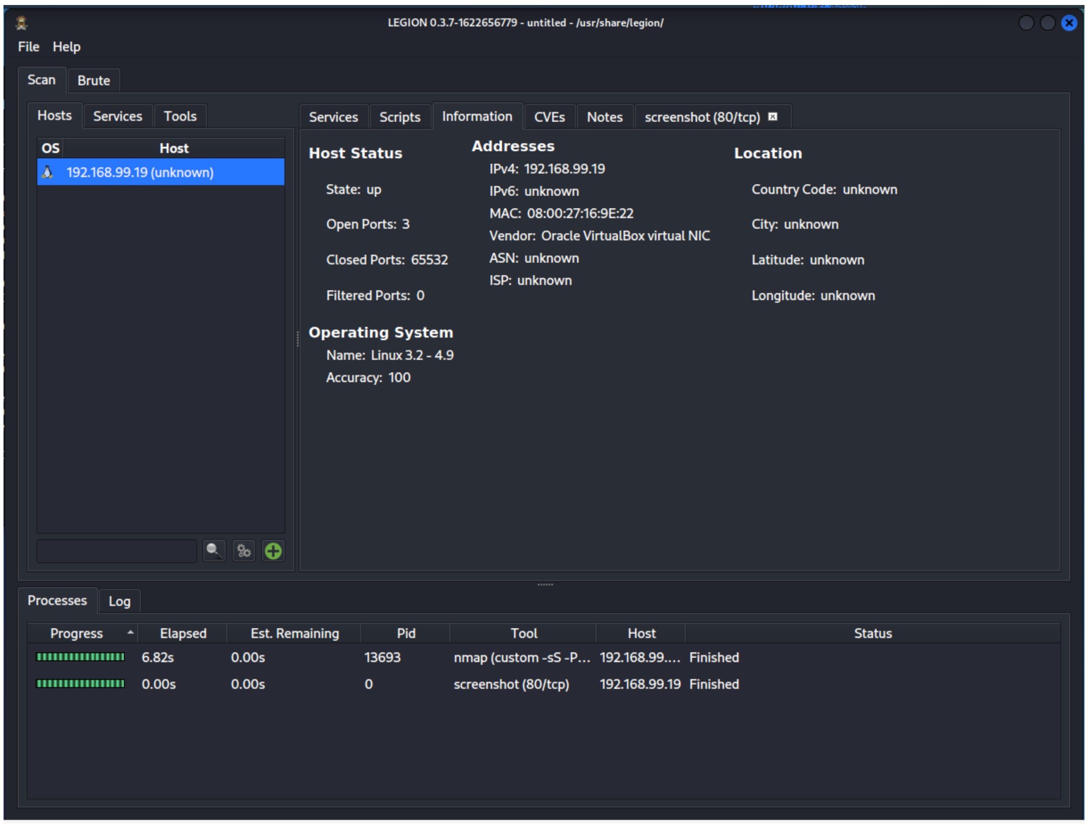
LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/

| OS | Host |
|---|---|
| | 192.168.99.19 (unknown) |

| Port | Protocol | State | Name | Version |
|---|---|---|---|---|
| 22 | tcp | open | ssh | OpenSSH 6.7p1 Debian 5+deb8u3 (protocol 2.0) |
| 80 | tcp | open | http | Apache httpd 2.4.10 ((Debian)) |
| 111 | tcp | open | rpcbind | 2-4 (RPC #100000) |

| Progress | Elapsed | Est. Remaining | Pid | Tool | Host | Status |
|---|---|---|---|---|---|---|
| | 6.82s | 0.00s | 13693 | nmap (custom -sS -P... | 192.168.99.... | Finished |
| | 0.00s | 0.00s | 0 | screenshot (80/tcp) | 192.168.99.19 | Finished |

Switch over to the `Information` tab. This information has me excited if it was complete. Imagine if the `Location` information was properly filled out!


LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/
File Help
Scan Brute
Hosts Services Tools
Services Scripts Information CVEs Notes screenshot (80/tcp)
Processes Log

OS Host
192.168.99.19 (unknown)

Host Status
State: up
Open Ports: 3
Closed Ports: 65532
Filtered Ports: 0

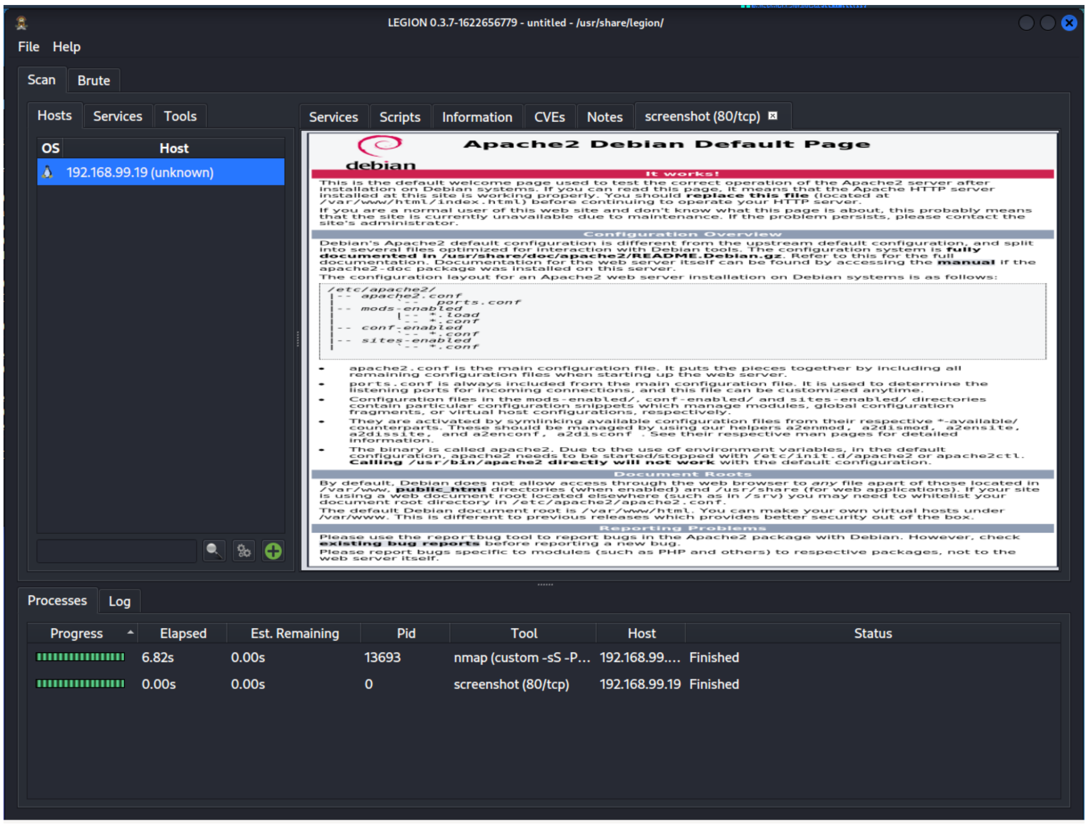
Addresses
IPv4: 192.168.99.19
IPv6: unknown
MAC: 08:00:27:16:9E:22
Vendor: Oracle VirtualBox virtual NIC
ASN: unknown
ISP: unknown

Location
Country Code: unknown
City: unknown
Latitude: unknown
Longitude: unknown

Operating System
Name: Linux 3.2 - 4.9
Accuracy: 100

| Progress | Elapsed | Est. Remaining | Pid | Tool | Host | Status |
|---|---|---|---|---|---|---|
| ▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮ | 6.82s | 0.00s | 13693 | nmap (custom -sS -P… | 192.168.99.… | Finished |
| ▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮ | 0.00s | 0.00s | 0 | screenshot (80/tcp) | 192.168.99.19 | Finished |

The screenshots is an interesting section as well with evidence of what was captured at the time of the scan.



LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/
File Help
Scan Brute
Hosts Services Tools
Services Scripts Information CVEs Notes screenshot (80/tcp)
Processes Log

| OS | Host |
|---|---|
| | 192.168.99.19 (unknown) |

**Apache2 Debian Default Page**

debian

**It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

**Configuration Overview**

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

- apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers a2enmod, a2dismod, a2ensite, a2dissite, and a2enconf, a2disconf . See their respective man pages for detailed information.
- The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with /etc/init.d/apache2 or apache2ctl. **Calling /usr/bin/apache2 directly will not work** with the default configuration.

**Document Roots**

By default, Debian does not allow access through the web browser to *any* file apart of those located in /var/www, **public_html** directories (when enabled) and /usr/share (for web applications). If your site is using a web document root located elsewhere (such as in /srv) you may need to whitelist your document root directory in /etc/apache2/apache2.conf.

The default Debian document root is /var/www/html. You can make your own virtual hosts under /var/www. This is different to previous releases which provides better security out of the box.

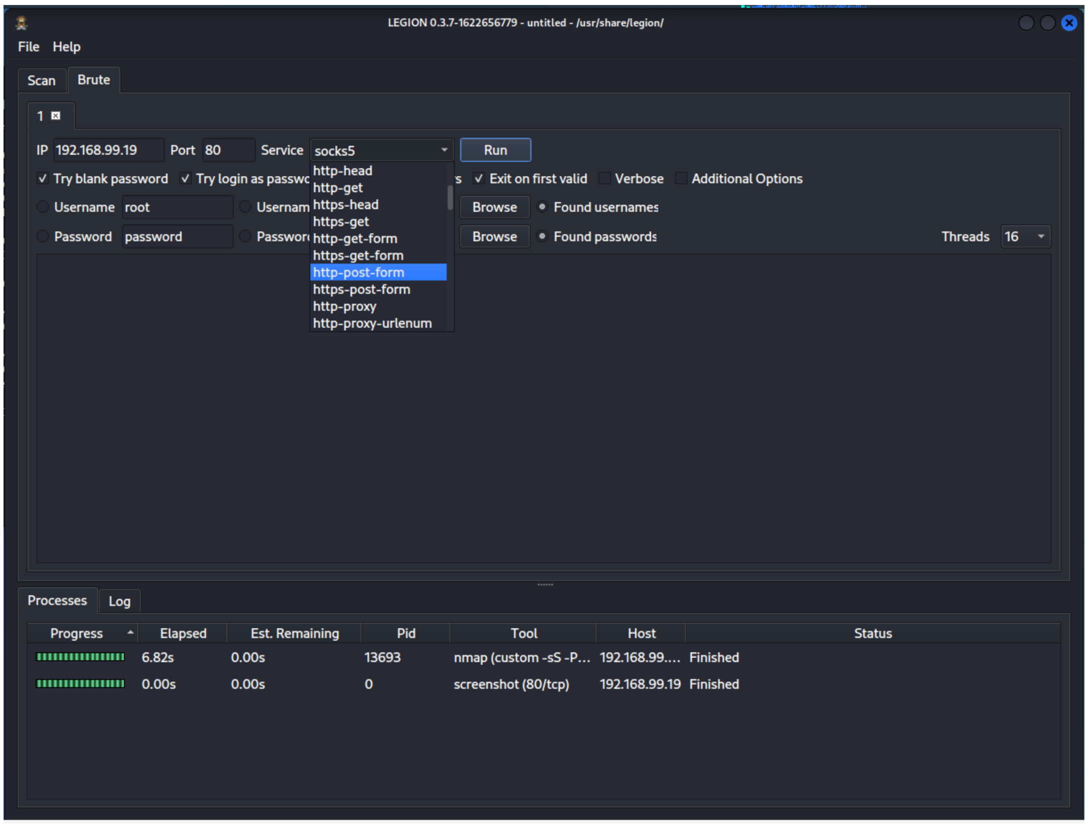
**Reporting Problems**

Please use the reportbug tool to report bugs in the Apache2 package with Debian. However, check **existing bug reports** before reporting a new bug.

Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

| Progress | Elapsed | Est. Remaining | Pid | Tool | Host | Status |
|---|---|---|---|---|---|---|
| ▮▮▮▮▮▮▮▮▮▮ | 6.82s | 0.00s | 13693 | nmap (custom -sS -P… | 192.168.99.… | Finished |
| ▮▮▮▮▮▮▮▮▮▮ | 0.00s | 0.00s | 0 | screenshot (80/tcp) | 192.168.99.19 | Finished |

Under `Brute` force tab for password guessing.



LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/

File Help

Scan Brute

1

IP 192.168.99.19 Port 80 Service socks5 Run

✓ Try blank password ✓ Try login as passwo… s ✓ Exit on first valid Verbose Additional Options

Username root Usernam… Browse Found usernames

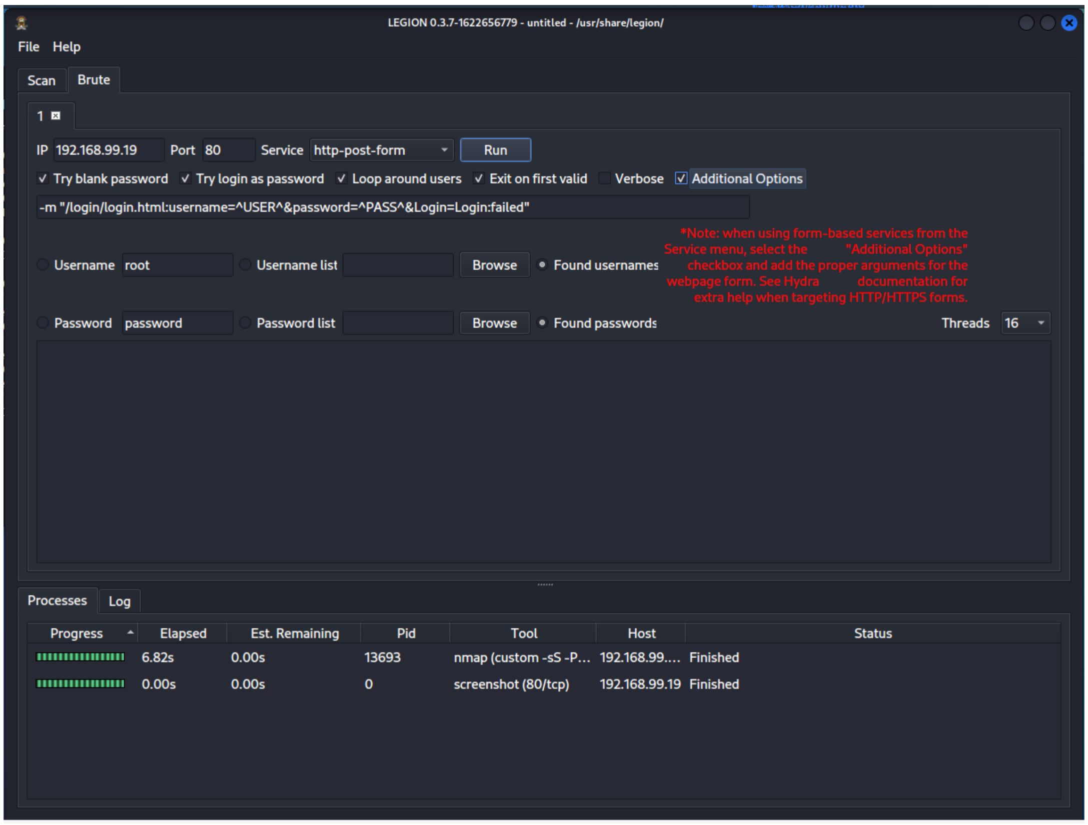
Password password Passwor… Browse Found passwords Threads 16

http-head
http-get
https-head
https-get
http-get-form
https-get-form
http-post-form
https-post-form
http-proxy
http-proxy-urlenum

Processes Log

```
Progress  Elapsed  Est. Remaining  Pid    Tool                  Host           Status
          6.82s    0.00s           13693  nmap (custom -sS -P…  192.168.99.…   Finished
          0.00s    0.00s           0      screenshot (80/tcp)   192.168.99.19  Finished
```

The `Run` portion of the Brute force password guessing.



LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/

IP 192.168.99.19 Port 80 Service http-post-form Run

✓ Try blank password ✓ Try login as password ✓ Loop around users ✓ Exit on first valid Verbose ✓ Additional Options

```
-m "/login/login.html:username=^USER^&password=^PASS^&Login=Login:failed"
```

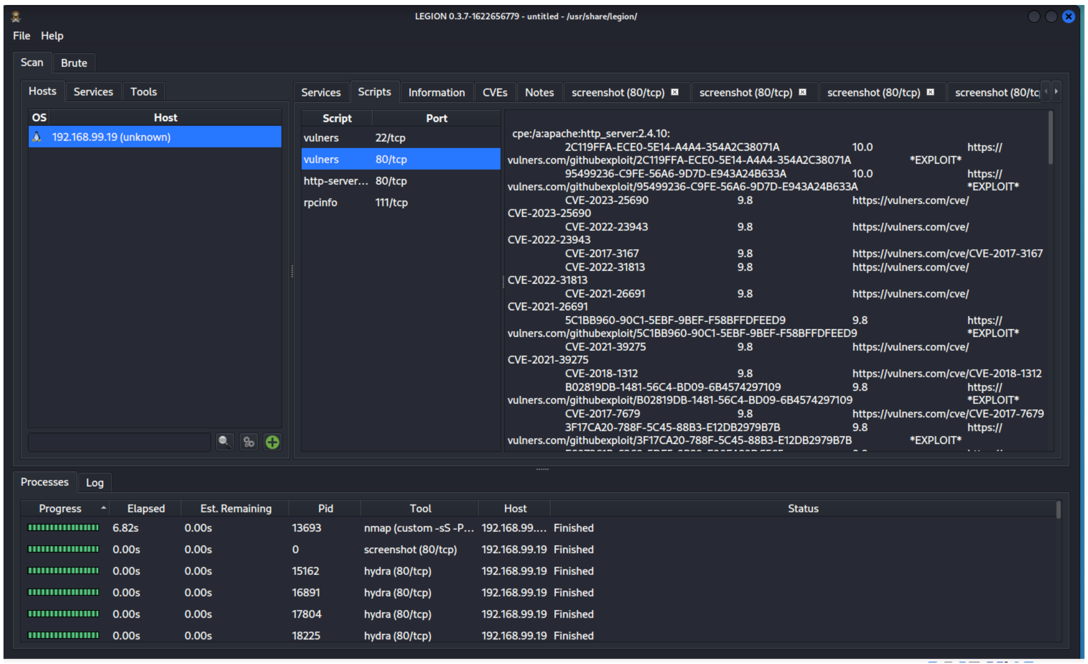
*Note: when using form-based services from the Service menu, select the "Additional Options" checkbox and add the proper arguments for the webpage form. See Hydra documentation for extra help when targeting HTTP/HTTPS forms.

Username root | Username list | Found usernames

Password password | Password list | Found passwords | Threads 16

```
Progress   Elapsed  Est. Remaining  Pid    Tool                 Host          Status
           6.82s    0.00s           13693  nmap (custom -sS -P… 192.168.99.… Finished
           0.00s    0.00s           0      screenshot (80/tcp)  192.168.99.19 Finished
```

Output one from the Brute force.



LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/

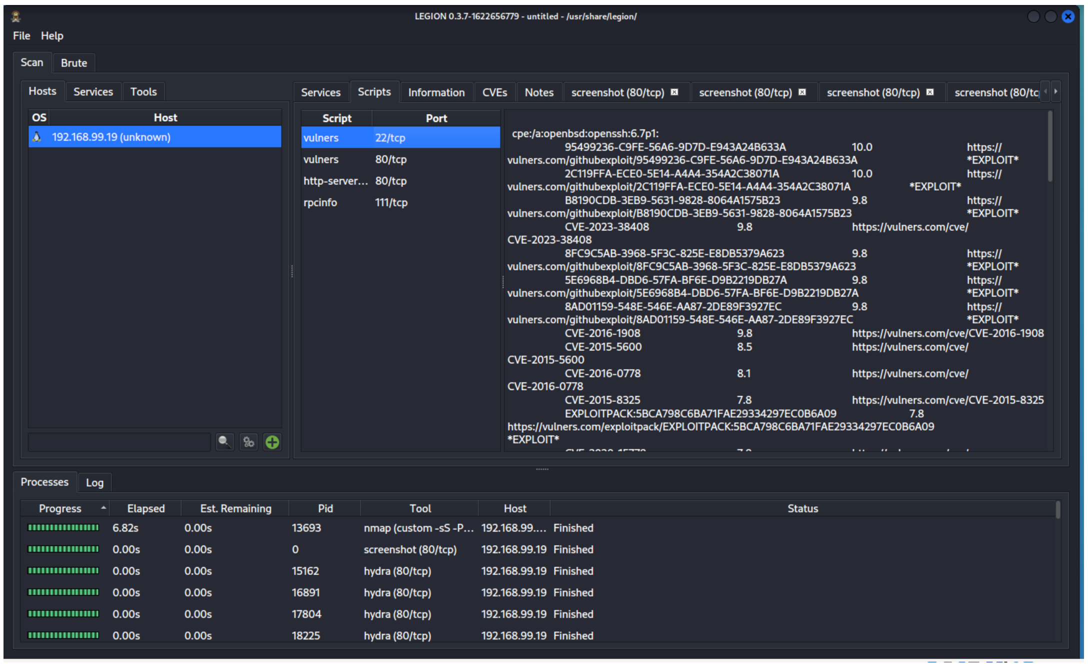
```
OS  Host
    192.168.99.19 (unknown)
```

```
Script          Port
vulners         22/tcp
vulners         80/tcp
http-server…    80/tcp
rpcinfo         111/tcp
```

```
 cpe:/a:apache:http_server:2.4.10:
          2C119FFA-ECE0-5E14-A4A4-354A2C38071A          10.0          https://
vulners.com/githubexploit/2C119FFA-ECE0-5E14-A4A4-354A2C38071A        *EXPLOIT*
          95499236-C9FE-56A6-9D7D-E943A24B633A          10.0          https://
vulners.com/githubexploit/95499236-C9FE-56A6-9D7D-E943A24B633A          *EXPLOIT*
          CVE-2023-25690          9.8          https://vulners.com/cve/
CVE-2023-25690
          CVE-2022-23943          9.8          https://vulners.com/cve/
CVE-2022-23943
          CVE-2017-3167           9.8          https://vulners.com/cve/CVE-2017-3167
          CVE-2022-31813          9.8          https://vulners.com/cve/
CVE-2022-31813
          CVE-2021-26691          9.8          https://vulners.com/cve/
CVE-2021-26691
          5C1BB960-90C1-5EBF-9BEF-F58BFFDFEED9          9.8          https://
vulners.com/githubexploit/5C1BB960-90C1-5EBF-9BEF-F58BFFDFEED9          *EXPLOIT*
          CVE-2021-39275          9.8          https://vulners.com/cve/
CVE-2021-39275
          CVE-2018-1312           9.8          https://vulners.com/cve/CVE-2018-1312
          B02819DB-1481-56C4-BD09-6B4574297109          9.8          https://
vulners.com/githubexploit/B02819DB-1481-56C4-BD09-6B4574297109          *EXPLOIT*
          CVE-2017-7679           9.8          https://vulners.com/cve/CVE-2017-7679
          3F17CA20-788F-5C45-88B3-E12DB2979B7B          9.8          https://
vulners.com/githubexploit/3F17CA20-788F-5C45-88B3-E12DB2979B7B        *EXPLOIT*
```

```
Progress   Elapsed  Est. Remaining  Pid    Tool                 Host          Status
           6.82s    0.00s           13693  nmap (custom -sS -P… 192.168.99.… Finished
           0.00s    0.00s           0      screenshot (80/tcp)  192.168.99.19 Finished
           0.00s    0.00s           15162  hydra (80/tcp)       192.168.99.19 Finished
           0.00s    0.00s           16891  hydra (80/tcp)       192.168.99.19 Finished
           0.00s    0.00s           17804  hydra (80/tcp)       192.168.99.19 Finished
           0.00s    0.00s           18225  hydra (80/tcp)       192.168.99.19 Finished
```

Output two from the Brute force.


LEGION 0.3.7-1622656779 - untitled - /usr/share/legion/
File Help
Scan Brute
Hosts Services Tools
Services Scripts Information CVEs Notes screenshot (80/tcp) screenshot (80/tcp) screenshot (80/tcp) screenshot (80/tc
Processes Log

| OS | Host |
|---|---|
| | 192.168.99.19 (unknown) |

| Script | Port |
|---|---|
| vulners | 22/tcp |
| vulners | 80/tcp |
| http-server… | 80/tcp |
| rpcinfo | 111/tcp |

```
 cpe:/a:openbsd:openssh:6.7p1:
        95499236-C9FE-56A6-9D7D-E943A24B633A          10.0              https://
vulners.com/githubexploit/95499236-C9FE-56A6-9D7D-E943A24B633A           *EXPLOIT*
        2C119FFA-ECE0-5E14-A4A4-354A2C38071A          10.0              https://
vulners.com/githubexploit/2C119FFA-ECE0-5E14-A4A4-354A2C38071A      *EXPLOIT*
        B8190CDB-3EB9-5631-9828-8064A1575B23          9.8               https://
vulners.com/githubexploit/B8190CDB-3EB9-5631-9828-8064A1575B23           *EXPLOIT*
        CVE-2023-38408              9.8              https://vulners.com/cve/
CVE-2023-38408
        8FC9C5AB-3968-5F3C-825E-E8DB5379A623          9.8               https://
vulners.com/githubexploit/8FC9C5AB-3968-5F3C-825E-E8DB5379A623           *EXPLOIT*
        5E6968B4-DBD6-57FA-BF6E-D9B2219DB27A          9.8               https://
vulners.com/githubexploit/5E6968B4-DBD6-57FA-BF6E-D9B2219DB27A           *EXPLOIT*
        8AD01159-548E-546E-AA87-2DE89F3927EC          9.8               https://
vulners.com/githubexploit/8AD01159-548E-546E-AA87-2DE89F3927EC           *EXPLOIT*
        CVE-2016-1908               9.8              https://vulners.com/cve/CVE-2016-1908
        CVE-2015-5600               8.5              https://vulners.com/cve/
CVE-2015-5600
        CVE-2016-0778               8.1              https://vulners.com/cve/
CVE-2016-0778
        CVE-2015-8325               7.8              https://vulners.com/cve/CVE-2015-8325
        EXPLOITPACK:5BCA798C6BA71FAE29334297EC0B6A09            7.8
https://vulners.com/exploitpack/EXPLOITPACK:5BCA798C6BA71FAE29334297EC0B6A09
*EXPLOIT*
```

| Progress | Elapsed | Est. Remaining | Pid | Tool | Host | Status |
|---|---|---|---|---|---|---|
| 🟩 | 6.82s | 0.00s | 13693 | nmap (custom -sS -P… | 192.168.99.… | Finished |
| 🟩 | 0.00s | 0.00s | 0 | screenshot (80/tcp) | 192.168.99.19 | Finished |
| 🟩 | 0.00s | 0.00s | 15162 | hydra (80/tcp) | 192.168.99.19 | Finished |
| 🟩 | 0.00s | 0.00s | 16891 | hydra (80/tcp) | 192.168.99.19 | Finished |
| 🟩 | 0.00s | 0.00s | 17804 | hydra (80/tcp) | 192.168.99.19 | Finished |
| 🟩 | 0.00s | 0.00s | 18225 | hydra (80/tcp) | 192.168.99.19 | Finished |

# Chapter 4. Conclusion

By employing the above methodology, we have successfully established a secure sandbox lab environment to evaluate and comprehend one of Kali Linux's tools.

The Legion tool integrates both the enumeration and scanning phases into a single operation. As a graphical tool, it offers numerous features that would be desirable in a professional setting, including:

- Nmap scanning of open ports and identifying responsive target systems.
- Automatic detection of Common Platform Enumerations (CPEs) and Common Vulnerabilities and Exposures (CVEs).

The author envisions Information System Security Officers within their respective organizations utilizing this tool to efficiently assess the security posture of their assigned subnets. This would allow them to verify compliance with security requirements prior to using more comprehensive tools that officially document the results. Essentially, this is an ideal tool for quickly obtaining results and conducting spot checks to identify any emerging issues within their areas of responsibility.

The author highly recommends this tool, humorously noting that it receives two thumbs up, which is the maximum count the Author currently possess.

Thank you for taking the time to read this white paper. As we conclude, we ask that you prepare for the end by metaphorically returning to your seat and securing any loose thoughts. Please fasten your mental seatbelt, ensuring your focus is in an upright and attentive position.

As you close this paper, we hope you retain the insights shared. The author sincerely appreciates your time and attention, knowing that you have countless options for reading material online. Thank you once again for choosing this white paper.

# Chapter 5. Appendix

*References*

https://www.kali.org/tools/legion/

https://gitlab.com/kalilinux/packages/legion

https://www.hackingloops.com/legion-framework/